Preface

This handbook contains information regarding all aspects of the Undergraduate Computer Science Programs offered by the Department of Computer Science, Memorial University of Newfoundland --from requirements for admission to our programs to degree requirements for graduation. It should prove to be a useful source of information for potential computer science majors as well as for those already in our program.

This handbook is to be considered a guide and not a legal document. Students are advised to consult the 2020-2021 university Calendar regarding specific degree regulations and university policies. For further information regarding the Department, please consult our website at www.mun.ca/computerscience or contact our general office located in the S.J. Carew Building, room EN-2021, or write us at the following address:

Head
Department of Computer Science
S.J. Carew Building, EN-2021
Memorial University of Newfoundland
St. John's, NL
Canada, A1B 3X5
E-mail: compsci@mun.ca

In order to consult with our Manager of Academic Programs regarding our programs and courses, please e-mail cs-ugradadv@mun.ca
# Table of Contents

1  General Information .......................................................................................................................... 1

1.1  The Computer Science Student .................................................................................................. 1

1.2  Improving Undergraduate Program Accessibility ...................................................................... 1

1.3  Computer Science Program Offerings ....................................................................................... 1

1.4  Admission to the Computer Science Program ........................................................................... 2

1.4.1  Eligibility .................................................................................................................................. 2

1.4.2  How to Apply ........................................................................................................................... 3

1.5  Selecting Your Courses ............................................................................................................... 3

1.5.1  Courses for Non-Majors ......................................................................................................... 3

1.5.2  Minor in Computer Science ................................................................................................... 4

1.5.3  Computer Science Majors ...................................................................................................... 5

1.5.4  First Year Course Selection (B.Sc.) ..................................................................................... 5

1.5.5  First Year Course Selection (B.A.) ....................................................................................... 6

1.6  Ethics of Computer Science Students ......................................................................................... 7

1.7  Information Required in Certificates from Health Professionals ............................................... 7

1.8  Computer Science Student Help Centre ..................................................................................... 8

1.9  Fees ............................................................................................................................................. 8

1.10  Job Opportunities and Continued Study .................................................................................... 8

1.11  Faculty Listing ........................................................................................................................... 9

1.12  Staff Listing ................................................................................................................................ 10

2  Basic Structural Information ........................................................................................................... 11

2.1  Prerequisite Structure of Core Courses ...................................................................................... 11

2.2  Prerequisite Structure of Required Mathematics Courses ......................................................... 30

2.3  Course Offerings ......................................................................................................................... 30

2.4  Undergraduate Programs involving Computer Science .............................................................. 32

2.4.1  Major in Computer Science .................................................................................................. 32

2.4.2  Major in Computer Science (Smart Systems) (B.Sc. only) .................................................... 32

2.4.3  Major in Computer Science (Visual Computing and Games) (B.Sc. only) ......................... 32

2.4.4  Honours in Computer Science ............................................................................................... 33

2.4.5  Honours in Computer Science (Software Engineering) (B.Sc. only) .................................... 34

2.4.6  Minor in Computer Science .................................................................................................. 34
| 2.4.7 | Co-operative Internship in Computer Science (CICS) | 35 |
| 2.4.8 | Computer Science and Statistics Joint Major | 36 |
| 2.4.9 | Computer Science and Statistics Joint Honours (B.Sc. only) | 37 |
| 2.4.10 | Computer Science and Pure Mathematics Joint Major | 37 |
| 2.4.11 | Computer Science and Pure Mathematics Joint Honours | 37 |
| 2.4.12 | Computer Science and Applied Mathematics Joint Major | 38 |
| 2.4.13 | Computer Science and Geography Joint Major | 38 |
| 2.4.14 | Computer Science and Geography Joint Honours | 38 |
| 2.4.15 | Computer Science and Physics Joint Major (B.Sc. only) | 39 |
| 2.4.16 | Computer Science and Physics Joint Honours (B.Sc. only) | 39 |
| 2.4.17 | Computer Science and Economics Joint Major | 40 |
| 2.4.18 | Joint Degrees of Bachelor of Science and Bachelor of Arts | 40 |
| 2.5 | General Degree Regulations | 41 |
| 2.6 | Second Degree in Computer Science | 41 |
| 2.7 | Deferred Exams | 42 |
| 3 | Description of Undergraduate Courses | 43 |
| 3.1 | 1000-Level Courses | 43 |
| 3.2 | 2000-Level Courses | 55 |
| 3.3 | 3000-Level Courses | 74 |
| 3.4 | 4000-Level Courses | 97 |
1 General Information

This section is an introduction to activities and resources available in the Department. It is intended to help students pursuing or wishing to pursue undergraduate studies in the Department of Computer Science.

1.1 The Computer Science Student

We strive to turn our students into good problem solvers—whose problem solving tools are the technology and ideas of computation. Whether your interest is game programming, media design, mathematics, science, humanities, or computer systems design, every discipline—practical, theoretical or creative—is experiencing an ever-increasing dependence on computer technology.

Algorithmic problem solving—creating solutions (algorithms) that can be executed on a machine—is fundamental to Computer Science. With a grounding in the fundamentals, a student can continue on to explore computer systems, software architecture, web applications, databases, computer networks, scientific applications, computer graphics, games systems, numerical applications, and any other subject for which computers have an application. Our mission in the Computer Science Department is to provide opportunities for students to explore their computing interests, and one way we pursue this objective is by offering a selection of required and elective courses covering topics of interest and importance. It is our hope that this handbook will reflect this objective, and that you will find this material useful in matching your own interests in computing.

The Department of Computer Science also provides a number of venues for activities, including seminars by prominent computer scientists, programming competitions and a student-run Undergraduate Computer Science Society.

1.2 Improving Undergraduate Program Accessibility

We have been introducing changes to make our program more accessible. There are courses available for non-majors, including one about the history of computing (Computer Science 1400), one contrasting depictions of various aspects of computing in movies and documentaries produced over the last 60 years (Computer Science 1401), one in multimedia programming (Computer Science 2300) and one in computation as a means of understanding the world around us (Computer Science 2000), and distance offerings for the information technology course (Computer Science 1600). Computer Science 2510 (Programming in C/C++) is available to both majors and non-majors.

If you have suggestions for developing our program, or you want to know if we have something to fit your interests, please come talk to us.

1.3 Computer Science Program Offerings

The Department of Computer Science has a wide range of offerings, incorporating hardware, software, and the theory and practice of computing. Students with a strong interest in another major subject can consider one of the joint degrees available in specific disciplines, or a minor in Computer Science to supplement their skills in their major discipline. If a minor will not fit your
program of studies, a few or even a single course in computing may be worthwhile addition(s) to your education.

The Department offers a number of joint programs, the majority of which are available for both majors and honours. Joint programs allow for a concentration of courses in computer science as well as in another discipline of the student's choice. Joint programs are available with the Departments of Mathematics and Statistics, Geography, Physics and Physical Oceanography, and Economics. Specialized programs include a Software Engineering Honours and a Co-operative Internship Option.

The undergraduate programs offered by the Department of Computer Science include:

- Major and Honours in Computer Science
- Major in Computer Science (Smart Systems) (B.Sc. only)
- Major in Computer Science (Visual Computing and Games) (B.Sc. only)
- Honours in Computer Science (Software Engineering, B.Sc. only)
- Co-operative Internship in Computer Science (CICS)
- Minor in Computer Science
- Computer Science Joint Major with either
  - Applied Mathematics
  - Pure Mathematics
  - Statistics
  - Physics
  - Economics, or
  - Geography
- Computer Science Joint Honours with either
  - Pure Mathematics
  - Statistics
  - Physics, or
  - Geography

Detailed information on each of these programs is available later in this handbook and in the university Calendar.

Our Manager of Academic Programs can provide information and academic advice. An incoming student enrolling as a computer science major is strongly advised to contact the Manager of Academic Programs when preparing their course schedules each semester, in dropping or adding courses during a semester, or whenever a need warrants. This can be done through email correspondence at mailto:ugradadv@mun.ca. Note that the final decision on the selection of courses is the responsibility of the student.

1.4 Admission to the Computer Science Program

Admission to the Computer Science Major program is competitive and selective. There is only one intake of Computer Science Majors each year. Students who would like to be considered for admission must submit an online major application form prior to June 1.

1.4.1 Eligibility

To be eligible for admission students must normally have completed 24 credit hours as listed below:
1. Computer Science 1001, 1002.
2. Six credit hours in Critical Reading and Writing (CRW) courses, including at least 3 credit hours in English courses.
3. Mathematics 1000 and 1001, or 1090 and 1000.
4. Six credit hours in other courses.

Students who fulfill the eligibility requirements compete for a limited number of available spaces. Selection is based on academic performance, normally cumulative average and performance in recent courses.

1.4.2 How to Apply

There is only one intake of Computer Science Majors each year, students who would like to be considered for admission must submit an online major application form prior to June 1.

A separate application is required for entrance into an honours program, normally at the beginning of the third year of studies. Acceptance of students into an honours program is based on their academic performance in the Computer Science courses completed at the time of application. Students intending to pursue graduate work in computer science are strongly encouraged to undertake an honours degree.

1.5 Selecting Your Courses

1.5.1 Courses for Non-Majors

The Computer Science Department offers a number of individual courses designed to appeal to non-majors who want to explore and develop their computing skills and knowledge:

- Computer Science 1000 is a general introduction to computer science and programming. In this course you will learn basic programming skills and find out what computer science is about. It has no prerequisite;
- Computer Science 1400 gives an overview of the development of computing technologies over the last 75 years. It has no prerequisite;
- Computer Science 1401 contrasts depictions of various aspects of computing in various movies and documentaries produced over the last 60 years. It has no prerequisite;
- Computer Science 1510 is about numerical problem solving and programming skills, directed at students majoring in the mathematical sciences, including mathematics, physics, and chemistry. It has Mathematics 1000 as its prerequisite;
- Computer Science 1600 gives students an understanding of basic concepts and necessary skills required to use spreadsheet, database, and presentation software and to manage, analyze and present data. It has no prerequisite;
- Computer Science 2000 explores the world from a computational perspective. Robots, the Internet, the human mind, biological growth, language and logic are investigated in a series of classroom and lab experiences. This course is recommended for B.A. students but is open to everyone. It has no prerequisite;
• Computer Science 2300 is a course in multimedia programming. You can build interactive animations for the web while learning to program. It has Computer Science 1003 as its prerequisite; and

• Computer Science 2510 gives a comprehensive treatment of the C/C++ programming languages. Computer Science 2718 enables students to improve their data management and data processing tasks. Both of these courses are intended for students with some first programming experience.

1.5.2 Minor in Computer Science

A Computer Science minor supplements any degree with career enhancing skills. Our minor is very flexible, allowing students to tailor their courses to a particular interest. For example, it is possible to accommodate a specific interest with courses related to computer graphics, software development, scientific computing, computer networking, artificial intelligence, robotics, information systems, mobile computing or games development.

For a Minor in Computer Science, a student must complete at least 24 credit hours in Computer Science courses, including:

3. Three additional credit hours at the 3000 level or above.
4. Additional courses as necessary, at the 2000-level or above, to fulfill the requirement for 24 credit hours in Computer Science.

In addition to the required courses listed above (i.e. 1001, 1002, 1003, 2001), here are some examples of courses that could be chosen to complete a minor depending on a student’s interest.

• Interest in Software Development
  2002 - Data Structures and Algorithms
  2005 - Software Engineering
  2006 - Computer Networking (1 credit hour)
  2007 - Introduction to Information Management (1 credit hour)
  2008 - Social Issues and Professional Practice (1 credit hour)
  3718 - Programming in the Small
  4768 - Software Development for Mobile Devices

• Interest in Numerical or Scientific Applications
  2002 - Data Structures and Algorithms
  2005 - Software Engineering
  2500 - Data Analysis with Scripting Languages
  3731 - Introduction to Scientific Computing
  4734 - Matrix Computations and Applications

• Interest in Theoretical Computer Science
  2002 - Data Structures and Algorithms
  2003 - Computer Architecture
  3602 – Introduction to the Theory of Computation
  4742 - Computational Complexity
  4743 - Graph Algorithms and Combinatorial Optimization
• Interest in Information Systems
  2002 - Data Structures and Algorithms
  2006 - Computer Networking (1 credit hour)
  2007 - Introduction to Information Management (1 credit hour)
  2008 - Social Issues and Professional Practice (1 credit hour)

• Interest in Games Programming
  2002 - Data Structures and Algorithms
  2003 - Computer Architecture
  2300 - Introduction to Multimedia Programming
  3300 - Interactive Technologies
  3600 – Algorithm Design and Analysis

1.5.3 Computer Science Majors

Computer science degree programs, like most general degree programs in the Faculty of Science or the Faculty of Humanities and Social Sciences, specify approximately half of the courses required for your degree. The required computer science courses are those felt by the Department to constitute the minimum which could be expected of any student in computer science. A great deal of your program, therefore, will be made up of elective courses, both in computer science and other disciplines. These electives can be chosen from many areas, subject to the general regulations for the Faculty of Science or the Faculty of Humanities and Social Sciences. Those general regulations can be found in the university Calendar. The quality and usefulness of your degree to both yourself and prospective employers depend, to some extent, on the elective courses that you choose. A general recommendation is to try to have a concentration of courses in one or two other areas: for example, an area in which you have a professional interest (an area in which you wish to work in the future) or an area in which you have a personal interest. In order to accomplish this, we recommend that you choose some area as a minor, and work to fulfil the requirements for a minor in that area as specified in the university Calendar.

The computer science programs offered by the Department of Computer Science keep pace with advancement in the discipline, and provide a well-balanced foundation in the science of information processing. Our programs cover all aspects and levels of computers and computing. We have expertise in a variety of areas, including artificial intelligence, bio-inspired computing, complexity theory, computer architecture, computer graphics and image processing, database systems, design and analysis of algorithms, distributed computing, human-computer interaction, information systems, parallel processing, pattern recognition, programming languages, robotics, scientific computing, software engineering, design and web-based systems. Computer science is a rapidly evolving discipline; the program at Memorial emphasizes the fundamental ideas and methods underlying the discipline.

Degree programs in computer science have considerable flexibility. It is your responsibility to ensure that you fulfil all the requirements of your chosen program. In your sixth semester or earlier, you should request a degree audit from the Registrar’s Office to determine your program status. This can be done online through Self-Service, or by emailing science_registrar@mun.ca (for B.Sc. Audits) or arts_registrar@mun.ca (for B.A. audits).

1.5.4 First Year Course Selection (B.Sc.)

A candidate for the B.Sc. degree shall complete the core requirements, which consist of the following:
(a) Six credit hours in Critical Reading and Writing (CRW) courses, including at least three credit hours in English courses,
(b) Six credit hours in Mathematics and Statistics courses, and
(c) Six credit hours in each of two sciences other than Mathematics and Statistics.

The following table shows possible first year course selections for B.Sc.:

<table>
<thead>
<tr>
<th>Sample Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semester I</td>
</tr>
<tr>
<td>Mathematics 1090 or 1000*</td>
</tr>
<tr>
<td>Computer Science 1001</td>
</tr>
<tr>
<td>Science elective</td>
</tr>
<tr>
<td>English 1090</td>
</tr>
<tr>
<td>Elective</td>
</tr>
</tbody>
</table>

*Students completing Mathematics 1090/1000 will be required to complete Mathematics 1001 as well.

**Students who have not completed Computer Science 1003 in their first year will not be able to register for Computer Science 2001/2002/2003 in the fall of their second year.

1.5.5 First Year Course Selection (B.A.)

A candidate for the B.A. degree shall complete the core requirements, which consist of the following:

(a) Minimum two designated courses for the Critical Reading and Writing (CRW) requirement, one of which must be a three-credit hour 1000-level CRW course offered by English.
(b) Minimum two designated courses for the Language Study (LS) requirement.
(c) Minimum two designated courses for the Quantitative Reasoning (QR) requirement.
(d) Courses from a minimum of six Faculty of Humanities and Social Sciences disciplines (Breadth of Knowledge).
(e) A minor program (or, alternatively, a second major).

The following table shows possible first year course selections:

<table>
<thead>
<tr>
<th>Sample program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semester I</td>
</tr>
<tr>
<td>English 1090</td>
</tr>
<tr>
<td>Language study (LS) or Breadth of Knowledge course</td>
</tr>
<tr>
<td>Mathematics 1090 or 1000*</td>
</tr>
<tr>
<td>Computer Science 1001</td>
</tr>
</tbody>
</table>
Elective course (breadth encouraged) | Computer Science 1003**

*Students completing Mathematics 1090/1000 will be required to complete Mathematics 1001 as well.

**Students who have not completed Computer Science 1003 in their first year will not be able to register for Computer Science 2001/2002/2003 in the fall of their second year.

1.6 Ethics of Computer Science Students

Students are requested to check their MUN email accounts frequently, and to always use this account when contacting the University. Throughout the semester, announcements concerning social functions, part-time jobs, scholarships and general items of interest to majors may be emailed. If the Department needs to contact a particular student, we will do so via email. Inappropriate use of the email could result in a student losing computer privileges.

The Department uses homework assignments both as a teaching device and as a major component of its assessment of each student. **It, therefore, requires that all programs, assignments, etc. submitted bearing an individual student's name be the work of that student alone.** Under normal circumstances, students may discuss assignments but may not jointly write solutions. In particular, direct copying of another student's assignment is regarded as cheating by all parties knowingly involved, and will be dealt with as such, following University Regulations on Academic Misconduct.

Also, devices, electronic or otherwise, are not permitted to be used in the room during examinations unless specifically allowed by your instructor. Under no circumstances will students be allowed to access phones or devices during examination periods.

Computer science students are expected to maintain a high degree of honesty, dignity and respect for their fellow students, faculty and associates, particularly while using the computing facilities. Students are advised to refer to the General Academic Regulations (Undergraduate) of the university Calendar for further information regarding academic misconduct.

1.7 Information Required in Certificates from Health Professionals

A student who requests permission to drop courses, to withdraw from university studies, to have examinations deferred or to obtain other waivers of University, departmental or course regulations based on health issues is required by the University to provide, in support of the request, a certificate from a health professional in the form of a note or letter. While not compulsory, this should normally be in the form of the Student Health Certificate, available at www.mun.ca/regoff/forms.php. If a note or letter is provided other than in the form of the Student Health Certificate, the submission must be on letterhead. Such certificates must be sufficiently specific to allow a proper consideration of a student's case. The University requires that all such certificates must be signed by the health professional, must confirm the specific dates on which the student visited the health professional and should include details on the following:

- The degree to which the health issue (or treatment, in the case of medication, for example) is likely to have affected the student’s ability to study, attend classes, or sit examinations;
• The length of time over which the student’s abilities were likely hampered by the condition (e.g., recurring and severe back pain over a two-month period would likely have a more adverse effect on studies than a single episode of back pain requiring bed rest for a week);

• The fitness of the student to resume studies (it is in the student’s best interest not to return to studies prematurely).

Confidentiality: The University respects the privacy of students and will keep confidential all such certificates. A student should request that the health professional retain a copy of such a certificate in case the certificate needs to be verified or reissued at a later date.

1.8  Computer Science Student Help Centre

Normally the Computer Science Student Help Centre is located in room EN-2031C of the S.J. Carew Building. Due to the COVID-19 pandemic, the Help Centre will be available remotely as a Brightspace shell (much like a course) for the Fall 2020 semester.

All students enrolled in 1000- and 2000-level lab-based Computer Science courses will have access to the Help Centre Brightspace shell. A schedule for the virtual Help Centre will be posted on the Computer Science website (www.mun.ca/computerscience) and within the Brightspace shell. Students will be able to drop in anytime the virtual Help Centre is open and the staff will answer questions in a first-come, first-served order. Alternatively, students can ask questions outside of the scheduled open hours and these questions will be answered the next time someone is staffing the virtual Help Centre.

1.9  Fees

Tuition Fees:
See current university Calendar. http://www.mun.ca/regoff/calendar/

Books and Supplies:
Students should be prepared to meet an expenditure to cover the cost of textbooks and supplies each semester. Students are cautioned to check edition number and printing date of textbooks and manuals prior to purchasing second-hand books; these books may have become obsolete. Within the Department of Computer Science, there is a charge for use of the printers.

1.10  Job Opportunities and Continued Study

Students graduating from Memorial with either a B.Sc. or a B.A. general degree, majoring in computer science, have many job opportunities available to them in Newfoundland and Labrador as well as throughout Canada and internationally. There is an ever-increasing demand for programmers and analysts in all aspects of data processing as well as in scientific applications. There are numerous career opportunities in hardware and software design, software and technical support, sales, management and various levels of education.

Students who choose to do an honours degree in computer science have the same job opportunities available to them as those with a general degree, and, in addition, have better opportunities to pursue further study in specialized areas of computer science at the M.Sc. or Ph.D. levels. Many Canadian universities, including Memorial, recommend an honours degree or equivalent in computer science or a closely related discipline for entrance to the M.Sc. program.
1.11 Faculty Listing

Following is a list of faculty members and their areas of interest/research. If you want to get a glimpse of what research in computer science is about, take some time to visit the respective faculty member’s website or www.mun.ca/computerscience/research. Find out more about the fascination and challenges computer science offers to you.

M. Bartha - programming language semantics, algebra and category theory in computer science, automata theory, graph theory

E. Brown - human-computer interaction, hypertext, scientific visualization

S. Bungay - genetic algorithms, mathematical modelling, numerical methods, optimization techniques, physiological systems, computational chemistry, dynamical systems

R. Byrne - software architecture for embedded systems

Y. Chen - computer networking, distributed computing, combinatorial optimization, approximation algorithms and heuristics, graph theory

D. Churchill - artificial intelligence, video games, heuristic search, reinforcement learning, neural networks, autonomous robotics

A. Fiech - programming languages, lambda calculus, polymorphism, domain theory

R. Gupta - e-learning, microcomputer based information systems, database systems, computer science education

M. Hatcher - programming languages, discrete-event simulation, artificial life

T. Hu - evolutionary computing, bioinformatics, complex networks, machine learning

X. Jiang - intelligent human machine interaction, human behaviour recognition, bio-signal processing, eye-tracking and pupil diameter, human factors

A. Kolokolova - theoretical computer science, complexity theory, mathematical logic

M. Mata-Montero - theoretical computer science, serial and parallel computational complexity

S. Mehrabi - theoretical computer science, algorithms and data structures, computational geometry

O. Meruvia-Pastor - interactive 3D graphics, non-photorealistic rendering, multimedia application development, biomedical visualization

G. Miminis - scientific computing, numerical methods in control engineering, numerical methods for vector and parallel architectures

L. Peña-Castillo - data analysis of large scale “omic” data, bioinformatics, machine learning

V. Prado da Fonseca – robotic manipulation, bio-inspired robotics
A. **Soares** - mobility data analysis, data analytics, data mining and knowledge discovery, machine learning, artificial intelligence, computer graphics

**J. Tang** - database systems, distributed computing, fault-tolerant computing, design and analysis of algorithms, data mining

A. **Vardy** - robot navigation, bio-inspired computing

K. **Vidyasankar** - distributed computing, database systems, graph theory

T. **Wareham** - computational biology, algorithmic design and analysis

### 1.12 Staff Listing

**Instructional Support Staff:**
- Stephen Anthony, [stephen@mun.ca](mailto:stephen@mun.ca) EN-1062 (Laboratory Instructor)
- Cindy Milley, [cindy@mun.ca](mailto:cindy@mun.ca) EN-1064
- Wendong Mao, [wm0330@mun.ca](mailto:wm0330@mun.ca) EN-1063

**Computer Support Staff:**
- Michael Rayment, [mtr@mun.ca](mailto:mtr@mun.ca) EN-1050 (Systems Manager)
- Aaron Casey, [casey@mun.ca](mailto:casey@mun.ca) EN-1051A
- Andrew Draskoy, [andrew@mun.ca](mailto:andrew@mun.ca) EN-1057
- Kate Gilbert, [krgilbert@mun.ca](mailto:krgilbert@mun.ca) EN-1051A
- Marian Wissink, [marian@mun.ca](mailto:marian@mun.ca) EN-1059

**Manager of Academic Programs:**
- Cathy Hyde, [cs-ugradadv@mun.ca](mailto:cs-ugradadv@mun.ca) EN-2025

**Internship Contact:**
- Rebecca Newhook, Academic Staff Member in Co-operative Education SN-1062
- Theresa Mackenzie, Academic Staff Member in Co-operative Education SN-1060.

**Our General Office is located in the S.J. Carew Building, room EN-2021 and is staffed by:**
- Jennifer Friesen, [jfriesen@mun.ca](mailto:jfriesen@mun.ca) EN-2020
- Regina Edwards, [redwards@mun.ca](mailto:redwards@mun.ca) EN-2021B
- Barbara Hynes, [bjhynes@mun.ca](mailto:bjhynes@mun.ca) EN-2021
2 Basic Structural Information

This section describes the structural relationship of computer science courses and the requirements leading towards the fulfilment of an undergraduate computer science degree.

2.1 Prerequisite Structure of Core Courses

Notes:
- Students are advised to check the specific prerequisite(s) of each elective course.
- Computer Science 2006, 2007 and 2008 are one credit-hour courses.
2.2 Prerequisite Structure of Required Mathematics Courses

Math 1090
or Math 109B

Math 1000

Math 1001

Math 2050

Stat 1510

Stat 2550

Math 2000

Corequisite

Prerequisite

*Math 1000 and Math 2050 can be taken concurrently (with appropriate prerequisites).

2.3 Course Offerings

A selection of the following computer science courses is normally offered during the academic year. Students should consult the departmental website, the postings on the bulletin board outside the general office, or contact the Academic Program Manager (cs-ugradadv@mun.ca) to determine special topics courses to be offered in any given semester. (Note: Core courses are in bold font.)

COMP 1000 - Computer Science - An Introduction
COMP 1001 - Introduction to Programming
COMP 1002 - Introduction to Logic for Computer Scientists
COMP 1003 - Foundations of Computing Systems
COMP 1400 - Computing in the 20th Century and Beyond
COMP 1401 - Computing at the Movies
COMP 1510 - An Introduction to Programming for Scientific Computing
COMP 1600 - Basic Computing and Information Technology
COMP 2000 - Collaborative and Emergent Behaviour
COMP 2001 - Object-Oriented Programming and Human-Computer Interaction
COMP 2002 - Data Structures and Algorithms
COMP 2003 - Computer Architecture
COMP 2004 - Introduction to Operating Systems
COMP 2005 - Software Engineering
COMP 2006 - Computer Networking
COMP 2007 - Introduction to Information Management
COMP 2008 - Social Issues and Professional Practice
COMP 2100 - Social Web Analysis
COMP 2300 - Introduction to Multimedia Programming
COMP 2500 - Data Analysis with Scripting Languages
COMP 2510 - Programming in C/C++
COMP 2718 - Development Tools, Work Flows and Concepts
COMP 3200 - Algorithmic Techniques for Artificial Intelligence
COMP 3201 - Introduction to Nature-Inspired Computing
COMP 3202 - Introduction to Machine Learning
COMP 3300 - Interactive Technologies
COMP 3301 - Visual Computing and Applications
COMP 3401 - Introduction to Data Mining
COMP 3550 - Introduction to Bioinformatics
COMP 3600 - Algorithm Design and Analysis
COMP 3602 - Introduction to the Theory of Computation
COMP 3700 - Industrial Experience
COMP 3710 - Vocational Languages
COMP 3718 - Programming in the Small
COMP 3731 - Introduction to Scientific Computing
COMP 3753 - Computational Aspects of Linear Programming
COMP 4300 - Introduction to Game Programming
COMP 4301 - Computer Vision
COMP 4302 - 3D Computer Graphics
COMP 4303 - Artificial Intelligence in Computer Games
COMP 4304 - Data Visualization
COMP 4550 - Bioinformatics: Biological Data Analysis
COMP 4711 - Structure of Programming Languages
COMP 4712 - Compiler Construction
COMP 4715 and 4717 - Special Topics in Programming Languages
COMP 4718 - Survey of Software Engineering
COMP 4721 - Operating Systems
COMP 4723 - Introduction to Microprocessors
COMP 4726 to 4729 - Special Topics in Computer Systems
COMP 4734 - Matrix Computations and Applications
COMP 4736 to 4739 - Special Topics in Numerical Computations
COMP 4741 - Formal Languages and Computability
COMP 4742 - Computational Complexity
COMP 4743 - Graph Algorithms and Combinatorial Optimization
COMP 4745 to 4749 (excluding 4748) - Special Topics in Theoretical Aspects
COMP 4750 - Introduction to Natural Language Processing
COMP 4754 - Database Systems
COMP 4759 - Computer Networks
COMP 4766 - Introduction to Autonomous Robotics
COMP 4768 - Software Development for Mobile Devices
COMP 4770 - Team Project
COMP 4780 - Honours Project
COMP 4800 to 4825 - Special Topics
2.4 Undergraduate Programs involving Computer Science

2.4.1 Major in Computer Science

As a component of the Degree Regulations for the General Degree of Bachelor of Science or the Degree Regulations for the General Degree of Arts, as appropriate, a student must complete the following courses:

1. Forty-five credit hours in computer science courses are required for a major:
   (b) At least six additional credit hours in computer science at the 4000 level.
   (c) Twelve additional credit hours in computer science at the 3000 level or beyond.

2. Additional courses required are: Mathematics 1000, 1001, 2000, 2050, and Statistics 1510 or 2550.

Notes:

• Students are encouraged to take Mathematics 3000 and Statistics 2560.
• A computer science major is encouraged to take a concentration of courses in another discipline in order to gain a broader background. An optional minor is available with a Bachelor of Science degree.

2.4.2 Major in Computer Science (Smart Systems) (B.Sc. only)

As a component of the Degree Regulations for the General Degree of Bachelor of Science, a student must complete the following courses:

1. Forty-five credit hours in Computer Science courses are required for a major in Computer Science (Smart Systems):
   (b) Computer Science 3200, 3201, 3202 and 3301; and
   (c) Six additional credit hours in Computer Science courses selected from Computer Science 3401, 3550, 4301, 4303, 4750, 4766.

2. Additional courses required are: Mathematics 1000, 1001, 2000, 2050, and Statistics 1510 or 2550.

2.4.3 Major in Computer Science (Visual Computing and Games) (B.Sc. only)

As a component of the Degree Regulations for the General Degree of Bachelor of Science, a student must complete the following courses:

1. Forty-five credit hours in Computer Science courses are required for a major in Computer Science (Visual Computing and Games):
   (b) Computer Science 3300, 3301, and 4300;
(c) Six additional credit hours in Computer Science courses selected from Computer Science 2300, 4301, 4302, 4303, 4304; and
(d) Three additional credit hours in Computer Science courses selected from those listed in (c) above, or Computer Science 2100, 4766, 4768.

2. Additional courses required are: Mathematics 1000, 1001, 2000, 2050, and Statistics 1510 or 2550.

2.4.4 Honours in Computer Science

Students must make a separate application for entrance into an honours program. This is normally done at the beginning of the third year of studies. Acceptance into an honours program is based on a student’s academic performance in the computer science courses completed at the time of application.

1. (a) In order to graduate with a Bachelor of Science (Honours) degree, a candidate shall obtain:

   (i) A grade of “B” or better, OR an average of 75% or higher (whichever is to the candidate’s advantage) in the minimum number of courses in the Honours subject (or subjects) prescribed by the Department (or, in the case of joint Honours, Departments) concerned, excluding the 1000-level courses,

   AND

   (ii) An average of at least 2.75 points on the total number of courses required for the degree.

(b) In order to graduate with a Bachelor of Arts (Honours) degree, a student shall obtain:

   (i) A grade of 70% or better, or an average of 75% or higher in the minimum number of courses (including the required courses in the Honours subject(s)) prescribed by the Department (or, in the case of Joint Honours, Departments) concerned. A grade of 70% or better must be obtained in the Honours project,

   AND

   (ii) An average of at least 2.75 points on the total number of credit hours in the courses required for the degree.

Notes:

- Students who wish to fulfil the requirements of Clause (i) above using repeated or substituted courses must obtain approval of the Head of the Department and the Committee on Undergraduate Studies. No more than three such repeated or substituted courses will be permitted.
- See also General Regulations for Honours Degree (B.A. or B.Sc., as appropriate) in the university Calendar.

2. Sixty-three credit hours in computer science courses are required for the honours degree in computer science:


   (b) Fifteen additional credit hours in computer science at the 4000 level.
Eighteen additional credit hours in computer science at the 3000 level or beyond.

3. Additional courses required are Mathematics 1000, 1001, 2000, 2050 and Statistics 1510 or 2550.

Notes:
- Students are encouraged to take Mathematics 3000 and Statistics 2560.
- A computer science honours student is encouraged to take a concentration of courses in another discipline in order to gain a broader background. An optional minor is available with a Bachelor of Science degree.

2.4.5 Honours in Computer Science (Software Engineering) (B.Sc. only)

Completion of the honours in computer science (software engineering) program does not qualify persons to hold the designation “Professional Engineer" as defined by various provincial acts governing the engineering profession.

1. See Section 1 (a) under Honours in Computer Science.

2. Sixty-three credit hours in computer science courses are required for the honours degree in computer science (software engineering), including:
   (b) Nine additional credit hours in computer science chosen from 4718, 4721, 4723, the former 4751, the former 4753, the former 4756, 4759, 4766 and 4768.
   (c) Nine additional credit hours in computer science at the 4000 level.
   (d) Twelve additional credit hours in computer science at the 3000 level or beyond.

3. Additional courses required are: Mathematics 1000, 1001, 2000, 2050, and Statistics 1510 or 2550.

Note: The honours project (Computer Science 4780) must be in the area of software engineering.

2.4.6 Minor in Computer Science

For a minor in computer science, a student must complete at least 24 credit hours in computer science courses, including:


3. Three additional credit hours in computer science courses at the 3000 level or above.

4. Additional courses as necessary, at the 2000 level or above, to fulfil the requirement for 24 credit hours in Computer Science.
2.4.7 **Co-operative Internship in Computer Science (CICS)**

The CICS provides an opportunity for qualified students to obtain rewarding placements that help them develop practical skills in a real work setting before graduation. The CICS is available to computer science majors who will typically apply between their third and fourth year of studies.

**Admission Requirements**

In order to be considered for admission to the CICS, an applicant:

- Must be a declared computer science major
- Must be registered as full-time student at the time of application.
- Must have at least 15 credit hours remaining after the internship in order to satisfy degree requirements, three of which must be in computer science, and
- Is expected to return to University as a full-time student after the internship.

In addition to the above, admission to the CICS is also subject to academic performance.

**Internship Duration**

Subject to the availability of job openings, a student may complete either an 8, 12 or 16 consecutive month internship period.

**Internship Guidelines**

- Internship employment is normally organized by Co-operative Education; however, students who have been accepted to the CICS may also obtain their own internship placements. All placements are subject to the approval of Co-operative Education and of the Head of the Department of Computer Science.

- Students who have applied to the internship program give permission to Co-operative Education to supply prospective employers with copies of their resume and transcript.

- After being placed with an employer, students are not permitted to drop their internship without prior approval from Co-operative Education and the Head of the Department of Computer Science. Students who drop an internship without permission, who fail to honour an agreement to work with an employer, or who conduct themselves in such a manner as to cause their discharge from the placements, will normally be awarded a fail grade for the internship period and may not be permitted to reapply.

**Note:** Students should also refer to the University Regulations – General Academic Regulations (Undergraduate).

**Registration, Assessment of Performance, and Assignment of Grades**

Students must register for the course Computer Science 3700 every semester during their internship. Computer Science 3700 is a non-credit course open only to students who have been accepted into the internship program.

During the internship the employer and intern will complete student performance evaluations every four months and will submit them to Co-operative Education. The final assessment of total work performed is the responsibility of Co-operative Education, and will be based upon
both input from the employer and Co-operative Education’s assessment of the intern’s written assignments.

The Internship evaluation shall consist of two components:

1. On-the-job Student Performance: Job performance shall be assessed by Co-operative Education in consultation with the Department using information gathered during the internship and input from the employer. Evaluation of the on-the-job student performance will result in one of the following classifications: PASS WITH DISTINCTION, PASS, FAIL.

2. Internship Report(s): Evaluation of the internship report will result in one of the following classifications: PASS WITH DISTINCTION, PASS, FAIL.

The evaluation of the on-the-job student performance and the internship report(s) are recorded separately on the transcript. Overall evaluation of the internship will result in one of the following final grades being awarded:

PASS WITH DISTINCTION indicates outstanding performance in both the internship report(s) and the on-the-job student performance. PASS WITH DISTINCTION has been awarded to each of the internship report(s) and the on-the-job student performance.

PASS indicates that performance meets expectations in both the internship report(s) and on-the-job student performance. The student meets the requirements of a passing mark in the internship report and on-the-job student performance.

FAIL indicates failing performance in either the internship report(s) or on-the-job student performance or both.

Also, the following will be noted on the transcript of the intern:

- Requirements for the Co-operative Internship in Computer Science have been completed. Internship Duration: - months.
- A grade of NC (No Credit) for Computer Science 3700 will be awarded in all semesters of the Co-operative Internship prior to the final semester.

CICS and Honours Program
In case a student is enrolled in both the Honours program and the CICS, the requirements of both must be met. Upon approval from the honours project supervisor, within the Department, the employer and the Head of the Department of Computer Science, an internship project may be submitted as a component of an honours project. These arrangements must be made within the first semester of the Internship placement.

2.4.8 Computer Science and Statistics Joint Major
As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:


(b) Statistics 1510 or 2500 or 2550, and 2501 or 2560.
(c) Mathematics 1000, 1001, 2000, 2050, 2051, 2320, 3340, Statistics 2410 or 3410, 3411, 3520, 3521, 3540, and 4590.

(d) Nine further credit hours in statistics courses numbered 3000 or higher including at least a three credit hour course numbered 4000 or higher, excluding Statistics 4581.

2.4.9 Computer Science and Statistics Joint Honours (B.Sc. only)

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:

1. Mathematics 1000, 1001, 2000, 2050, 2051, 2320, 3340, Statistics 1510 or 2500 or 2550, 2410 or 3410, 2501 or 2560, 3411, 3520, 3521, 3540, 4530, 4590.

2. Eighteen further credit hours in statistics courses including at least 12 credit hours in courses numbered 4000 or higher, but not including Statistics 4581 and 459A/B.


4. Twenty-one additional credit hours in computer science courses at the 3000 level or higher, not including Computer Science 4780.

5. Either Computer Science 4780 or Statistics 459A/B.

2.4.10 Computer Science and Pure Mathematics Joint Major

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:


(b) Eighteen additional credit hours in computer science numbered 3000 or higher.

(c) Mathematics 1000, 1001, 2000, 2050, 2051, 2130, 2260, 2320, 3000, 3202, 3320, 3340 and Statistics 2550.

(e) Nine additional credit hours in courses numbered 3000 or higher offered by the Department of Mathematics and Statistics, excluding Mathematics 3330.

2.4.11 Computer Science and Pure Mathematics Joint Honours

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:

1. At least 51 credit hours in computer science are required including the following:


   (b) Excluding Computer Science 4780, 24 additional credit hours from courses numbered 3000 or higher, at least nine credit hours of which must be in courses at the 4000 level.

2. The following courses in mathematics and statistics are required:
(a) Mathematics 1000, 1001, 2000, 2050, 2051, 2130, 2260, 2320, 3000, 3001, 3202, 3210, 3320, 3340, Statistics 2550;

(b) Either Mathematics 4000 or 4001;

(c) Excluding the former Mathematics 3330, the former 4399, and 439A/B, 15 additional credit hours in courses offered by the Department of Mathematics and Statistics numbered 3000 or higher including at least nine credit hours from courses numbered 4000 or higher and at least nine credit hours in pure mathematics courses;

(d) An honours dissertation in one of the departments, with the topic chosen in consultation with both departments.

2.4.12 Computer Science and Applied Mathematics Joint Major

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:


(b) Mathematics 1000, 1001, 2000, 2050, 2051, 2130, 2260, 2320, 3000, 3100, 3132, 3161, 3202, 4160 and 4190.

In addition, Statistics 2550 is highly recommended.

2.4.13 Computer Science and Geography Joint Major

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:


2. Geography requirements: Thirty-nine credit hours in geography courses are required: 1050, 2001, 2102, 2195, 2302, 2425, 3202, 3222, 3250, 3260, 4202, 4250, 4261.


2.4.14 Computer Science and Geography Joint Honours

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:

1. Computer Science Requirements

Forty-eight credit hours in computer science courses are required for the joint honours:

(b) Six additional credit hours in courses at the 4000 level not including Computer Science 4780.

(c) Twelve additional credit hours in courses at the 3000 level or beyond.

2. Geography Requirements

Forty-eight credit hours in geography courses are required for the joint honours: 1050, 2001, 2102, 2195, 2226, 2302, 2425, 3202, 3222, 3226, 3250, 3260, 3303, 4202, 4250, 4261 and the former 4291.

3. Additional Requirements

(a) Mathematics 1000, 1001, 2000 and 2050.

(b) An honours dissertation (either Computer Science 4780 or Geography 4999). The topic for dissertation must be chosen with the prior approval of the Heads of both departments.

2.4.15 Computer Science and Physics Joint Major (B.Sc. only)

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:

1. Chemistry 1050 and 1051 (or Chemistry 1010, 1011 and the former 1031).


3. Physics 1050 (or 1020) and 1051 plus at least 30 additional credit hours in Physics including 2053, 2055, 2750, 2820, 3220, 3400, 3500, 3750, 3800.

4. (a) Mathematics 1000 and 1001.

(b) Mathematics 2000, 2050, 2260, 3202.

(c) Additional electives to bring the credit hours to 120. Computer Science 2500 and Statistics 2550 are recommended.

2.4.16 Computer Science and Physics Joint Honours (B.Sc. only)

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:

1. Chemistry 1050 and 1051 (or Chemistry 1010, 1011, and the former 1031) (or 1200 and 1001).

(b) Nine additional credit hours in Computer Science courses numbered 3000 or higher, including at least 3 credit hours in courses at the 4000 level.

3. (a) Physics 1050 (or 1020) and 1051.

(b) Physics 2053, 2055, 2750, 2820, 3220, 3400, 3500, 3750, 3800, and 3820.

(c) Three additional credit hours in Physics at the 4000 level.

4. Physics 490A and Physics 490B or Computer Science 4780, and 3 additional credit hours in Computer Science at the 4000 level.

5. (a) Mathematics 1000 and 1001.

(b) Mathematics 2000, 2050, 2260, and 3202.

6. Six credit hours in Critical Reading and Writing (CRW) courses, including at least 3 credit hours in English courses.

7. Two electives to bring the total credit hours to 120. Computer Science 2500 and Statistics 2550 are recommended.

The topic for the honours project or thesis, Computer Science 4780 or Physics 490A/B, must be chosen with the prior approval of both departments.

2.4.17 Computer Science and Economics Joint Major

As a component of the Degree Regulations for the General Degree of Bachelor of Science, the following courses are required:

1. Computer Science requirements:


2. Economics requirements:

A total of 42 credit hours in economics courses is required: 1010 (or the former 2010), 1020 (or the former 2020), 2550, 3000, 3001, 3010, and 6 credit hours from either 3550 and 3551, or 4550 and 4551 are obligatory.

The remaining 18 credit hours shall be chosen from among the various economics courses in consultation with the Head of the Department or delegate, and will include at least nine credit hours in courses at the 4000-level.

3. Additional requirements:


2.4.18 Joint Degrees of Bachelor of Science and Bachelor of Arts

Students who wish to simultaneously complete a Bachelor of Science program and a Bachelor of Arts program may do so by completing a minimum of 135 credit hours in courses and are not
required to complete a minor. See the university *Calendar*, Faculty of Science entry for the Joint Degrees of Bachelor of Science and Bachelor of Arts.

2.5 **General Degree Regulations**

Students are encouraged to familiarize themselves with the general degree requirements (as printed in the *Calendar*) pertaining to their chosen faculty (science, or humanities and social sciences).

In order to graduate with the General Degree of Science a student shall have:

(a) Satisfied the conditions of General Academic Regulations (Undergraduate);

(b) Obtained an average of at least 2.0 points in the 78 credit hours in Science required for the degree;

(c) Obtained an average of at least 2.0 points in the minimum number of prescribed courses in the major subject(s).

In order to graduate with the Bachelor of Arts General Degree, a student shall obtain:

(a) An average of 60% or higher on the minimum number of courses prescribed for the Major program, and

(b) An average of 60% or higher on the minimum number of courses prescribed for the Minor program.

2.6 **Second Degree in Computer Science**

A student can receive a Bachelor’s degree in computer science from Memorial, even if they already have a Bachelor’s degree in another subject from Memorial. In order to receive a second degree majoring in computer science, a student must consider the following regulations:

(a) Section 6.2.3 of the General Academic Regulations (Undergraduate) of the university *Calendar* states: “A student will not be awarded the same Bachelor’s degree more than once by this university.” Thus if a student already has a Bachelor of Science degree, then they must complete the computer science degree requirements for a Bachelor of Arts degree. If they already have a Bachelor of Arts degree, then they must complete the requirements for a Bachelor of Science degree. A student can also convert a general degree into an honours degree (e.g., B.Sc. to B.Sc. Honours).

(b) Section 6.3.3 of the General Academic Regulations (Undergraduate) of the university *Calendar* states: “Every student for a second Bachelor’s degree shall complete at least 30 credit hours at this university beyond those required for the first degree. These credits must be applicable to the degree sought.” This means that a second degree can be obtained with a minimum of 30 extra credit hours. However, meeting the requirements for the computer science degree can result in taking more than 30 extra credit hours.

Any student wishing to take computer science as a second degree should consult with the Department of Computer Science to ensure that all regulations can be met as efficiently as possible.
2.7 Deferred Exams

(a) Final Examinations: Please refer to Section 6.8.2 of the General Academic Regulations (Undergraduate) of the university Calendar.

(b) Midterms and Term Tests: Please note that deferred midterms and term tests are at the discretion of the instructor of each course.
3 Description of Undergraduate Courses

Throughout the following sections, an asterisk * placed after a course number indicates that there are other prerequisites. (Note: Core courses are in bold font.)

3.1 1000-Level Courses

COMP 1000 Computer Science - An Introduction
COMP 1001 Introduction to Programming
COMP 1002 Introduction to Logic for Computer Scientists
COMP 1003 Foundations of Computing Systems
COMP 1400 Computing in the 20th Century and Beyond
COMP 1401 Computing at the Movies
COMP 1510 An Introduction to Programming for Scientific Computing
COMP 1600 Basic Computing and Information Technology
COMP 1000
Computer Science - An Introduction

Students Interested
This course is an introductory course to computer science for students who are not completing a computer science major or minor.

Objectives of the Course
This course gives students an overview of computer science providing them with a foundation from which they can better appreciate and understand computing in everyday life.

Prerequisite(s)
None

Successor(s)
None

Representative Workload
<table>
<thead>
<tr>
<th>Workload</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td>10%</td>
</tr>
<tr>
<td>Lab Quizzes</td>
<td>20%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>45%</td>
</tr>
</tbody>
</table>

Representative Course Outline
- Algorithms and Programming (9 lectures)
- Inside the Dream Machine (9 lectures)
  - Number Systems, Digital Circuits, Computer Organization, System Software
- Techniques and Applications (14 lectures)
  - Databases, Networks, Security, Theoretical Computer Science, Artificial Intelligence

Comments or Notes
- In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable (approximately 6 labs per semester).
- Credit cannot be obtained for both Computer Science 1000 and the former Computer Science 2742, plus Computer Science 1003 uni-directional credit, Engineering 4424 and Mathematics 2320.
- Credit cannot be received for Computer Science 1000 if Computer Science 1003 has already been completed.
COMP 1001
Introduction to Programming

Students Interested
This course is the first course in programming for all computer science majors and minors.

Objectives of the Course
This course is an introduction to fundamental programming techniques, primitive data types, and simple algorithms and their design concepts.

Prerequisite(s)
None

Successor(s)

Representative Workload
Assignments 20%
Lab Quizzes 15%
Test(s) 20%
Final Exam 45%

Representative Course Outline
- Fundamental programming constructs: data types, variables, operations, expressions, assignment statements, input, output, selection, repetition, functions, file input/output (12 hours)
- Algorithms and problem solving (4 hours)
- Fundamental data structures: strings, arrays, lists, sets and dictionaries (6 hours)
- Object-oriented programming: objects, methods, operator overloading, inheritance, method overriding, polymorphism (6 hours)
- Recursion (3 hours)
- Exception handling (2 hours)
- Searching (linear, binary) and sorting (insertion, selection, bubble) (2 hours)
- Linked lists (3 hours)

Comments or Notes
- In addition to three hours hour lectures there is a structured laboratory as scheduled in the university timetable.
- Credit cannot be obtained for both Computer Science 1001 and Computer Science 1710.
COMP 1002
Introduction to Logic for Computer Scientists

Students Interested
This course is an introduction to discrete structures for all computer science majors.

Objectives of the Course
The objective of this course is to provide basic understanding of logic and discrete structures used throughout computer science, with the focus on computer science-specific applications such as Boolean circuits and basic algorithm analysis.

Prerequisite(s)
None

Successor(s)
COMP 2002*, COMP 2003*

Representative Workload
Quizzes 30%
Assignments 40%
Final Exam 30%

Representative Course Outline
• Propositional and predicate logic (9 hours)
• Set theory, functions, relations and incomputability (4 hours)
• Proof techniques including induction (10 hours)
• Basic counting and modular arithmetic (5 hours)

Comments or Notes
• In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
• Credit cannot be obtained for both Computer Science 1002 and either of Computer Science 2742 or Engineering 4424.
• Students cannot received credit for Computer Science 1002, if completed with, or subsequent to, Mathematics 2320.
COMP 1003
Foundations of Computing Systems

Students Interested
This course is a follow up to a course in computer programming and would be of interest to students who want to pursue a degree in computer science or to those who are interested in learning foundational ideas in the science of computing.

Objectives of the Course
The objective of this course is to provide a solid introduction to foundational topics in computer science: algorithms and data structures, theory of computing, machine architecture and their historical context.

Prerequisite(s)
COMP 1001

Corequisite(s)
COMP 1002 or Mathematics 2320

Successor(s)

Representative Workload
Assignments 10%
Lab Quizzes 10%
Midterm Exam 25%
Final Exam 35%

Representative Course Outline
- Algorithms and data structures (4 weeks)
  - Abstract data types and their implementation; using, designing and creating data types
  - Algorithms for sorting and searching
  - Fundamental data structures and their performance characteristics
- Theory of computing (4 weeks)
  - Alphabets, formal languages and their descriptions
  - Abstract machines, finite state automata, Turing machines
  - Universality
  - Computability
  - Intractability
• Machine architecture (4 weeks)
  - From abstract machines to “real computers”
  - Representation of information, characters, integers and floating point numbers
  - Machine language programming
  - Virtual machines
  - Boolean logic and the circuit model
  - Combinational circuits, gates, multiplexors, decoders, arithmetic and logic unit
  - Sequential circuits, flip-flops, memory, registers, clock
  - Digital devices, program counter, control, central processing unit

Comments or Notes
• In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
Students Interested
Given the ubiquitous nature of computing in everyday life as well as the increasing number of reports in the news of computing-related calamities, the material in this course will give members of the general student population the tools to understand how computing impacts them and what they can do to both embrace the potential and mitigate the risks of applying computing technologies in their professional and private lives.

Objectives of the Course
This course will give an overview of the development of computing technologies over the last 75 years as well as both the perception of these technologies by, and their impact on, society. The course will be organized chronologically by decade, and within each decade will examine the dominant computing developments, their image in various print and pictorial media, and their social impact. The aim is to give students of all disciplines an appreciation of the abilities and limitations of computer technology and how such technologies interact with society.

Prerequisite(s)
None

Successor(s)
None

Representative Workload
In-class Exams (4) 100%

Representative Course Outline
In each course unit below, material will be organized by, and presented sequentially, as three themes: Technology (e.g., processing/memory/I/O/network), Applications (e.g., areas of application/user-group/level of knowledge required), and Impact (e.g., social/economic issues and problems raised by technology, depictions of technology and its creators and users in the media).

- How We Got Here: Computing in the Past (6 weeks)
  (In the Beginning (pre-1940); The First Computers (1940-1955); The Rise of the Machines (1955-1970); The Personal Computer Revolution (1970-1990); The Wired Society (1990-now))

- Where We Are: Computing Now (4 weeks)
  (An examination of selected issues in current computing, e.g., life online, big data and privacy, robots and artificial intelligence)

- Where We’re Going: Computing in the Future (1 week)
Students Interested
Misconceptions about the nature, abilities, and limitations of computing devices, as well as the computing profession and those within it, are widespread – fostered in large part by (mis)representations of computing in print and audiovisual media. Such misconceptions can have far-reaching consequences given the increasing prominence of computing in personal, commercial, and political life. The main objective of this course is to critically examine these misconceptions through viewing, discussing, and writing about representations of computing in various movies and documentaries produced over the last 60 years. A secondary, but nonetheless still important objective, will be to stimulate interest in computing in the context of a non-technical and easily accessible introduction to computing and the computing profession.

Objectives of the Course
This course will both examine and counter common misconceptions about computing and the computing profession. This will be done by contrasting depictions of various aspects of computing in various movies and documentaries produced over the last 60 years with the reality of these aspects as given in selected readings and course lecture notes.

Prerequisite(s)
None

Successor(s)
None

Representative Workload
Course Participation 10 %
Critical Film Commentaries (2) 30%
In-class Exams (2) 30%
Term Paper 30%

Representative Course Outline
This course will be taught once a week in a three-hour lecture slot. Several days before each lecture, introductory notes on the area of computer science treated in the film will be provided on the course website; the expectation is that this content will be read before the lecture. In each lecture, there will be a brief (15-25 minute) introduction by the instructor before the screening of that week's film. The screening will be followed by further notes from the instructor and class discussion.

Over the course of the term, the students will submit 2 four-page (double-spaced) papers on two different films presented in the course lectures (one from the films in weeks 1-5, the other from the films in weeks 7-11) in which a critical assessment is given of both the (in)accuracy of the portrayal of computing in that film as well as how the film affects the public's perception of computing. Each student will also submit an 8-10 page (double-spaced) term paper assessing,
comparing, and contrasting three of the films shown in the course plus one other film selected in consultation with the instructor.

The films will follow a roughly chronological course from 1955 to the present, with each film touching on one or more of the following three broad themes: Computing in the Popular Imagination (CI), Computing and Society (CS), and the Computing Profession (CP).

A sample set of films and associated readings is as follows:

Week 1: Introduction: Computing and Hollywood v1.0
   Presented Viewing: Westworld (1973) (CI1)

Week 2: Inside the Dream Machine

Week 3: Computing and the Workplace
   Presented Viewing: Desk Set (1957) (CS1)

Week 4: Computing and the Military

Week 5: Computing and Medicine
   Presented Viewing: The Terminal Man (1974) (CS2)

Week 6: The Personal Computer Revolution
   Presented Viewing: Pirates of Silicon Valley (TV) (1999) (CP1)

Week 7: The Hacker Mystique

Week 8: Computing and Privacy

Week 9: The Business of Computing I

Week 10: The Wired World

Week 11: Artificial Intelligence

Week 12: The Business of Computing II
   Presented Viewing: Revolution OS (2001) (CP4)
COMP 1510
An Introduction to Programming for Scientific Computing

Students Interested
Those who are interested in learning a programming language, as well as various numerical methods relevant to scientific computing. Numerical methods to solve selected problems from Physics, Chemistry and Mathematics will be covered.

Objectives of the Course
To introduce students to basic programming in the context of numerical methods, with the goal of providing the foundation necessary to handle larger scientific programming projects.

Prerequisite(s)
Mathematics 1000

Successor(s)
COMP 2500, COMP 2510, Mathematics 2130*

Representative Workload
Lab Quizzes / Assignments 40%
Midterm Exam 30%
Final Exam 30%

Representative Course Outline
• Computer terminology and fundamental concepts, problem solving, floating point arithmetic
• Programming in Fortran 90
data types, expressions, I/O, formatted I/O, if statement, logical operators, loops, arrays, subprograms
• Numerical methods to solve selected problems from Physics, Chemistry, and Mathematics.
• Programming in C
data types, expressions, I/O, formatted I/O, if statement, logical operators, loops, arrays, subprograms

Comments or Notes
• In addition to three one-hour lectures, there is a structured laboratory as scheduled in the university timetable. Laboratory sections will meet for the first two and one-half hours of the laboratory slots.
• Students can receive credit for only one of Computer Science 1510 or the former Computer Science 2602.
• Students who have received credit for the former Applied Mathematics 2120 cannot receive credit for Computer Science 1510.
COMP 1600
Basic Computing and Information Technology

Students Interested
Any students who will benefit from learning the basic concepts and skills required for information management and data analysis.

Objectives of the Course
To provide students with an understanding of basic concepts and necessary skills required to use spreadsheet, database and presentation software to manage, analyze and present data.

Prerequisite(s)
None

Successor(s)
None

Representative Workload

<table>
<thead>
<tr>
<th></th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments (5)</td>
<td>15%</td>
</tr>
<tr>
<td>Quizzes/Labs (9)</td>
<td>9%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>56%</td>
</tr>
</tbody>
</table>

Representative Course Outline

- Use Windows operating system to organize files and manage files
- Use MS-Excel software to
  - understand the basics of electronic spreadsheets
  - process data from large multi-sheet workbooks by performing simple and/or complex calculations with built-in functions and formula
  - make decisions, find desired outcomes and graph data
  - summarize and analyze data
  - design workbooks
- Use MS-Access software to
  - understand the basics of a database and Database Management System (DBMS)
  - design simple and complex queries to retrieve the desired information from a database
  - implement security and integrity for a database
  - design a database
- Use MS-PowerPoint to create efficient presentations
- Integrate data from more than one application (MS-Word, MS-Excel, MS-Access and MS-PowerPoint)
• Labs
  - Lab 1 - System Software
  - Lab 2 - Spreadsheet I
  - Lab 3 - Spreadsheet II
  - Lab 4 - Spreadsheet III
  - Lab 5 - Spreadsheet IV
  - Lab 6 - Database I
  - Lab 7 - Database II
  - Lab 8 - Database III
  - Lab 9 - Presentation Software and Integration

Comments or Notes
• In addition to three one-hour lectures, there is a structured laboratory as scheduled in the university timetable.
• Students can receive credit for only one of Computer Science 1600, the former Computer Science 2650, the former Computer Science 2801, or the former Business 2700.
3.2 2000-Level Courses

- COMP 2000 - Collaborative and Emergent Behaviour
- COMP 2001 - Object-Oriented Programming and Human-Computer Interaction
- COMP 2002 - Data Structures and Algorithms
- COMP 2003 - Computer Architecture
- COMP 2004 - Introduction to Operating Systems
- COMP 2005 - Software Engineering
- COMP 2006 - Computer Networking
- COMP 2007 - Introduction to Information Management
- COMP 2008 - Social Issues and Professional Practice
- COMP 2100 - Social Web Analysis
- COMP 2300 - Introduction to Multimedia Programming
- COMP 2500 - Data Analysis with Scripting Languages
- COMP 2510 - Programming in C/C++
- COMP 2718 - Development Tools, Work Flows and Concepts
COMP 2000
Collaborative and Emergent Behaviour

Students Interested
A grasp of computation as a significant paradigm for understanding both technology and modern models of natural phenomena, and its connection to other fields of human inquiry, is valuable for students generally. Existing courses related to computation are not generally accessible to majors in non-mathematical disciplines, and require extensive background in programming. This course is intended to be generally accessible to undergraduate students, and will be of particular interest to students in Communications Studies.

Objectives of the Course
Collaborative and Emergent Behaviour is a survey of computation as a means of understanding, modeling, and describing artificial and natural systems. The emergence of complex behaviour from the interaction of simple rules governing individual components is illustrated and discussed, as well as the role of communication between system components. Selected systems to be studied will be drawn from different topic areas which may include the worldwide web, the mind (cognitive science), formal logic, autonomous robotics, chaos and fractals, and bioinformatics. Each topic will incorporate an associated laboratory experience.

Prerequisite(s)
None

Successor(s)
None

Representative Workload
Topic Tests (5) 25%
Lab Reports (6) 25%
Final Exam 50%

Representative Course Outline
• Introduction
  - Complex systems: Examples of complex systems, concepts of causality, predictability, and determinism in classical natural science, introduction to chaos, randomness, and emergence as properties of dynamical systems.
  - Communication and logic: “Real-world” reasoning versus formal logic, semantic limitations, concept of “truth” and “knowable,” communication, representation, and encoding of information.

• Selected Topic Area: Chaos and fractals
  - Examples of self-similarity in nature and mathematics, dimensionality in fractal objects, feedback and stability of dynamical systems.
  - Lab work: Guided exploration of rules for generation of fractal objects and landscapes.
• Selected Topic Area: Internet and Mobile Computing
  - Models of networks (client-server, peer-to-peer, etc.), web technologies and applications, protocols, layers, switching.

  - Lab Work: Use mobile devices to explore underlying functionality of network (connectivity, authentication, security).

• Selected Topic Area: Bioinformatics
  - Function and encoding (DNA/RNA/transfer RNA), pattern matching, sequence alignment, gene regulation and metabolism.

  - Lab Work: Sequence reconstruction and pattern matching (using GENBANK).

• Selected Topic Area: Robotics
  - Kinematics, perception, and control, swarm intelligence, communication and emergence.

  - Lab Work 1: Experiment with control parameters for a two-wheeled robot.

  - Lab Work 2: Experiment with rules for robot co-operation and group behaviour.

• Selected Topic Area: Cognitive Science
  - Metaphors for mind, mind as computation, limits on computational minds, human problem solving, artificial intelligence.

  - Lab Work: Beat the computer at problem solving; can you pass a reverse Turing test?

Comments or Notes
• Students will be expected to attend six bi-weekly three-hour lab sessions and to submit a lab report at the end of each lab.
COMP 2001
Object-Oriented Programming and Human-Computer Interaction

Students Interested
This course is required for all computer science majors and minors.

Objectives of the Course
Advancing from Introduction to Programming, this course studies object-oriented programming. Additional topics include event-driven programming, program correctness and simple refactoring, as well as interfaces and human-computer interaction. A brief overview of programming languages is also provided.

Prerequisite(s)
COMP 1001, COMP 1003 and Mathematics 1000

Successor(s)

Representative Workload
Assignments 20%
Lab Quizzes 15%
Midterm Exam 20%
Final Exam 45%

Representative Course Outline
- Object-oriented programming (10 hours)
- Event-driven programming (2 hours)
- An overview of programming languages (3 hours)
- Program correctness and simple testing and refactoring (4 hours)
- User interfaces, and human-computer interaction (8 hours)

Comments or Notes
- In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
- Credit cannot be obtained for both Computer Science 2001 and the former Computer Science 2710
COMP 2002
Data Structures and Algorithms

Students Interested
This course is required for all computer science majors.

Objectives of the Course
The objective of this course is to teach problem solving techniques via fundamental algorithms and data structures, basic design techniques and analysis. The course will be taught primarily in a language-neutral manner, with material presented mainly in pseudocode; however, it will incorporate a significant programming (implementation) component through its assignments and labs.

Prerequisite(s)
COMP 1001, COMP 1002 and COMP 1003

Successor(s)

Representative Workload
Assignments 30%
Lab Quizzes 15%
Test(s) 20%
Final Exam 35%

Representative Course Outline
- Algorithm analysis (5 hours)
- Algorithm design techniques: brute-force, greedy, divide-and-conquer, backtracking, dynamic programming (5 hours)
- Data structures (stacks, queues, binary search trees, hash tables, graphs) (6 hours)
- Fundamental algorithms: sorting, searching, BFS/DFS, MST, shortest path (12 hours)

Comments or Notes
- In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
- Credit cannot be obtained for both Computer Science 2002 and the former Computer Science 2711.
COMP 2003
Computer Architecture

Students Interested
This course is required for all computer science majors. Since this course addresses the how and why of computer organization, anyone who is interested in an understanding of how computers are constructed should take this course.

Objectives of the Course
The objective of this course is to explore the architecture of computers and how computers are constructed. The course will examine the classical components of a computer system, such as control, data path, memory, input and output. The course will also examine the common levels of abstraction used to reason about computer organization and architecture. These include: instruction set architectures, hardware components, register transfer level and logic design.

Prerequisite(s)
COMP 1001, COMP 1002 and COMP 1003

Successor(s)
COMP 2004*, COMP 4711*, COMP 4712*, COMP 4723, COMP 4770*

Representative Workload
Assignments  30%
In-class Exam   20%
Final Exam     50%

Representative Course Outline
• Computer components (CPU, ALU, buses, memory, I/O devices) (5 hours)
• Integer arithmetic, bitwise operations (3 hours)
• State machines (4 hours)
• Instruction set architecture (5 hours)
• Boolean algebra and logic design (2 hours)
• Register transfer level (3 hours)
• Memory management (3 hours)
• Interruption and IO (1 hour)
• Multiprocessing and alternative architectures (2 hours)

Comments or Notes
• In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
• Credit cannot be obtained for both Computer Science 2003 and the former Computer Science 3724.
COMP 2004
Introduction to Operating Systems

Students Interested
This course is the first course in operating systems for all computer science majors.

Objectives of the Course
The main objectives of the course are to learn how an operating system interfaces the hardware and software resources with the user’s environment, to learn the compositions and connections of multilevel operating systems and to design substantial parts of an operating system.

Prerequisite(s)
COMP 2002 and COMP 2003

Successor(s)
COMP 2006*, COMP 2007*, COMP 4721, COMP 4754*, COMP 4759*

Representative Workload
Assignments 30%
In-class Exam 20%
Final Exam 50%

Representative Course Outline
- Introduction, overview, and history (2 hours)
- Process management (5 hours)
- Process coordination (4 hours)
- Memory hierarchy and management (4 hours)
- File management (2 hours)
- Interface communication (3 hours)
- Protection (2 hours)
- Performance evaluation (3 hours)

Comments or Notes
- Credit cannot be obtained for both Computer Science 2004 and the former Computer Science 3725.
COMP 2005
Software Engineering

Students Interested
This course is required for all computer science majors and is aimed at helping students build an understanding of how to develop a software system from scratch by guiding them through the development process and giving them the fundamental principles of systems development with object oriented technology.

Objectives of the Course
The objective of this course is to create a software system from requirements capture, perform requirements capture with use case analysis, create an object-oriented domain analysis model from the use cases, transform the domain model into software classes and apply design patterns in selecting and creating the software.

Prerequisite(s)
COMP 2001

Successor(s)
COMP 4718, COMP 4768*, COMP 4770*

Representative Workload
<table>
<thead>
<tr>
<th>Workload</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project 1</td>
<td>15%</td>
</tr>
<tr>
<td>Project 2</td>
<td>15%</td>
</tr>
<tr>
<td>Project 3</td>
<td>20%</td>
</tr>
<tr>
<td>Midterm 1</td>
<td>25%</td>
</tr>
<tr>
<td>Midterm 2</td>
<td>25%</td>
</tr>
</tbody>
</table>

Representative Course Outline
- Software development process definitions (3 hours): Software lifecycles, activities within software lifecycles, systems level considerations, i.e., the interaction of software with its intended environment, introduction and evaluation of software process models, programming in the large vs. individual programming.
- Use cases and UML use case notation (2 hours)
- Applying use case for requirement capture (2 hours)
- Design patterns (8 hours)
- Software construction (2 hours)
- Software project management (2 hours)
- Software verification and validation (4 hours)
- Software evolution (1 hour)
- Software reliability (1 hour)
- Professional communication (1 hour)
- Professional ethics (1 hour)
- Defensive programming (1 hour)
- Optional: Software tools, cvs, ant, junit

Comments or Notes
- Credit cannot be obtained for both Computer Science 2005 and the former Computer Science 3716.
Students Interested
This course is required for all computer science majors.

Objectives of the Course
The objective of this course is to study design of Internet protocols and their performance. It will focus on the most commonly used of those protocols that are in the vast majority of modern computer systems.

Prerequisite(s)
COMP 2001 and COMP 2002

Co-requisite(s)

Successor(s)
COMP 4759*, COMP 4768*, COMP 4770*

Representative Workload
Assignments 30%
Project 20%
Final Exam 50%

Representative Course Outline
• Introduction and protocol stack layering (1 hour)
• Essential application-layer protocols (1 hour)
• Programming with socket (1 hour)
• Transport layer, reliable data transfer, and TCP/UDP (2 hours)
• Network layer, switching, routing, and IP (2 hours)
• Link and access technologies (2 hours)

Comments or Notes
• Credit cannot be obtained for both Computer Science 2006 and the former Computer Science 3715.
• Computer Science 2006 is a one credit-hour course.
COMP 2007
Introduction to Information Management

Students Interested
This course is required for all computer science majors.

Objectives of the Course
The objective of this course is to present basic concepts in database systems and information management.

Prerequisite(s)
COMP 2002

Co-requisite(s)

Successor(s)
COMP 3200*, COMP 3201*, COMP 3401*, COMP 4754*, COMP 4770*

Representative Workload
Assignments 20%
Projects (2) 50%
Final Exam 30%

Representative Course Outline
• Basic information storage and retrieval, capture and representation (1 hour)
• Queries and links, analysis, indexing, quality issues such as scalability (2 hours)
• Database systems, core DBMS function design, architecture, use of a declarative query language (3 hours)
• Data modeling (2 hours)
• Security and privacy (1 hour)

Comments or Notes
• Credit cannot be obtained for both Computer Science 2007 and the former Computer Science 3754.
• Computer Science 2007 is a one credit-hour course.
COMP 2008
Social Issues and Professional Practice

Students Interested
This course is a required course for all computer science majors.

Objectives of the Course
This course covers ethical and social considerations of computing to provide students with a basis to address these issues by appropriate actions. Case studies are used to illustrate these and other professional challenges of computing.

Prerequisite(s)
COMP 1003

Co-requisite(s)
COMP 2006, COMP 2007

Successor(s)
COMP 4767*, COMP 4768*

Representative Workload
<table>
<thead>
<tr>
<th>Component</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-class participation</td>
<td>10%</td>
</tr>
<tr>
<td>Assignments</td>
<td>30%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>40%</td>
</tr>
</tbody>
</table>

Representative Course Outline
- Social implications of computing (both positive and negative) (1 hour)
- Fundamentals of ethical analysis (3 hours)
  - Ethical argumentation and theories
  - Analysis of case studies
- Professional ethics (2 hours)
  - Professional codes of ethics
  - Ethical responsibilities in software development
- Intellectual property (IP rights, copyright, plagiarism, software piracy, open source) (1 hour)
- Privacy and civil liberties (1 hour)
- Sustainability (1 hour)

Comments or Notes
- Credit cannot be obtained for both Computer Science 2008 and the former Computer Science 2760.
- Computer Science 2008 is a one credit-hour course.
COMP 2100
Social Web Analysis

Students Interested
This course would be of interest to students completing the Visual Computing and Games stream.

Objectives of the Course
This course covers the analysis of social network structures, the flow of data within them and the methods to extract useful information about these networks, their participants and the content of their communication. Security and trust issues are also covered.

Prerequisite(s)
COMP 1003

Successor(s)
None

Representative Workload
Assignments (6) 30%
Project 10%
Midterm Exam 20%
Final Exam 40%

Representative Course Outline
• Network Structures and Measures (6 hours)
• Network Visualization (4 hours)
• Understanding Structure through User Attributes and Behaviour (4 hours)
• Propagation in Networks (5 hours)
• Location Based Social Interaction (4 hours)
• Security and Privacy (2 hours)
• Trust, Social Issues and Other Topics (2 hours)
COMP 2300
Introduction to Multimedia Programming

Students Interested
This course would be of interest to students completing the Visual Computing and Games stream.

Objectives of the Course
This course contains an introduction to programming and computer science with an emphasis on the development of multimedia applications. The course introduces the fundamental principles of programming, including object-oriented and event-driven programming. Students will develop an understanding of how to use and create classes and methods and combine them with multimedia libraries to produce animations, handle input from keyboard and mouse, and import sounds and videos to produce multimedia applications which can be directly deployed on the Internet.

Prerequisite(s)
COMP 1003

Successor(s)
None

Representative Workload
In-class Evaluations 15%
Laboratory Work 25%
Midterm Exams (2) 30%
Final Exam 30%

Representative Course Outline
- Introduction: definitions of multimedia, digital data fundamentals, multimedia authoring overview (2.5 hours)
- Flash and ActionScript: overview of Flash, ActionScript syntax overview (2.5 hours)
- Using objects: getting familiar with variables, using objects, calling methods, using properties and events (2.5 hours)
- Data types: primitive data types, expressions, strings, string manipulation (2.5 hours)
- Conditionals: if statement, boolean expressions, switch-case statements, nesting conditionals (2.5 hours)
- Loops: while, do and for loops (2.5 hours)
- Arrays and Functions: using arrays, defining functions (2.5 hours)
- Object References: understanding objects vs. object references, calling objects from the library, defining the document class (2.5 hours)
- Implementing Object Classes: implementing methods, understanding constructors, instance fields, and local variables (2.5 hours)
- Events and Event Handling: ActionScript event basics, assessing objects through event handlers, event listeners (3 hours)
- Style and HCI Guidelines for Multimedia: fonts, color, user feedback, splash screens, loaders and publishing (2.5 hours)

Comments or Notes
- In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
- Credit cannot be obtained for both Computer Science 2300 and the former Computer Science 1550.
Students Interested
Data analysis is an essential part of many university courses and research projects. Computer systems have become indispensable for many data analysis tasks. Students and researchers can use scripting languages to perform many data analysis tasks more easily than with traditional programming languages. This course is of interest to people who need to readily perform data analysis. Scripting languages have been adopted by many research communities to aid in their research.

Objectives of the Course
This course will provide practical examples of using a scripting language to perform data analysis and data visualization. Common tasks of storing, searching, curve fitting, statistical analysis, plotting and data computation for data sets will be covered. Possible scripting languages include Perl, Python, and Ruby.

Prerequisite(s)
COMP 1510, COMP 1700, COMP 1710 or COMP 1000 or COMP 1001 (or equivalent)

Successor(s)
COMP 2718, COMP 3550*

Representative Workload
Assignments 30%
Tests 30%
Final Exam 40%

Representative Course Outline
- Control statements and expressions of the scripting language
- Subroutines and modules
- String processing
- Managing data files and sets
- Processing collections of data with lists and dictionaries
- Graphing (plotting) and data visualization
- Statistical data analysis
- Numeric computations with packages
- Finding and using application specific packages (e.g., astronomy, physics, chemistry)
COMP 2510  
Programming in C/C++

Students Interested
C++ is an object-oriented relative of C and one of the most widespread computer languages today. Especially, in the computer and software industry, it is one of the most sought-after skills. Many of the more recent computer languages and advanced data processing techniques require previous exposure to C/C++. This course is open to Computer Science students and students from other disciplines.

Objectives of the Course
This course gives a comprehensive treatment of the C/C++ programming languages. It is intended for students with some first programming experience. This course starts with a discussion of fundamentals of C and C++, moves on to the object-oriented aspects of C++, and introduces some advanced topics. It is an essential course for mastering the power of this rich programming language.

Prerequisite(s)
COMP 1001 or COMP 1510 or the former 1710 or Engineering 1020 (or equivalent)

Successor(s)
COMP 2718

Representative Workload
Assignments (5-8) 20%
Labs (10) 10%
Midterm Exam 20%
Final Exam 50%

Representative Course Outline
• Overview of C (3 weeks)
  - fundamental data types and conversion
  - flow control and functions
  - pointers, compound types, and structures
  - preprocessing
  - essential libraries
  - memory models, separate compilation, storage duration, scope, linkage, and namespaces

• Objects and classes (3 weeks)
  - class constructors and destructors
  - class scope
  - this pointer
  - function and operator overloading
  - friends
• Inheritance (3 weeks)
  - access control
  - single and multiple inheritance
  - polymorphism
  - class type cast
  - static and dynamic binding
  - abstract based classes

• More topics (3 weeks)
  - runtime type identification
  - the string class
  - exceptions
  - smart pointers
  - class templates
  - Standard Template Library

Comments or Notes:
• In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
• Students can receive credit for only one of Computer Science 2510 or Engineering 3891.
COMP 2718
Development Tools, Work Flows and Concepts

Students Interested
This course will enable computer science majors and minors to improve their software
development productivity at the start of their program. Students who take the Co-operative
Internship Option will be prepared for the tools and work flows in use by industry. Non
computer science majors can also benefit from this course since the majority of these tools and
work flows can also be used by non software developers to improve their data management and
data processing tasks.

Objectives of the Course
This course covers tools, work flows and concepts used in software development in a concentrated
introductory set of topics. The essential work flows (with their underlying concepts) used to edit,
built, test, combine with existing software and find existing software are introduced. The tools
covered include text editors, programming language translators, file management tools, debuggers,
scripting tools, source control tools, and building, testing and deployment tools. The architecture
and use of an Integrated Development Environment are discussed.

Prerequisite(s)
COMP 2500 or COMP 2510 or COMP 2710

Successor(s)
None

Representative Workload
Assignments (5-8) 15%
Labs (8-10) 20%
Midterm Exam 15%
Final Exam 50%

Representative Course Outline
- Files, File Systems, File Types (1 hour)
- File Management Tools (2 hours)
- Text Processing Tools (2 hours)
- Shell Scripting (4 hours)
- Compiling Tools (1 hour)
- Building Tools (2 hours)
- Packaging and Deployment Tools (1 hour)
- Resources on the Internet (2 hours)
- Network Tools (3 hours)
- Installing, Configuring Software Libraries (2 hours)
- Source Control Tools (2 hours)
- Debugging Tools (3 hours)
- Testing Frameworks (2 hours)
• Lint like Tools (2 hours)
• Work Flow Management Tools (2 hours)
• Integrated Development Environments (3 hours)
• Software Licenses and Intellectual Property (1 hour)

Comments or Notes
• In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
• It is recommended that Computer Science students take Computer Science 2718.
3.3 **3000-Level Courses**

COMP 3100 - Web Programming
COMP 3200 - Algorithmic Techniques for Artificial Intelligence
COMP 3201 - Introduction to Nature-Inspired Computing
COMP 3202 - Introduction to Machine Learning
COMP 3300 - Interactive Technologies
COMP 3301 - Visual Computing and Applications
COMP 3401 - Introduction to Data Mining
COMP 3550 - Introduction to Bioinformatics
COMP 3600 - Algorithm Design and Analysis
COMP 3602 - Introduction to the Theory of Computation
COMP 3700 - Industrial Experience
COMP 3710 - Vocational Languages
COMP 3718 - Programming in the Small
COMP 3731 - Introduction to Scientific Computing
COMP 3753 - Computational Aspects of Linear Programming
COMP 3100
Web Programming

Students Interested
This course would be of interest to students who want to design web applications.

Objectives of the Course
This course studies the web information system from a programming perspective. It teaches how web data are transferred across the network, how to design interactive browser content and how to provide dynamic pages from the server.

Prerequisite(s)
COMP 2006, COMP 2007

Successor(s)
None

Representative Workload
Assignments (5) 30%
Midterm Exam 20%
Project 20%
Final Exam 30%

Representative Course Outline
- Web information systems foundation (3 hours)
  - HTTP
  - Web servers
  - Browsers and other clients
- Client side (12 hours)
  - HTML5
  - CSS3
  - JavaScript
  - Browser API
  - Browser Development Tools
  - Mobile web applications
- Server side (11 hours)
  - CGI
  - Python Server Frameworks (e.g. flask, ...)
- Application frameworks (4 hours)

Comments or Notes
- Credit cannot be obtained for Computer Science 3100 and the former Computer Science 3715.
COMP 3200
Algorithmic Techniques for Artificial Intelligence

Students Interested
This course is required for the Smart Systems stream and would be of interest to Visual Computing and Gaming students.

Objectives of the Course
This course covers basic algorithmic techniques and data structures that are used to embed basic intelligent behaviors, such as problem solving, reasoning and learning in software systems and agents.

Prerequisite(s)
COMP 2001 or the former COMP 2710, COMP 2002 or the former COMP 2711, and Statistics 2550 or any of the courses listed in the credit restrictions of Statistics 2550

Successor(s)
COMP 3202, COMP 4303

Representative Workload
Assignments (5) 45%
Tests (2) 30%
Final Exam 25%

Representative Course Outline
- Background: AI and Agents (3 hours)
  - AI definition and areas
  - Agent definition, structure and types
- Search (10 hours)
  - Exhaustive search
  - Heuristic search
  - Local search (e.g., hill-climbing)
  - Constraint satisfaction
  - Adversarial search
  - Search under uncertainty
- Logical Reasoning (9 hours)
  - Knowledge-based systems
  - Reasoning
  - Planning
  - Fuzzy logic
- Probabilistic Reasoning (8 hours)
  - Quantifying uncertainty
  - Bayesian networks
- Dynamic Bayesian networks

Comments or Notes
- Credit cannot be obtained for both Computer Science 3200 and the former Computer Science 4753.
COMP 3201
Introduction to Nature-Inspired Computing

Students Interested
This course is required for the Smart Systems stream.

Objectives of the Course
This course provides an overview of popular nature-inspired computing methods. Methods that are inspired by both biological and non-biological systems are considered. These methods have been applied to solve problems in various areas of computing such as optimization, machine learning, and robotics. Particular examples of nature-inspired computing methods studied include cellular automata, neural networks, evolutionary computing, swarm intelligence, artificial life, and complex networks. Contributions made in the field of nature-inspired computing that have led to advances in the natural sciences are also discussed.

Prerequisite(s)
COMP 2002 or the former COMP 2711

Successor(s)
None

Representative Workload
Assignments (5) 30%
Midterm Exam 25%
In-class Participation 10%
Final Exam 35%

Representative Course Outline

• Introduction to nature-inspired computing (1 hour)
  - History
  - Major tasks
  - Natural paradigms

• Cellular automata (1 hour)
  - Dynamical systems simulation
  - Self-replication

• Neural Networks (8 hours)
  - Background and history of artificial neural networks (ANNs)
  - Learning algorithms based on ANNs
  - Optimization with ANNs
  - Selected applications of ANNs

• Evolutionary Computing (8 hours)
  - Background and history of evolutionary computation (EC)
- Different branches of EC: GA, GP, EA, EP, DE
- Selected applications of EC methods

- Swarm Intelligence (8 hours)
  - Background and history of collective and swarm intelligence
  - Examples of swarm intelligence in biology
  - Mechanisms of swarm behaviour (e.g., recruitment, quorum sensing)
  - Selected application of swarm methods

- Artificial Life (2 hours)
  - Background and history of Artificial Life research
  - Self-organizing systems
  - Artificial Chemistry

- Complex networks and emergence (2 hours)
  - Background and history of network science
  - Random networks, small-world networks and networks in nature
  - Artificial networks and their features
  - Selected phenomena in network science

Comments or Notes
- Credit cannot be obtained for both Computer Science 3201 and the former Computer Science 4752.
COMP 3202
Introduction to Machine Learning

Students Interested
This course is required for the Smart Systems stream.

Objectives of the Course
This course introduces concepts and algorithms in machine learning for regression and classification tasks. The course gives the student the basic ideas and intuition behind model selection and evaluation, and selected machine learning methods such as random forests, support vector machines, and hidden Markov models.

Prerequisite(s)
COMP 3200; or COMP 2001 or the former COMP 2710, COMP 2002 or the former COMP 2711, and Statistics 2550 or any of the courses listed in the credit restrictions of Statistics 2550; and Mathematics 2050

Successor(s)
None

Representative Workload
Assignments (5) 30%
Midterm Exam 25%
In-class Participation 10%
Final Exam 35%

Representative Course Outline
• Introduction to Machine Learning (3 hours)
  - Definition and examples of machine learning tasks, e.g., classification
  - Types of learning: supervised, unsupervised and reinforcement

• Linear methods for regression and classification (5 hours)

• Model Assessment and Selection (3 hours)
  - Bias, variance, overfitting, and model complexity

• Measuring classifier performance (3 hours)
  - Cross-validation
  - Precision / Recall
  - Area under ROC curve

• Supervised learning (6 hours)
  - Nearest-neighbour
  - Decision Trees

• Combining classifiers (6 hours)
  - Boosting
  - Random Forests

• Other approaches such as support vector machines, hidden Markov models, etc. (4 hours)
COMP 3300
Interactive Technologies

Students Interested
This course is required for the Visual Computing and Games stream.

Objectives of the Course
This course provides exposure to traditional desktop, mobile and games contexts with respect to interaction design theory and practice. The impact of context on design principles is explored. An introduction to each programming context will be provided and a minimal set of software development tools for each context will be introduced. Practical application of interaction design principles will involve design and prototyping of desktop, mobile and games applications.

Prerequisite(s)
COMP 2001 or the former COMP 2710

Successor(s)
None

Representative Workload
Original Implementation Assignments (3)  60%
Tests and Design Problems (3)  40%

Representative Course Outline
• Introduction to Interaction Design
  - Interaction Design goals, guidelines and principles (3 hours)
  - Cognate areas of impact: cognitive psychology, graphic design, industrial design, ergonomics, human-computer interaction, media studies, user-interface design (3 hours)
  - Design methodology (2 hours)
  - Prototyping tools (1 hour)

• Desktop Context
  - Desktop design constraints and objectives (1 hour)
  - Introduction to a Desktop GUI builder (WindowBuilder/Eclipse) (3 hours)
  - GUI component frameworks and systems (3 hours)

• Mobile Context
  - Mobile device design constraints and objectives (1 hour)
  - Introduction to a Mobile App development suite (Android Studio) (3 hours)
  - Mobile App programming (6 hours)

• Games Context
  - Games design space and objectives (1 hour)
  - Introduction to a game design suite (Unity3D) (3 hours)

• Games programming (6 hours)
COMP 3301
Visual Computing and Applications

Students Interested
This course is required for the Smart Systems Stream, and the Visual Computing and Games stream.

Objectives of the Course
This course provides students with the fundamental knowledge and skills in the fields of computer vision, computer graphics, and visualization. Visual perception is responsible for most of our impressions about the world around us. This course introduces how computers are used to both mimic the human visual system (e.g., recognize shapes) and to create visual content (e.g. synthesize images). Related techniques on image synthesis, processing and analysis are discussed under a unified framework. How visual computing principles were used to create visual effects in movies and commercials is also examined.

Prerequisite(s)
COMP 2002 or the former COMP 2711, Mathematics 2000, and Mathematics 2050

Successor(s)
COMP 4301, COMP 4302

Representative Workload
Assignments (4) 40%
Midterm Exam 20%
Final Exam 40%

Representative Course Outline
• Introduction (4 hours)
  - Human perception; intensity & color; display hardware
• Image basics (4 hours)
  - Imaging capture; image representation & sampling; 2D transformations; image warping
• Raster graphics (4 hours)
  - Line & circle drawing; line clipping; polygon filling
• Image processing (8 hours)
  - Per-pixel operation; histogram; filtering; Fourier transformation
• Image analysis (6 hours)
  - Edge detection; Hough transform; thresholding; segmentation; template matching
• Image synthesis (6 hours)
  - Alpha compositing; matting; halftoning; feature-based warping; morphing
COMP 3401
Introduction to Data Mining

Students Interested
This course would be of interest to students completing the Smart Systems stream.

Objectives of the Course
This course introduces students to the basic concepts and techniques for data mining and knowledge discovery. Students will develop an understanding of the essential data mining technologies, and be able to design and evaluate methods for simple data mining applications.

Prerequisite(s)
COMP 2002 or the former COMP 2711, COMP 2007 or the former COMP 3754, and Statistics 2550 or any of the courses listed in the credit restrictions of Statistics 2550

Successor(s)
None

Representative Workload
Assignments (5) 15%
Midterm Exam 25%
Project 10%
Final Exam 50%

Representative Course Outline
• Basic concepts for data mining (3 hours)
• Attributes (4 hours)
• Data pre-processing (3 hours)
• Mining frequent patterns (4 hours)
• Classification (9 hours)
• Clustering (6 hours)
COMP 3550
Introduction to Bioinformatics

Students Interested
This course is designed as an interdisciplinary introductory course in bioinformatics for both Computer Science and Biology students and as a bridge between both disciplines. The course is intended to be a course for a mixed audience of students with different backgrounds (e.g. computer science and biology). The course will focus on the fundamental concepts, ideas and related biological applications of existing bioinformatics tools. The purpose is to provide the students with hands-on experience on the major computational approaches applied to a wide variety of bioinformatics problems.

Biology students will appreciate the impact of these approaches for addressing biological questions and will gain insight on the limitations and strengths of these approaches. Computer Science students will appreciate the practical use of the concepts they have been taught in other courses, but most importantly, the challenges posed by biological questions, and the need for the robust algorithms that deal with the very large, noisy datasets typically present in biology. Computer scientists and biologists will both recognize the large diversity of questions addressed by bioinformatics applications. Many industry and research jobs now require cross-disciplinary collaboration. With this course, students will start becoming aware of the interdisciplinary nature of bioinformatics and appreciate the contribution of people outside their field of study.

Objectives of the Course
Bioinformatics deals with the development and application of computational methods to address biological problems. The course will focus on the fundamental concepts, ideas and related biological applications of existing bioinformatics tools. This course will provide hands-on experience in applying bioinformatics software tools and online databases to analyze experimental biological data, and it will also introduce scripting language tools typically used to automate some biological data analysis tasks.

Prerequisite(s)
COMP 2500 or COMP 2710 or COMP 2001, and one Biology course at the 1000-level or above (excluding Biology 2040 and Biology 2041); or Biology 2060 or Biochemistry 2201 or the former 2101, and one Computer Science Course at the 1000-level or above (excluding COMP 1400, COMP 1600 and COMP 2000); or permission of the course instructor

Successor(s)
COMP 4550*

Representative Workload
Assignments and Projects 25%
Lab work and Quizzes 20%
Midterm Exam 25%
Final Exam 30%
Representative Course Outline

- Introduction
  What is Bioinformatics?
  Why is Bioinformatics required?
  Importance of interdisciplinary collaboration

- Sequences
  Why compare sequences?
  Sequence similarity
  Where to look for information about a sequence
  Sequence alignment: Pairwise and multiple

- Genomics
  How are genomes sequenced?
  How are genomes annotated?
  Genomic variation
  Gene expression

- How is gene expression measured?

- Pre-processing the data: denoising and normalization

- Differential analysis
  Interpreting a list of genes

- Gene functional annotation - Gene Ontology (GO)

- Finding over-represented gene functions in gene lists

- Other source of annotations
  Gene function prediction

- Proteomics
  Protein Interaction Networks
  Protein Domains
  How are proteins measured and identified?

- Transcriptomics
  Motif finding
  Determining binding preferences
  Inferring regulatory networks

- Metabolomics
  Detection and identification of metabolites
  Human metabolome project

- Labs

  Students will be expected to attend a weekly lab session, and to submit a lab report or to answer a lab quiz at the end of each lab.
* Script programming and using bioinformatics libraries (BioPerl)
* Sequences
  - Using BLAST, BLAT
  - Using alignment tools (e.g., ProbCons, M-Coffee)
* Working with sequenced genomes
  - Ensembl, BioMart, UCSC Genome Browser
  - Linking own data to a Genome browser
* Analysis of gene expression data using existing tools (e.g., Babelomics, GeneXPress, Gene Pattern)
* Annotating a list of genes with functional annotation
* Using over-representation or enrichment analysis tools (e.g., GSEA, DAVID, GenMAPP, GOMiner)
* Using gene function prediction systems (e.g., GeneMANIA, FuncBase, NBrowse, STRING, FunCoup)
* Using motif finding tools in a set of sequences (e.g. MEME, AlignACE)
* Using regulatory networks prediction systems (e.g. COALESCE, Allegro)

**Comments or Notes**
- In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
- Students can receive credit for only one of Computer Science 3550 or Biology 3951.
COMP 3600
Algorithmic Design and Analysis

Students Interested
This course would be of interest to all computer science majors.

Objectives of the Course
This course is intended to refine the student's ability to solve computational problems algorithmically; it goes beyond specific algorithms to cover general methods of algorithm design and analysis and algorithmic problem solving.

Prerequisite(s)
COMP 2002 or the former COMP 2711

Successor(s)
COMP 4743, COMP 4750

Representative Workload
Assignments (6) 40%
Midterm Exam 20%
Final Exam 40%

Representative Course Outline
- Greedy algorithms (4 hours)
- Divide and conquer (6 hours)
- Dynamic programming (5 hours)
- Network flows (4 hours)
- Notion of NP-completeness, examples of reductions (6 hours)
- Dealing with intractability: approximation algorithms, heuristics, randomized algorithms (5 hours)
COMP 3602
Introduction to the Theory of Computation

Students Interested
This course would be of interest to all computer science majors.

Objectives of the Course
This course addresses complexity and limitations of algorithmic problem solving relative to a variety of computational resources and machine models. It provides a formal introduction to theory of computation, presenting abstract machine models, the basics of computability theory and complexity theory.

Prerequisite(s)
COMP 2002 or the former COMP 2711

Successor(s)
COMP 4711, COMP 4741, COMP 4742

Representative Workload
Assignments (6) 40%
Midterm Exam 20%
Final Exam 40%

Representative Course Outline
- Regular languages and finite automata (6 hours)
- Context-free languages and pushdown automata (5 hours)
- Turing machines, undecidability, many-one reductions (6 hours)
- Complexity theory, polynomial-time reductions, P vs. NP (6 hours)
- Other topics, including recursion theorem, space complexity, circuit complexity, parameterized complexity, basic cryptography (8 hours)
COMP 3700  
Industrial Experience

Students Interested
Students who are admitted to the Co-operative Internship in Computer Science (CICS) are required to register for this non-credit course every semester during their internship.

Objectives of the Course
To provide an opportunity for qualified students to obtain rewarding placements that help them develop practical skills in a real work setting before graduation. The CICS is available to computer science majors who will typically apply between their third and fourth year of studies.

Prerequisite(s)
Admission to the Co-operative Internship in Computer Science

Successor(s)
None

Representative Workload
Co-operative Internship students are required to submit assignments each semester as outlined in the course syllabus. The objectives of work term assignments are to:

- Help students learn how to set goals and measure their own progress
- Help students understand and practice reflection
- Help students learn how to articulate their skills and core competencies
- Connect course content to workplace activities
- Conduct career research and prepare for future work.

Representative Course Outline
Not applicable

Comments or Notes
Students must register for the course Computer Science 3700 every semester during their internship. Computer Science 3700 is a non-credit course open only to students who have been accepted into the internship program.

During the internship the employer and intern will complete student performance evaluations every four months and will submit them to Co-operative Education. The final assessment of total work performed is the responsibility of Co-operative Education, and will be based upon both input from the employer and the intern’s assignments.
The Internship evaluation shall consist of two components:

1. On-the-job Student Performance: Job performance shall be assessed by Co-operative Education in consultation with the Department using information gathered during the internship and input from the employer. Evaluation of the on-the-job student performance will result in one of the following classifications: PASS WITH DISTINCTION, PASS, and FAIL.

2. The Internship Report(s): Evaluation of the internship report will result in one of the following classifications: PASS WITH DISTINCTION, PASS, and FAIL.

The evaluation of the on-the-job student performance and the internship report are recorded separately on the transcript.

Overall evaluation of the internship will result in one of the following final grades being awarded:

PASS WITH DISTINCTION: indicates outstanding performance in both the internship report(s) and the on-the-job student performance. PASS WITH DISTINCTION has been awarded to each of the internship report and the on-the-job student performance.

PASS: indicates that performance meets expectations in both the internship report(s) and on-the-job student performance. The student meets the requirements of a passing mark in the internship report and on-the-job student performance.

FAIL: indicates failing performance in either the internship report(s) or on-the-job student performance or both.

• The following will be noted in the transcript of the intern:
  - Requirements for the Co-operative Internship in Computer Science have been completed.
    Internship Duration: - months.
  - A grade of NC (No Credit) for Computer Science 3700 will be awarded in all semesters of the internship option prior to the final semester.

• In case a student is enrolled in both the Honours program and the CICS, the requirements of both must be met. Upon approval from the honours project supervisor, within the Department, the employer and the Head of the Department of Computer Science, an internship project may be submitted as a component of an honours project. These arrangements must be made within the first semester of the internship placement.

• For more information, see Section 2.4.7 of this handbook
COMP 3710
Vocational Languages

Students Interested
Students and data processing professionals, with a good knowledge in at least one modern high-level programming language, who intend to learn other languages currently in vogue.

Objectives of the Course
This course is intended to provide students with a working knowledge of a variety of high-level programming languages. When given a wide range of languages with which to solve a particular problem, the student will be able to choose the most appropriate language for implementing the solution. Emphasis will be on the general semantic characteristics and the underlying decisions implicit in the design and implementation of these languages.

Prerequisite(s)
COMP 2002 or the former COMP 2711

Successor(s)
None

Representative Workload
Assignments 20%
In-class Exams 30%
Final Exam 50%

Representative Course Outline
• Overview of a selection of languages (normally three)
COMP 3718
Programming in the Small

**Students Interested**
This course is for students interested in the study of tools and techniques used in the construction of small software systems.

**Objectives of the Course**
The goal of this course is to demonstrate the tools and techniques used in the construction of small software systems. In software engineering, software development is characterized by programming in the large or programming in the small. Programming in the large deals with requirement analysis, system architecture design, module specification, module development, testing and maintenance - in other words, the entire software life cycle. Programming in the small deals with how a person creates the set of software components necessary to implement a module. Thus programming in the small is part of the larger software development process.

**Prerequisite(s)**
COMP 2002 or the former COMP 2711

**Successor(s)**
None

**Representative Workload**

<table>
<thead>
<tr>
<th>Task</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td>20%</td>
</tr>
<tr>
<td>Software Project</td>
<td>25%</td>
</tr>
<tr>
<td>In-class Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>35%</td>
</tr>
</tbody>
</table>

**Representative Course Outline**

- Review/Introduction of the Java programming language
  - data and control structures
  - abstraction features and styles
  - pointers and memory management
  - common mistakes
  - classes

- An example of the complete system
  - the requirements
  - design decomposition techniques and strategies
  - the interface specification of submodules
  - the use of software libraries
  - the actual source code
  - the supporting tool and files
  - testing

- Programming in the small
  - Parnas' information hiding
- examples of decomposition according to Parnas
- ADT and modules
- testing and automatic testing

• Software development tools
  - source control systems: svn, git, hg, etc.
  - Integrated Development Environments: eclipse, netbeans, etc.
  - testing frameworks: junit

• The where, when and why of software libraries

• The design of utility software components
  - abstract data types
  - generic programming
  - libraries

• Strategies to the reading and modification of large programs

• Performance issues (time and memory)
COMP 3731
Introduction to Scientific Computing

Students Interested
This course is of interest to students who wish to perform numerical computations in such areas as engineering, physics, statistics, modelling, graphics or operations research, and also to those students who wish to study the effectiveness of these solutions in a given computing environment.

Objectives of the Course
The development of algorithms for the numerical solution of mathematical problems, and the study of the numerical stability of these algorithms are the main objectives of this course. The efficiency of these algorithms with respect to speed and storage requirements is considered as well. Emphasis is also placed on the study of the sensitivity of selected problems to perturbations in the data. There is also a brief introduction to the development of numerical algorithms that take advantage of advanced computer architectures, such as pipeline processors, array processors and parallel processors.

Prerequisite(s)
COMP 1001 or the former COMP 2710, Mathematics 2000 and Mathematics 2050

Successor(s)
COMP 4734, Special Topics in Numerical Computations

Representative Workload
Assignments 50%
In-class Exams 20%
Final Exam 30%

There will be about nine assignments given throughout the semester. Programming assignments are chosen to illustrate topics discussed in the lecture material. They emphasize the numerical dangers that may appear because of the finite precision of computers. Assignments can be written in any programming language unless specified otherwise. Nonprogramming problems are also assigned.

Representative Course Outline
• Errors in computations, computer arithmetic, stability of algorithms and conditioning of problems
• Principles of pipelining and parallel processing
• Evaluation of functions, computation of infinite alternating series
• Nonlinear equations
• Methods for systems of linear equations
• Norms, condition numbers
• Polynomial interpolation, curve fitting, least squares problem, cubic spline
• Numerical integration
Comments or Notes
• Credit cannot be obtained for both COMP 3731 and Mathematics 3132.
COMP 3753
Computational Aspects of Linear Programming

Students Interested
Those who wish to have an introduction to linear optimization problems that arise in many areas such as operations research.

Objectives of the Course
To analyze the Linear Programming (linear optimization) Problem, to investigate the recent developments in the theory necessary to solve this problem, to design efficient algorithms for its solution and to analyze the complexity of these algorithms and their numerical efficiency.

Prerequisite(s)
COMP 1001 or the former COMP 2710 and Mathematics 2050

Successor(s)
Special Topics in Numerical Computations

Representative Workload
Assignments 50%
In-class Exams 20%
Final Exam 30%

Programming assignments emphasize the numerical dangers that may appear because of the finite precision of computers. Some of the programming will use MATLAB.

Representative Course Outline
• Brief review of necessary linear algebra
• Introduction to the Linear Programming Problem (LPP)
• The simplex algorithm
• Sparse matrix techniques for the LPP problem
• Duality and postoptimality analysis
• Extensions to the simplex algorithm
• A brief introduction to interior algorithms for the LPP
3.4 4000-Level Courses

COMP 4300 - Introduction to Game Programming
COMP 4301 - Computer Vision
COMP 4302 - 3D Computer Graphics
COMP 4303 - Artificial Intelligence in Computer Games
COMP 4304 - Data Visualization
COMP 4550 - Bioinformatics: Biological Data Analysis
COMP 4711 - Structure of Programming Languages
COMP 4712 - Compiler Construction
COMP 4715 and COMP 4717 - Special Topics in Programming Languages
COMP 4718 - Survey of Software Engineering
COMP 4721 - Operating Systems
COMP 4723 - Introduction to Microprocessors
COMP 4726 to 4729 - Special Topics in Computer Systems
COMP 4734 - Matrix Computations and Applications
COMP 4736 to 4739 - Special Topics in Numerical Computations
COMP 4741 - Formal Languages and Computability
COMP 4742 - Computational Complexity
COMP 4743 - Graph Algorithms and Combinatorial Optimization
COMP 4745 to 4749 (excluding 4748) - Special Topics in Theoretical Aspects
COMP 4750 - Introduction to Natural Language Processing
COMP 4754 - Database Systems
COMP 4759 - Computer Networks
COMP 4766 - Introduction to Autonomous Robotics
COMP 4768 - Software Development for Mobile Devices
COMP 4770 - Team Project
COMP 4780 - Honours Project
COMP 4800 to 4825 - Special Topics
COMP 4300
Introduction to Game Programming

Students Interested
This course is required for the Visual Computing and Games stream.

Objectives of the Course
This is an introductory course for students interested in learning the fundamentals of game programming. Topics include vector math for games, fundamentals of rendering, introduction to animation and artificial intelligence, collision detection, game physics and user-interfaces. Students are required to write a fully functional game during the course.

Prerequisite(s)
COMP 2001 or the former COMP 2710, Mathematics 2050, and 6 credit hours in Computer Science courses at the 3000-level or above (COMP 3301 and COMP 3731 are recommended)

Successor(s)
None

Representative Workload
<table>
<thead>
<tr>
<th>Workload</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laboratory Work (10)</td>
<td>15%</td>
</tr>
<tr>
<td>In-class Evaluations (every lecture)</td>
<td>15%</td>
</tr>
<tr>
<td>Midterm Exams (2)</td>
<td>30%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Final Game Project</td>
<td>20%</td>
</tr>
</tbody>
</table>

Representative Course Outline
- Course Introduction, Game Programming Basics (3 hours)
- Vector Math and Games (3 hours)
- Rendering (3 hours)
- Quaternions and Input/Sound (3 hours)
- Artificial Intelligence (3 hours)
- Collision Detection and Game Physics (3 hours)
- Camera Systems (3 hours)
- User Interface Systems (3 hours)
- Event-Based Systems and Scripting Languages (3 hours)
- Animation and Assorted Gameplay (3 hours)
- Basic Networking (3 hours)
COMP 4301
Computer Vision

Students Interested
This course would be of interest to students completing the Smart Systems stream or the Visual Computing and Games stream.

Objectives of the Course
This course studies how to develop methods that enable a machine to “understand” or analyze images. The course introduces the fundamental problems in computer vision and the state-of-the-art approaches that address them. Topics include feature detection and matching, geometric and multi-view vision, structure from X, segmentation, object tracking and visual recognition.

Prerequisite(s)
COMP 3301 or Engineering 7854 or permission of the instructor

Successor(s)
None

Representative Workload
<table>
<thead>
<tr>
<th>Assignment(s) (2)</th>
<th>20%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm Exam</td>
<td>20%</td>
</tr>
<tr>
<td>Project</td>
<td>30%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>30%</td>
</tr>
</tbody>
</table>

Representative Course Outline
- Unit 1: Grouping and fitting (4 hours)
  - K-means, Hough transform, RANSAC
- Unit 2: Feature detection and matching (4 hours)
  - Interest point detection (corners/blobs), SIFT, HOG
- Unit 3: Geometric and multi-view vision (4 hours)
  - Geometric transformation, Camera model and camera calibration, Image stitching
- Unit 4: Feature based alignment (4 hours)
  - 2D and 3D feature based alignment, Pose estimation
- Unit 5: Structure from X (4 hours)
  - Epipolar geometry, Stereo vision, Essential and fundamental matrix, Structure from motion
- Unit 6: Segmentation and tracking (4 hours)
  - Foreground segmentation in video, Optical flow, Tracking
- Unit 7: Recognition (4 hours)
  - Introduction to recognition, Object detect and recognition (face detection, pedestrian recognition), General category recognition (bags of features)
Comments or Notes

- Credit cannot be obtained for both Computer Science 4301 and Engineering 8814.
Students Interested
This course would be of interest to students completing the Visual Computing and Games stream.

Objectives of the Course
This course introduces students to state-of-the-art concepts and developments in the field of 3D computer graphics. The underlying algorithms, as well as the basic techniques to develop interactive 3D graphics systems including games and simulators, are presented. Topics of the course include 3D geometrical transformations, 3D projections, 3D modeling and rendering, 3D graphics languages and systems. Advanced photorealistic rendering and image-based rendering techniques may also be covered.

Prerequisite(s)
COMP 3301

Successor(s)
None

Representative Workload
Midterm Exam 25%
Assignments (3-4) 45%
Final Exam 30%

Representative Course Outline
- Introduction to 3D Graphics (1 hour)
- Graphics Pipelines (1 hour)
  - Fixed & programmable pipelines
- Graphic APIs (1-4 hours)
- Processing input for 3D graphics systems (1 hour)
- Geometrical Transformation (4 hours)
  - Math Preliminary, 3D Transformations, 3D Projections, 3D Viewing
- 3D Modeling (3-6 hours)
  - Geometric Primitives, Defining 3D Geometry, Parametric Curve, Parametric Surface,
  - Solid Modeling, Point Set Surface
- Shading (3 hours)
- 3D Rendering (3-6 hours)
  - Visibility Determination, Rasterization and Rendering Buffers, Illumination Model,
  - Polygon Shading, Texture Mapping, Anti-Aliasing
- Photorealistic Rendering (1-4 hours)
  - Global Illumination, Ray Casting, Ray Tracing, Radiosity Rendering, Photon Mapping

Comments and Notes
- Credit cannot be obtained for both Computer Science 4302 and the former Computer Science 4751.
COMP 4303
Artificial Intelligence in Computer Games

Students Interested
This course would be of interest to students completing the Smart Systems stream or the Visual Computing and Games stream.

Objectives of the Course
This course provides an introduction to specific state-of-the-art algorithmic techniques and data structures that are used to efficiently implement humanlike abilities (e.g., awareness, memory, rational decision-making (under uncertainty), movement, co-operation in groups) in computer game agents.

Prerequisite(s)
COMP 3200

Successor(s)
None

Representative Workload
Assignments (4) 40%
Tests (2) 30%
Course Project 30%

Representative Course Outline
- Background: Computer Games (3 hours)
- Basic Decision-Making (12 hours)
  - Scripting; Finite-state Automata; Rule-based systems; Utility systems
- Movement (6 hours)
- Advanced Decision-Making (6 hours)
  - Strategy and tactics; group activities
- Awareness and Memory (6 hours)
- Learning from Experience (3 hours)
COMP 4304
Data Visualization

Students Interested
This course would be of interest to students completing the Visual Computing and Games stream.

Objectives of the Course
This course covers interactive representation of data using a modern programming library. Topics include an introduction to the software platform and the principles for data selection, analysis, design and creation of dynamic visualizations. Students produce interactive web-based objects, addressing problems in the presentation and understanding of large data collections. The techniques discussed are applicable to different sources and types of data.

Prerequisite(s)
COMP 2001 or the former COMP 2710 and COMP 2002 or the former COMP 2711

Successor(s)
None

Representative Workload
Assignments (4) 40%
Project 30%
Term Test(s) (1 or 2) 15%
Final Exam 15%

Representative Course Outline
- Introduction to visual design
  - Visual representations of data (3 hours)
  - Human visual perception (2 hours)
  - Visual encoding and representation (4 hours)
  - Evaluation techniques (2 hours)
- Software techniques and tools
  - Introduction to a data visualization toolkit (D3.js) (6 hours)
  - Interaction techniques and extensions (3 hours)
- Big Data Visualization
  - Types of data / multidimensional data (2 hours)
  - Finding relationships in data (3 hours)
  - Data analysis objectives (1 hour)
  - Data selection and filtering (2 hours)
  - Interaction with data sets (4 hours)

Comments and Notes
- Credit cannot be obtained for both Computer Science 4304 and the former Computer Science 4767.
**COMP 4550**

**Bioinformatics: Biological Data Analysis**

**Students Interested**
The course is designed as an interdisciplinary advanced course for both Computer Science and Biology students in bioinformatics, and as a bridge between both disciplines.

This is an advanced course to provide students with the basis to perform their own analysis of high-throughput data using R and Bioconductor. Students, who succeed in this course, should be comfortable programming in R and be able to use available Bioconductor packages to analyse a variety of biological data such as expression data, high-throughput cell-based assay data and mass spectrometry protein data, and to use a variety of approaches available within the R environment, such as clustering, graphs, classification approaches, such as random forests and support vector machines, and enrichment analysis methods.

In the introductory Bioinformatics course (Computer Science 3550), students should (1) understand the basis of bioinformatics methods, for example, how multiple sequences aligners actually construct the alignments, what steps are involved in the analysis of gene expression, what multiple testing correction is and how it is done; (2) achieve basic Perl programming skills; and (3) use online databases and computational tools. On the other hand, in this advanced course, although some topics such as gene expression, enrichment analysis and proteomics are also covered, the students will be learning how to do the analysis on their own, that is, without relying on the existence of a graphical and friendly computer program that will do the required analysis by choosing the appropriate parameters and clicking on some buttons.

**Objectives of the Course**
This course provides students with the basis to analyse a variety of biological data within an integrated programming environment for data manipulation, calculation and graphical display. Students will learn to extract meaningful information from data generated by high-throughput experimentation. The course will introduce one such integrated programming environment and will explore the computational and statistical foundations of the most commonly used biological data analysis procedures.

**Prerequisite(s)**
Biology 3951 or COMP 3550, and Statistics 2550 or any of the courses listed in the credit restrictions of Statistics 2550, or permission of the course instructor.

**Successor(s)**
None

**Representative Workload**
- Assignments and Projects 25%
- Lab work and Quizzes 20%
- Midterm Exams 30%
- Final Exam 25%

**Representative Course Outline**
- Introduction to R and Bioconductor
• Exploratory data analysis and hypothesis testing
• Gene Expression data analysis
• Mass Spectrometry Protein data analysis
• Clustering and visualization
• Machine learning: concepts and packages
  - Feature selection
  - Cross-validation
  - Multiclass problems
  - Ensemble methods
  - Bayesian methods
• Graphs and Networks
  - Protein interactions
  - Pathways
  - Co-expression graphs
• Biological Annotation
• Gene set enrichment analysis

• Labs

  Students will perform hands on analysis of experimental biological data using mainly R and Bioconductor. Additional software that may be used includes Cytoscape.

  - R programming exercises
  - Exploratory data analysis: graphics/plots generation
  - Processing expression data
  - Processing proteomics data
  - Clustering data and cluster visualization
  - Data classification using supervised machine language
  - Using graphs for data visualization
  - Annotating data
  - Performing enrichment analysis
  - Introduction to Cytoscape

**Comments or Notes**

• In addition to the weekly lecture component, there is a structured laboratory as scheduled in the university timetable.
• Students can receive credit for only one of Computer Science 4550 or Biology 4606.
COMP 4711
Structure of Programming Languages

Students Interested
Those who wish to pursue the study of design aspects of programming languages.

Objectives of the Course
To develop an understanding of the organization of programming languages and the runtime behaviour of programs and to introduce basic concepts of formal description of programming languages.

Prerequisite(s)
COMP 2003 or the former COMP 3724, and COMP 3602 or the former COMP 3719

Successor(s)
Special Topics in Programming Languages

Representative Workload
Assignments 20%
In-class Exams 40%
Final Exam 40%

Representative Course Outline
- Formal specification of syntax
- Data, domains and domain constructors
- Storage and assignments
- Control structures, selection and iteration
- Binding, environment and scope
- Procedures and parameters
- Sequencers and continuations
- Concurrency
- Types and type checking, type constructors
- Introduction to denotational semantics
COMP 4712
Compiler Construction

Students Interested
Those who are interested in theoretical foundations of compiler construction.

Objectives of the Course
To provide a theoretical basis for practical approaches to compiler construction. Primary emphasis is upon popular methods of syntax analysis and code generation. Some background in formal languages and automata theory is required.

Prerequisite(s)
COMP 2003 or the former COMP 3724, and 3 credit hours in Computer Science at the 3000-level or above (COMP 3600 is recommended)

Successor(s)
Special Topics in Programming Languages

Representative Workload
Assignments and Projects 20%
In-class Exams 40%
Final Exam 40%

Representative Course Outline
- Formal languages and their grammars, Chomsky hierarchy, context-free languages, derivation trees, leftmost and rightmost derivations, ambiguity, grammar transformations, normal forms
- Definition of programming languages, BNF and EBNF, syntax and semantics, syntax-driven semantics, lexical, syntax and semantics analysis
- Lexical analysis, finite automata and regular expressions, construction of scanners, lex
- Bottom-up parsing, general and deterministic, LR(k) grammars and parsers, LR, SLR, LALR parsers, code generation for bottom-up parsers, yacc
- Top-down parsing, general and deterministic, LL(k) grammars and parsers, action symbols and code generation for top-down parsing, recursive descent parsing
- Attribute grammars, inherited and synthesized attributes, attribute evaluation schemes
COMP 4715 and COMP 4717
Special Topics in Programming Languages

Students Interested
Those who want a more in-depth study in the area of programming languages.

Objectives of the Course
To give students exposure to current and topical information in the area of programming languages.

Prerequisite(s)
Special topics courses are not offered on a regular basis, but whenever departmental resources permit. For this reason the prerequisites can vary each time the courses are offered.

Successor(s)
None

Representative Workload
Variable

Representative Course Outline
Variable

Comments or Notes
All relevant information on a special topics course is posted on the Computer Science website and the bulletin board outside the Computer Science General Office several weeks before the beginning of the semester in which the course is being offered.
COMP 4718
Survey of Software Engineering

Students Interested
This course is of interest to those students wanting exposure to the design and development of a large-scale software system.

Objectives of the Course
This course introduces methods and tools for developing, managing and maintaining large-scale software systems and provides students with practical experience in the development of a software prototype that goes through the life cycle of large-scale software development.

Prerequisite(s)
COMP 2005 or the former COMP 3716

Successor(s)
None

Representative Workload
Assignments 35%
Project Report 15%
Tests 20%
Final Exam 30%

Representative Course Outline
• Software products and development models
• Project management and cost estimation
• Software requirement engineering (system requirements, software requirements, definition, specification, validation)
• Software prototyping
• Formal specification
• Software design methodologies (object-oriented vs. function-oriented design, user interface design, real-time systems)
• Software validation (program verification and validation, testing, techniques)
• Programming techniques and environments (reliability and reuse CASE)
• Software maintenance, configuration management, software re-engineering

Project Outline
• Use the concepts and techniques of software engineering to develop a software prototype of your own interests (recommended: a potential commercial system).
• Hand in a report of the project. In the report, you need to use your software prototype to show your understanding and practice of software development. Your report must cover the specified stage(s) of the software development life-cycle. As a guideline for you to schedule the project, the report will be broken down into four stages that will be specified respectively in assignments. The report will be marked according to its readability and your discussions on:
- What should be done?
- What methods or techniques are available?
- What did you do and how did you do it?
- What you did not do and why?
- (optional) What feedback did you get from the succeeding steps and what modifications did you make?
COMP 4721
Operating Systems

Students Interested
This course is of interest to those who want to learn about the informal structures of operating systems in batch processing, multi-programming, multi-processing and time-sharing environments.

Objectives of the Course
The main objectives of the course are to learn how an operating system interfaces the hardware and software resources with the user's environment, to learn the compositions and connections of multilevel operating systems and to design substantial parts of an operating system.

Prerequisite(s)
COMP 2004 or the former COMP 3725

Successor(s)
Special Topics in Computer Systems, recommended prerequisite for COMP 4726

Representative Workload
Assignments (up to 6) 25%
In-class Exams 35%
Final Exam 40%

Representative Course Outline
- Introduction and overview
- History of operating systems
- Operating system structures
- Process management: process concept, concurrent processes, CPU scheduling, scheduling algorithms
- Process co-ordination: critical section, process synchronization, semaphores, monitors, critical regions, process communication
- Deadlocks: deadlock prevention, avoidance, detection and recovery
- Memory hierarchy: cache memory, associative memory
- Memory management: swapping, fixed and variable partitions, relocation, paging and segmentation (external and internal fragmentation)
- Virtual memory: page replacement algorithms, thrashing
- Secondary storage management
- File management: file-system organization, file operations, access methods, directory-structure organization
- Protection: access matrix, security, encryption
- Elements of distributed operating systems
- Selected case studies, e.g. UNIX

Comments or Notes
- Students can receive credit for only one of Computer Science 4721 or Engineering 8894.
COMP 4723
Introduction to Microprocessors

Students Interested
This course will be of interest to students who want to gain knowledge of embedded systems with microcontrollers.

Objectives of the Course
This course aims to give students an introduction to the architecture of microcontrollers. Students design and implement simple interface and control functions using C and assembly language. Basic electric circuit theory and electronics for the interfacing of microcontrollers with sensors and actuators will be covered.

Prerequisite(s)
COMP 2003 or the former COMP 3724

Successor(s)
Special Topics in Computer Systems

Representative Workload
Labs 15%
Midterm Exam 15%
Project (including proposal, presentation and documentation) 40%
Final Exam 30%

The project includes an initial presentation of the project proposal, with a brief write-up, a final project presentation and a project write-up.

Representative Course Outline
• Introduction to C
• Basic circuits
• Basic architecture of the ATMEL processors
• Assembly language programming
• On-chip components
• Serial peripherals
• Analog/digital control
• Electronics
• Applications

Comments or Notes
• Enrollment is limited depending on laboratory resources.
• Laboratory: Three hours per week for the first six or seven weeks, after which the laboratory time will be devoted to project development.
COMP 4726-4729
Special Topics in Computer Systems

Students Interested
Those who want a more in-depth study of the area of computer systems.

Objectives of the Course
To give students exposure to current and topical information in the area of computer systems.

Prerequisite(s)
Special topics courses are not offered on a regular basis, but whenever departmental resources permit. For this reason the prerequisites can vary each time the courses are offered.

Successor(s)
None

Representative Workload
Variable

Representative Course Outline
Variable

Comments or Notes
All relevant information on a special topics course is posted on the Computer Science website and the bulletin board outside the Computer Science General Office several weeks before the beginning of the semester in which the course is being offered.
COMP 4734  
Matrix Computations and Applications

**Students Interested**  
Students in any discipline who are dealing with computer solutions of numerical linear algebra problems. Matrix computations may be found useful in disciplines such as engineering, statistics, physics, optimization, operations research, computational chemistry and signal processing.

**Objectives of the Course**  
An introduction to the techniques of numerical linear algebra. Emphasis is placed upon developing the most recent and reliable algorithms. The stability of these algorithms as well as the sensitivity of the problems they solve will also be studied.

**Prerequisite(s)**  
COMP 3731

**Successor(s)**  
Special Topics in Numerical Computations

**Representative Workload**  
Assignments  50%  
Midterm Exam  20%  
Final Exam  30%

There will be about nine assignments given throughout the semester. Programming assignments are chosen to illustrate topics discussed in the lecture material and can be written in any programming language unless otherwise specified. Nonprogramming problems are also assigned.

**Representative Course Outline**  
- An introduction to necessary topics of linear algebra  
- Systems of linear equations; scaling; iterative refinement; estimating the condition number of a matrix  
- Introduction to pipelining and parallel matrix computations  
- The linear least squares problem  
- The symmetric and unsymmetric eigenproblems  
- The singular value problem of a matrix
Students Interested
Those who want a more in-depth study in the area of numerical computations.

Objectives of the Course
To give students exposure to current and topical information in the area of numerical computations.

Prerequisite(s)
Special topics courses are not offered on a regular basis, but whenever departmental resources permit. For this reason the prerequisites can vary each time the courses are offered.

Successor(s)
None

Representative Workload
Variable

Representative Course Outline
Variable

Comments or Notes
All relevant information on a special topics course is posted on the Computer Science website and the bulletin board outside the Computer Science General Office several weeks before the beginning of the semester in which the course is being offered.
COMP 4741
Formal Languages and Computability

Students Interested
This course is of interest to those students seeking a deeper understanding of classical formal language theory and computability.

Objectives of the Course
This course is an in-depth discussion of classical models of computation, their computational power and their use in the classification of problems into classes. In addition, the correspondence between the models of computation and the different types of grammars is established.

Prerequisite(s)
COMP 3602 or the former COMP 3719

Successor(s)
Special Topics in Theoretical Aspects

Representative Workload
Assignments (5) 40%
Midterm Exam 25%
Final Exam 35%

Representative Course Outline
• Review of mathematical preliminaries: sets, binary relations, equivalence relations, partial orders, functions, finite and infinite sets, countable and non-countable sets, alphabet strings, string operations, languages, operations on languages
• Finite state automata, minimization, nondeterminism, closure properties of regular languages, regular expressions, pumping lemma
• Pushdown automata, context-free languages and grammars, equivalence, ambiguity, pumping lemma, parsing
• Turing machines, nondeterminism, multiple tapes, recursive and recursively enumerable languages
• The Chomsky hierarchy
• Decidability of problems concerning regular languages, context free languages and general languages
• Undecidability of the Halting Problem
• Reducibility and its application in proving undecidable and decidable languages
• The Post correspondence problem
• Oracle reductions and the arithmetic hierarchy
• The Recursion Theorem and its applications

Comments or Notes
• Students can receive credit for only one of Computer Science 4741 or the former Computer Science 3740.
COMP 4742
Computational Complexity

**Students Interested**
This course is of interest to students wishing to deepen their understanding of the nature of problem complexity.

**Objectives of the Course**
This course is an in-depth look at the theory of algorithms and algorithm complexity from a structural point of view. The emphasis will be placed on complexity classes containing problems of practical relevance such as P, NP and the parallel class of problems NC.

**Prerequisite(s)**
COMP 3602 or the former COMP 3719

**Successor(s)**
None

**Representative Workload**
Assignments 40%
In-class Exams 20%
Final Exam 40%

**Representative Course Outline**
- Review of the basic formal models of computation and complexity
  - random access machines
  - Turing machines
  - oracle machines
  - alternating Turing machines
  - combinational circuits model
  - uniform and nonuniform complexity measures
  - resource bounded computations
- Complexity classes
  - resource bounded reducibility (Turing-Cook, polynomial time, logarithmic space)
  - the classes NP, P, NC, PSPACE, LOGSPACE and their complements
  - problems complete and hard for a complexity class
  - relationships between complexity classes
- The polynomial time hierarchy
- Randomized computations
  - randomized algorithms
  - randomized complexity classes
  - randomized sources
COMP 4743
Graph Algorithms and Combinatorial Optimization

Students Interested
This course is of interest to students wanting to deepen their understanding of graph and network optimization problems.

Objectives of the Course
To give students efficient algorithms for solving some graph and network optimization problems and to provide them with techniques to show the apparent intractability of others.

Prerequisite(s)
COMP 3600 or the former COMP 3719

Successor(s)
None

Representative Workload
Assignments (5) 40%
Midterm Exam 25%
Final Exam 35%

Representative Course Outline
• Graph theory fundamentals
• Algorithms on graphs: graph connectivity and traversals, matching, shortest path, isomorphism, testing membership on families of graphs: bipartite, planar, of bounded tree-width
• Some NP-complete and hard problems on graphs, colourability, independent sets, vertex cover, clique
• Approximation algorithms for some graph theoretic problems
• Resource scheduling problems
• Greedy algorithms and scheduling problems, dynamic programming algorithm for scheduling of weighted intervals
• The maximum flow problem and the Ford-Fulkerson algorithm, maximum flow and minimum cuts, the preflow-push maximum-flow algorithm
• Applications of network flow
COMP 4745-4749 (excluding COMP 4748)  
Special Topics in Theoretical Aspects

Students Interested  
Those who want a more in-depth study in the area of theoretical computing.

Objectives of the Course  
To give students exposure to current and topical information in the area of theoretical computing.

Prerequisite(s)  
Special topics courses are not offered on a regular basis, but whenever departmental resources permit. For this reason the prerequisites can vary each time the courses are offered.

Successor(s)  
None

Representative Workload  
Variable

Representative Course Outline  
Variable

Comments or Notes  
All relevant information on a special topics course is posted on the Computer Science website and the bulletin board outside the Computer Science General Office several weeks before the beginning of the semester in which the course is being offered.
COMP 4750
Introduction to Natural Language Processing

Students Interested
The ever-increasing number of people communicating with computer applications (either via stand-alone devices or over the internet) has led to a corresponding ever-increasing demand that this communication be carried out via natural human languages, either as written text or speech. In this course, an introduction will be given to Natural Language Processing (NLP), including an integrated systematic examination of the full range of rule-based and statistical techniques used in NLP.

Objectives of the Course
This course covers tasks involving human languages, such as speech recognition, text understanding, and keyword-based information retrieval which underlie many modern computing applications and their interfaces. To be truly useful, such natural language processing must be both efficient and robust. This course will give an introduction to the algorithms and data structures used to solve key NLP tasks, including utterance understanding and generation and language acquisition, in both of the major algorithmic paradigms used today (rule-based and statistical). The emphasis will be primarily on text-based processing though speech-based processing will be addressed where possible.

Prerequisite(s)
COMP 3600 or the former COMP 3719

Successor(s)
None

Representative Workload
Assignments 20%
In-Class Exams (2) 40%
In-Class Presentation(s) 10%
Course Project 30%

Representative Course Outline
- Overview of Natural Language Processing (1 week)
- Background: Linguistics and Language Processing (overview of classical linguistics; representations of natural language utterances, grammars, and lexicons; implementation of processes on natural language representations) (3 weeks)
- Utterance comprehension (2 ½ weeks)
- Utterance production (1 week)
- Language acquisition (1 week)
- Special applications, e.g., language-language translation, question answering, text mining (1 ½ weeks)
- Student presentations (2 weeks)
COMP 4754
Database Systems

Students Interested
This course will be of interest to those who wish to have an introduction to the problems involved in dealing with large amounts of data.

Objectives of the Course
To introduce students to database processing, database management systems and database design considerations. Additional topics covered include the theory and methodologies essential for the relational database design, implementation, manipulation, optimization and management.

Prerequisite(s)
COMP 2004 or the former COMP 3725, and COMP 2007 or the former COMP 3754

Successor(s)
Special Topics in Database Systems

Representative Workload
Assignments and Project 40%
Midterm Exam 20%
Final Exam 40%

Representative Course Outline
- Introduction to databases and database management systems
- Entity relationship model
- Database design
- Requirement analysis
- Conceptual modelling
- Implementation design
- Physical design
- Modelling of data for various database models
- Dependencies, normal forms and other database design considerations
- Some sample query languages and query processing
- Query optimization
- Concurrency
- Recovery
- Security and integrity
- Distributed databases
Students Interested
It is becoming more difficult to avoid hearing about the Internet and its promise to hook-up the world. The purpose of this course is to show how the Internet really works. It is concerned with exactly how one computer successfully communicates with a “net” of other computers.

Objectives of the Course
The main objective of this course is to study the protocols, the design of protocols, and the implementations of these protocols used to communicate between computers. Several standard protocols will be examined.

Prerequisite(s)
COMP 2006 or the former COMP 3715, and COMP 2004 or the former COMP 3725

Successor(s)
None

Representative Workload
Projects 40%
Tests 20%
Final Exam 40%

Representative Course Outline
• Basic terms of communication networks
• The Berkeley Sockets
• The data line layer
• The physical layer
• Error/correcting and detecting codes
• Local area networks
• The network layer
• Transport, session and presentation
• Encryption
• Applications
Students Interested
Autonomous robotics is a very active research area in computer science. It addresses fundamental questions of how an agent can move, navigate, and carry out high-level tasks in unknown and noisy environments. This course will introduce students to algorithms and technologies that have emerged from research in autonomous robotics. These concepts are critical in any study of robotics systems. Further, they have wider applicability in various industries where properties of the real world must be sensed, modelled, and acted upon.

Objectives of the Course
This course will introduce students to the fundamental constraints, technologies, and algorithms of autonomous robotics. The focus will be on computational aspects of autonomous wheeled mobile robots. The most important themes will be mobility, perception, and navigation. Assignments will require the implementation of controllers for robots using the Webots and the LEGO Mindstorms NXT robot kits.

Prerequisite(s)
COMP 2002 or the former COMP 2711, Mathematics 2000, Mathematics 2050, and Statistics 2550 or any of the courses listed in the credit restrictions of Statistics 2550

Successor(s)
None

Representative Workload
Assignments (5) 40%
Midterm Exam 20%
Final Exam 40%

Representative Course Outline
- Introduction - Major paradigms in robotics
- Mobility - Methods of locomotion; kinematics; simple control systems
- Perception - Sensor technologies; stereo vision; modelling uncertainty of sensors and positional information
- Localization and Navigation - Environmental representation; Kalman and particle filtering; simultaneous localization and mapping (SLAM)
- Motion Planning - 2-D path planning; obstacle avoidance

Comments or Notes
- In addition to three hours of lectures per week, there is a structured laboratory as scheduled in the university timetable.
Students Interested
This course is of interest to students who wish to develop software in a networked mobile environment.

Objectives of the Course
The purpose of this course is to introduce students to the unique challenges of developing software in a networked mobile environment. Four fundamental aspects will be addressed in this course: multi-touch interface, network computing in a mobile environment, graphics programming for mobile devices, and human-computer interaction for mobile devices. In addition to these general topics, there will be components of the course that are specific to the iOS and Android OS software development kit.

Prerequisite(s)
COMP 2008 or the former COMP 2760, COMP 2006 or the former COMP 3715, and COMP 2005 or the former COMP 3716

Successor(s)
None

Representative Workload
Assignments (4) 20%
Milestone-based Project 40%
Presentation/Demo 5%
Final Exam 35%

There will be one assignment for each major component of the course. The milestone-based project will be a group project with specific design/development milestones that must be met throughout the course. Each project will be required to contain some element from each of the four major components of the course. A short presentation of the final outcomes of the project will be required. A weekly lab will provide tutorials to assist students in becoming familiar with the tools in the SDK. There will be no marks assigned in the lab. Student groups will be given a choice from a selection of project topics. For each of these, the general requirements will be provided. Examples include multi-device ping-pong, network tic-tac-toe, instant messaging, and location-aware search.

Representative Course Outline
- iOS and Android OS development work flows
  - IDE
  - iPhone simulator
  - Device builder
  - testing and debugging
- Network computing in a mobile environment
- location services
- context-aware mobile computing

- Graphics programming for mobile devices
  - OpenGL ES
  - game design for mobile devices

- Human-computer interaction for mobile devices
  - multi-touch
  - motion sensor

**Comments or Notes**
- In addition to three hours of lectures per week, there will be a laboratory of one and one-half hours per week.
COMP 4770
Team Project

Students Interested
This course is required for all computer science majors who are following regulations prior to the 2016 - 2017 academic year.

Objectives of the Course
The objective of this course is to develop a working prototype of a software system as a team effort. A group of students will work on a project for a term, experiencing the advantages and difficulties of team projects.

Prerequisite(s)
COMP 2003, COMP 2005, COMP 2006, COMP 2007, and six credit hours in Computer Science course at the 3000-level or above; or the former COMP 3715, the former COMP 3716, the former COMP 3724, and the former COMP 3754

Successor(s)
None

Representative Workload

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Percentage</th>
<th>Effort</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Document</td>
<td>25%</td>
<td>(team effort)</td>
</tr>
<tr>
<td>Architecture Document</td>
<td>15%</td>
<td>(team effort)</td>
</tr>
<tr>
<td>Module Document</td>
<td>35%</td>
<td>(individual effort)</td>
</tr>
<tr>
<td>System Demonstration</td>
<td>25%</td>
<td>(team effort)</td>
</tr>
</tbody>
</table>

Representative Course Outline
The following milestones are required for a Computer Science 4770 project:

- Requirements Document
  - This document must describe the use cases and functional and non-functional requirements of the software system, as well as a schedule of the development tasks and the team member(s) responsible for each task.
  - Use cases are a requirements capturing technique.

- Architecture Document
  - The document must present the system’s decomposition into modules and the assignment of responsibility of team members to subsets of the modules.

- Module Documents
  - These documents must include the complete description of the functionality and interface of each module. The module testing plans must be described and demonstrated.

- System Demonstration
- The integrated system must be demonstrated by the team. The integrated system includes all software and data required to use the system. It contains an integration of all the modules.
- A document describing the system testing must be submitted.

The milestones should be reached in the 4th, 6th, 9th and 13th week (respectively) of the semester.

Comments or Notes
- Attendance is required.
- Students register for Computer Science 4770 as a regular three-hour per week course. Due to the nature of this course, CICS students on a placement should not register for Computer Science 4770 as well.
- Prior to the beginning of the term, registered students must submit a short CV indicating any experience or expertise relevant to the course.
- In the time slot assigned to the course, groups will meet to discuss their progress. A graduate student may be assigned to each group for help and advice.
COMP 4780
Honours Project

Students Interested
This course is required of computer science majors who have been accepted into an honours program.

Objectives of the Course
To introduce computer science honours students to research activities, to familiarize these students with a special problem in computer science and to provide independent study on an advanced topic under the direct supervision of a member of the computer science faculty.

Prerequisite(s)
Admission to the honours program and permission of the Head of the Department. All required core courses and certain 4000 level computer science courses. Students normally register for this course during their last semester in the program.

Successor(s)
None

Representative Workload
- Students are expected to spend as much time on this course as is spent on any 4000-level computer science course. An honours student should meet regularly with his/her chosen supervisor to discuss the progress of the work and the future direction it should take.

- The topic is decided in consultation with the supervisor. The student is required to produce a written report on the project, to include the literature search on the topic and to present this work at a departmental seminar prior to the last week of the semester. The final grade received will depend on the written report, the seminar presentation and the discussions held between the student and supervisor, with the weighting of each of these to be at the discretion of the supervisor.

- No grade will be submitted for this course until all of the required work has been completed.

Representative Course Outline
Not applicable

Comments or Notes
Registration for this course is by Course Change form only.

An honours degree offers the student greater specialization in a particular area of computer science and as such, requires higher than average academic achievement. An honours degree is frequently a requirement for admission to an M.Sc. program. Any student interested in enrolling in an honours program is advised to consult with the Head of Department at their earliest convenience.
For the degree **Honours in Computer Science (Software Engineering)**, the honours project must be in the area of software engineering.
Students Interested
Those who want a more in-depth study in a specified area.

Objectives of the Course
To give students exposure to current and topical information.

Prerequisite(s)
Special topics courses are not offered on a regular basis, but whenever departmental resources permit. For this reason the prerequisites can vary each time the courses are offered.

Successor(s)
None

Representative Workload
Variable

Representative Course Outline
Variable

Comments or Notes
All relevant information on a special topics course is posted on the Computer Science website and the bulletin board outside the Computer Science General Office several weeks before the beginning of the semester in which the course is being offered.