UTM, Major Calendar Changes, 2011-12
Appendix A: Creation of the Minor – Computer Science, B.Sc.

UNIVERSITY OF TORONTO MISSISSAUGA
NEW PROGRAM FORM for 2011-2012 CALENDAR

1. Program, Degree, Department	Computer Science Minor, HBSc, Mathematical and Computational Sciences (MCS)

Minor Program ERMIN1688 Computer Science

4.0 credits are required.

As a discipline, computer science is concerned with the study of computation and its application across a wide variety of domains. The computer science minor provides an introduction to the tools and thought processes used by computer scientists as well as the opportunity to focus on a specific application area in the third and fourth year. Graduates will be capable of applying computational thinking in domains like biology, statistics, and commerce.

First Year: CSC108H5, 148H5; MAT102H5
Second Year: CSC207H5, 236H5; one of (CSC209H5, 258H5, 263H5)
Third and Fourth Years: Two half courses from any 300/400 level U of T Mississauga CSC courses, except for CSC492H5 and CSC493H5.

Notes:
1. All third year courses in computer science have a writing requirement. The recommended course to satisfy that requirement is CSC290H5. If a student wishes to substitute another course to satisfy the writing requirement, the student should consult the Computer Science Faculty Advisor.
2. In order to enter this program, you must have completed Grade 12 Advanced Functions (MHF4U) or equivalent, which is a prerequisite for MAT102H5.
3. Students enrolled in this program may participate in the PEY program. For more information visit **www.pey.utoronto.ca**

2. Academic Rationale

Computer science is split between two approaches: a theoretical approach that stresses the relationship between computation and mathematics and an “applications” approach that places computation within the context of a domain that uses it. At the University of Toronto Mississauga (UTM), the Department of Mathematical and Computational Sciences (MCS) currently offers three specialist programs and a major that are related to computer science. The specialists are designed to provide a strong background in both approaches and to develop a deep understanding of their application within a specific domain. In contrast, while the Computer Science Major provides an introduction to both approaches, students are only expected to develop a deep understanding of a single approach and its use within a domain. All four programs are designed to enable graduates to be practicing computer scientists, but we have identified a need for a program that will enable graduates from other domains to apply computer science techniques to their work.

The proposed Computer Science Minor is designed to fill this need for a strong introduction to the tools used by computer scientists without the depth required by the major and specialist programs. The required courses have been chosen to provide an introduction to both the theoretical and applied approaches that computer scientists use. At the same time, these courses will allow students to enroll in a wide range of upper year half-courses that could reasonably be expected to be applicable to studies in other areas. For example, students in the life sciences could enroll in CSC321H5 (machine learning) and CSC338H5 (numerical methods), and commerce students interested in e-commerce could choose to enroll in CSC343H5 (databases) and CSC309H5 (web programming). There may also be possibilities for future collaborations with the ICCIT.
In particular, we believe this program will be of interest to students in the Concurrent Teacher Education Program (CTEP) and students wishing to attend teachers college. If these students complete 4.0 credits of computer science, they are certified to teach computer science. We believe that CTEP students with an anchor subject of mathematics are excellent candidates for picking a second teachable in computer science. CSC300H5 (computers and society), CSC320H5 (visual computing), CSC324H5 (programming languages), and CSC384H5 (artificial intelligence) would all provide background that could be used in a secondary schooling environment.

58499

3. Learning Outcomes

The Computer Science Minor provides a strong introduction to the theoretical and applied tools used by computer scientists. A graduate of this Minor shall be able to:

1. Identify opportunities for applying computational tools and thought processes to a problem.
2. Design and implement computer programs.
3. Communicate effectively as a member of a technical team that is analyzing a problem and developing a solution.
4. Employ counting techniques and induction proofs to verify the correctness of solutions.

Theoretical Approaches: Students will be exposed to mathematical logic and proof techniques in their first year (MAT102H5) and will use these techniques to verify the correctness of programs in the second year (CSC236H5). They will also be exposed to the idea of "complexity" and how it affects our ability to solve problems efficiently – or solve them at all (CSC148H5).

Applied Approaches: In the first year, students will learn how to analyze real-world problems and to implement solutions in the form of computer programs (CSC108H5, CSC148H5). They will build on this knowledge in the second year by learning a new programming language and applying that knowledge in a large, group project (CSC207H5). The second year course focuses on tools to facilitate collaboration and communication within technical teams.

Key Courses in the Program:
CSC108H5, Introduction to Computer Programming
CSC148H5, Introduction to Computer Science
CSC207H5, Software Design
CSC236H5, Introduction to the Theory of Computation
MAT102H5, Introduction to Mathematical Proofs

4. Program Expectations

1. **DEPTH AND BREADTH OF KNOWLEDGE**

Computer Science is a broad, practical discipline. By the end of the second year, students in the minor will be exposed to tools and perspectives from both the theoretical (CSC236H5) and applied (CSC207H5) approaches used in the discipline.

Depth is achieved using the choice of second year follow-on course and the third year requirement. Students are asked to select courses that can be applied to their own discipline, so that they can use computational approaches in a context outside of computer science. A graduate of this program will be able to apply a limited number of advanced computational tools in his or her home discipline.

2. **KNOWLEDGE OF METHODOLOGIES**

The goal of this program is to expose students to a range of methodologies used by computer scientists. Students will be exposed to perspectives and tools including counting arguments and induction (MAT102H5, CSC236H5), complexity (CSC148H5), the use of state machines (CSC207H5, CSC236H5), and the development of software either individually or in technical teams (CSC108H5, CSC148H5, CSC207H5).

3. **APPLICATION OF KNOWLEDGE**

Computer science is an applied discipline. In the first year of the program, students are expected to learn how to program a computer. Programming is a practical but creative activity that requires strong problem solving skills, the ability diagnose errors through testing, and the development of sound stylistic and design judgement. Students in the minor are expected to continue to improve and apply their programming skills in the knowledge domains introduced by courses in the second, third, and fourth years.

In particular, students in the minor will be expected to apply computational techniques to knowledge domains from other disciplines. For example, students in statistics will be able to implement statistical techniques in programs that collect and mine digital information. Mathematics students will be able to write numerical simulations and analyses and will be aware of concerns with floating point error on modern machines. Students in sciences like psychology will be prepared to implement computer-presented or assisted experiments and will be able to manipulate the data that is generated by the experiment.

4. **COMMUNICATION SKILLS**

From the first year, students in the minor will be introduced to the necessity of working in teams and communicating ideas to non-technical audiences. In CSC108H5 and CSC148H5, students are expected to work in pairs to understand written problems and to design computer programs to solve them ("pair programming"). The programs they develop must be well documented so that other programmers and non-technical clients can make use of the system. In the second year software design course (CSC207H5), students continue to develop their technical communication skills by

authoring system specifications, design documents, and audit trails for their peers, "bosses", and non-technical clients. In that course, they work in teams of three and four and interact with other teams through code reviews, product proposals, and product release presentations.

5. AWARENESS OF LIMITS OF KNOWLEDGE

The theoretical approaches taught in computer science stress the need to understand the limits of the cognitive tools available to a computer scientist. Students learn to analyze possible solutions and prove them correct (CSC236H5), and they learn to estimate how difficult a problem is by looking at its complexity and can generate an approximation of complexity by measuring program runtime (CSC148H5). The software development practices taught in CSC207H5 also contribute to students' ability to cope with ambiguity. In that course, the students are expected to identify requirements specified by a customer and to account for unstated or hidden requirements.

6. **AUTONOMY AND PROFESSIONAL CAPACITY**

Computer science courses stress teamwork and effective collaboration (CSC108H5, 148H5, 207H5). In particular, students work in teams of two in the first programming courses ("pair programming"), and they learn agile development practices and larger team collaboration strategies in the second year programming course. All of these courses discuss professionalism and responsibility, effective communication, and the need for attribution, and within their teams, students uphold their personal responsibilities to maintain their reputation within their cohort.

At the same time, students in the minor will be expected to demonstrate individual and independent mastery of the material. In both the first and second year, the programming courses discuss appropriate methods for learning a language and finding information about resources available within a specific language. These skills translate to the upper year courses, where students are expected to pick up software tools independently. Both of these skills are necessary, since computer scientists are expected to work in large teams but to rapidly and independently become proficient with new tools and technologies.

[redacted]

Estimated Enrolment per Academic Year in this program (please explain)	We are aiming for an enrolment of 20 students. We believe this program will be an attractive minor for students in disciplines including Biology, Statistics, and Commerce. Additional accommodations for increased enrolments can be met if there is sufficient student interest. Depending on the demands of our current courses the possibility of an influx of student demand in either the new Minor or existing Major or Specialist programs may result in caps on enrolments to restrict course registrations; and or additional resources may have to be considered in order to accommodate the increased demand to allow for additional TAs or equipment where needed.
New courses necessary to mount this program	None. This program relies on courses already offered by the Department of Mathematical and Computational Sciences.
Additional Instructor(s) Requirements	None.
New Teaching Assistant(s) Requirements	If course enrolments increase in the first and second year courses, an additional tutorial section per affected course may be required to handle the expected 20 additional students.
New Laboratory Equipment Requirements	None.
New Computing Resources Requirements	None.
Other	None.

[redacted]

UTM, Major Calendar Changes, 2011-12
Appendix A: Creation of the Minor – Computer Science, B.Sc.

I will provide these resources required for this Program from my existing budget.	
October 19, 2010	Konstantin Khanin, Chair Department of Mathematical and Computational Sciences

UTM, Major Calendar Changes, 2011-12
Appendix A: Creation of the Minor – Computer Science, B.Sc.

List of Required Courses:

Year 1:

CSC108H5 Introduction to Computer Programming
CSC148H5 Introduction to Computer Science
MAT102H5 Introduction to Mathematical Proofs

Year 2:

CSC207H5 Software Design
CSC236H5 Introduction to the Theory of Computation

One of:
CSC209H5 Software Tools and Systems Programming
CSC258H5 Computer Organization
CSC263H5 Data Structures and Analysis

Years 3 & 4:
Two half courses from any 300/400 level U of T Mississauga CSC courses, except for
CSC492H5 Computer Science Implementation Project
and
CSC493H5 Computer Science Expository Work

Listing of Courses that Appear in the Proposal:

CSC108H5 Introduction to Computer Programming
CSC148H5 Introduction to Computer Science
MAT102H5 Introduction to Mathematical Proofs
CSC207H5 Software Design
CSC209H5 Software Tools and Systems Programming
CSC236H5 Introduction to the Theory of Computation
CSC258H5 Computer Organization
CSC263H5 Data Structures and Analysis
CSC300H5 Computers and Society
CSC309H5 Programming on the Web
CSC320H5 Introduction to Visual Computing
CSC321H5 Introduction to Neural Networks and Machine Learning
CSC324H5 Principles of Programming Languages
CSC338H5 Numerical Methods
CSC343H5 Introduction to Databases
CSC384H5 Introduction to Artificial Intelligence