



Computer Science Intermediate and Advanced

Teacher Contact Info:

Period 3	Mitchell Way / Karen Wilde	mdway@cbe.ab.ca	Room 2104
Period 4	Sonika Gupta	songupta@cbe.ab.ca	Room 2106

Course Description:

Students in Computer Science can expect to learn programming concepts including:

- Algorithms
- Design, debugging, and testing software
- Control flow statements
- Object-oriented programming
- Processing input and output
- Data structures

Course Objectives and Assessment:

As this is a Career and Technology Studies (CTS) course, all curricular outcomes will be drawn from the [Alberta CTS Program\(s\) of Studies](#)

Competencies – 20%

In CTS at Nelson Mandela the following competencies will be assessed:

Competency	Descriptor
Information Management	Students demonstrate this competency when they identify and clarify clear criteria for problems, explore a variety of problem-solving techniques, and then select and defend the most viable option.
Personal Management	Students demonstrate this competency when they take personal responsibility for their habits and wellbeing. This includes career skills like punctuality and professionalism as well as striving for personal excellence.
Communication and Collaboration	Students demonstrate this competency when they communicate clearly and effectively with a wide variety of audiences and peers. It also includes the effective use of communication technologies and the application of strategies to be an effective collaborator in a group.

Innovation and Creativity	Students demonstrate this competency when they take risks when exploring a variety of creative processes. They also adapt and persevere when exploring ways to create value and achieve excellence.
Complex Problem Solving	Students demonstrate this competency when they draw on multiple perspectives, disciplines and resources to select the most viable solution. They also approach all problems with optimism and hope.
Career Connections	Students demonstrate this when they effectively research and understand the post secondary and career options that are available to them.

Outcomes –

20% Knowledge and Understanding and

60% Plan, Create, Evaluate

Some credits have no Knowledge and Understanding outcomes in which case there is a 80%/20% Competency/Outcome split

Students taking Computer Science in their second or third semester of the course will be working towards the Intermediate or Advanced Credits respectively.

These are the credits that may be offered in the course and they may change due to duplicate CTS credits in other courses and/or missing pre-requisites.

Intermediate Credits

Credit	Outcome
CSE2010: Computer Science 2	Students explore hardware, software and processes at an intermediate level. Students extend their understanding of software development by learning how to layer modular programming approaches over structured programming techniques to improve the efficiency and robustness of algorithms and programs. They also are introduced to derived data types to provide them with data structures suitable for more demanding problems. Students add to their understanding of the hardware side of computer science by exploring a stylized von Neumann computer system at the machine level, and of the social side of computer science by examining some of the issues that have arisen from the implementation of computer technology
CSE2140:	Students who have mastered the basics of one programming language are given the opportunity to learn the basics of another. Designed for students who have learned how to write

Second Language Programming 1	structured and/or modular programs in a more accessible programming environment, this course gives students an opportunity to develop a similar skill set in a more demanding language. In the process, they have a further opportunity to hone their structured and modular programming skills.
CSE2120: Data Structures 1	Students learn how to design code and debug programs that use a set of data structures that can be used to handle lists of related data. Building on their knowledge of basic or primitive data types, they learn how to work with fundamental data structures such as the array and the record. As part of this process, they learn what types of problems benefit from the use of these types of data structures.
CSE3120: Object-Oriented Programming 1	Students add to their understanding of programming paradigms by moving from a procedural programming approach, in which modularity is handled through subprograms, to an object-oriented approach, in which it is handled through objects. They learn a simple object-oriented analysis and design approach based on the use of object diagrams and write programs that use objects associated with one another in a client/server relationship.
CSE2910: CSE Project B	Students develop project design and management skills to extend and enhance competencies and skills in other CTS courses through contexts that are personally relevant.

Advanced Credits

Credit	Outcome
CSE3010: Computer Science 3	Students explore hardware, software and associated processes at an advanced level. They extend their understanding of software development by moving from procedural programming approaches to an object-oriented approach. In the process they learn how object-oriented programming (OOP) can improve the efficiency and robustness of algorithm development and program construction. They deepen their understanding of the hardware side of computer science by exploring the connection between the binary/hexadecimal number systems and some of the simple logic gates that are the basis of the von Neumann computer. They also add to their understanding of the social implications of computer science by examining the emerging information society.
CSE3110: Iterative Algorithm 1	Students learn a number of standard iterative data processing algorithms useful for working with data structures such as arrays. These include an iterative version of the binary search, the three basic sorts—exchange (bubble), insertion and

	selection, and a simple merge. In the process, they learn when and where to apply these algorithms.
CSE3130: Object-Oriented Programming 2	Students extend their knowledge of object-oriented programming (OOP). They add to their expertise in object-oriented design by using some of the techniques associated with the UML design approach and to their programming expertise by writing programs that explore association between classes. Students work with abstract classes, developing algorithms that employ the object diagram approach and programs that use templated classes, containment, and inheritance to promote reusability.
CSE3310: Recursive Algorithms 1	Students learn how to use a new program control flow mechanism called recursion. They then use this mechanism to write a number of basic recursive algorithms and programs such as a recursive version of the binary search, the quicksort and the merge sort.
CSE3910: CSE Project D	Students develop project design and management skills to extend and enhance competencies and skills in other CTS courses through contexts that are personally relevant.

Course Materials and Resources:

Students are expected to provide their own storage devices or use Google Drive to take electronic work home when needed.

During the course students may be required to use online tools to aid and further their learning such as code.org or replit.com. They will be required to sign up for an account using their school email and may need to share personal information including their name. For more information about the use of this information, please contact your teacher.

Style Guide

All assignments are to be formatted using the provided style guide on D2L. Assignments which are not following the style guide will be returned and students will be required to resubmit their work after ensuring their work is formatted correctly.

Java

Students will be using Java to code in most modules. Java can be downloaded/used for free at home or students can use tutorial time to complete unfinished work.