COMPUTING SCIENCE COURSE SUMMARIES

INTRODUCTORY

CSE1010: COMPUTER SCIENCE 1

Students explore hardware, software and processes. This includes an introduction to the algorithm as a problem-solving tool, to programming languages in general and to the role of programming as a tool for implementing algorithms.
Prerequisite: *None*

CSE1110: STRUCTURED PROGRAMMING 1

Students are introduced to a general programming environment in which they write simple structured algorithms and programs that input, process and output data, use some of the more basic operators and data types, and follow a sequential flow of control.
Prerequisite*: None*

CSE1120: STRUCTURED PROGRAMMING 2

Students work with structured programming constructs by adding the selection and iteration program control flow mechanisms to their programming repertoire. They write structured algorithms and programs that use blocks to introduce an element of modularity into their programming practice.
Prerequisite: *CSE1110: Structured Programming 1*

CSE1210: CLIENT-SIDE SCRIPTING 1

Students are introduced to Internet computing through the use of one or more Web-specific markup languages. As part of this process, students learn how the Web uses markup languages to provide a client-side approach to display static information. Students also learn how to analyze, modify, write and debug algorithms and documents that use a markup language.
Prerequisite: *None*

CSE1220: CLIENT-SIDE SCRIPTING 2

Students deepen their understanding of Internet computing by using more advanced markup language techniques and by being introduced to one or more Web-specific scripting languages. As part of this process, students learn how the Web uses these resources as a means of displaying dynamic client-side information. Students learn how to analyze, modify, write and debug algorithms and scripts that use structured programming approaches.
Prerequisite: *None*
Note: *CSE1210: Client-side Scripting 1 or an equivalent course dealing with markup scripting is strongly recommended*

CSE1240: ROBOTICS PROGRAMMING 1

Students use an appropriate Robot Control Language (RCL) to design, develop, implement and debug robotics programs that employ standard structured programming constructs and simple data structures. In the process, they develop a general understanding of robots and the robotics environment.
Prerequisite: *CSE1110: Structured Programming 1*

CSE1910: CSE PROJECT A

Students develop project design and management skills to extend and enhance competencies and skills in other CTS courses through contexts that are personally relevant.
Prerequisite: *None*

INTERMEDIATE

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.

Prerequisites: *CSE1010: Computer Science 1*
CSE1120: Structured Programming 2

CSE2110: PROCEDURAL PROGRAMMING 1

Students develop their understanding of the procedural programming paradigm. They move from a structured programming approach in which modules were handled through the use of program blocks to a more formal modular programming approach in which they are handled through subprograms. In the process, students also learn to use a number of new design approaches made possible by the new paradigms. As part of this process, they also learn what types of problems are amenable to modular algorithms and programs.

Prerequisite: *CSE1120: Structured Programming 2*

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.

Prerequisite: *CSE2110: Procedural Programming 1*

CSE2130: FILES & FILE STRUCTURES 1

Students learn how to design, code and debug programs that use data files to store and retrieve data on secondary storage devices. Building on their knowledge of derived data structures, they learn how to use those structures to organize data for efficient file handling. As part of this process, they learn what types of problems benefit from the use of external files.

Prerequisite: *CSE2120: Data Structures 1*

CSE2140: SECOND LANGUAGE PROGRAMMING 1

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 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.

Prerequisites: *CSE2110: Procedural Programming 1* ***or***
CSE1120: Structured Programming 2

CSE2210: CLIENT-SIDE SCRIPTING 3

Students add to their understanding of Internet scripting by employing procedural programming techniques and fundamental data structures to create both static and dynamic client-side sites. Students learn how to analyze, modify, write and debug algorithms and scripts that use subprograms such as functions and data structures such as arrays.

Prerequisites: *CSE1220: Client-side Scripting 2*
CSE1120: Structured Programming 2

CSE2240: ROBOTICS PROGRAMMING 2

Students add to their understanding of robotics programming by employing procedural programming techniques and fundamental data structures to create programs that display greater agency and autonomy. They learn how to analyze, modify, write and debug robotics algorithms and programs in which modularity is achieved through subprograms such as functions and fundamental data structures such as arrays.

Prerequisites: *CSE1240: Robotics Programming 1*
CSE1120: Structured Programming 2

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.

Prerequisite: *None*

CSE2920: CSE PROJECT C

Students develop project design and management skills to extend and enhance competencies and skills in other CTS courses through contexts that are personally relevant.

Prerequisite: *None*

CSE2950: CSE INTERMEDIATE PRACTICUM

Students apply prior learning and demonstrate the attitudes, skills and knowledge required by an external organization to achieve a credential/credentials or an articulation.

Prerequisite: *None*

ADVANCED

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.

Prerequisites: *CSE2010: Computer Science 2*
CSE2110: Procedural Programming 1

CSE3020: COMPUTER SCIENCE 4

Students enhance their learning by studying a set of standard abstract data types and the dynamic data structures conventionally used to implement them. They also add to their general understanding of algorithms by learning how to conduct asymptotic analyses of algorithmic efficiency and indicate that efficiency using big O notation. Students continue their exploration of the hardware aspect of computer science by exploring a different type of computer architecture, the Turing machine.

Prerequisites: *CSE3010: Computer Science 3*
CSE3110: Iterative Algorithm 1

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.

Prerequisites: *CSE2120: Data Structures 1*

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.

Prerequisite: *CSE2110: Procedural Programming 1*

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.

Prerequisite: *CSE3120: Object-oriented Programming 1*

CSE3140: SECOND LANGUAGE PROGRAMMING 2

Designed for students who have mastered procedural programming and static data structures in a more accessible programming environment, this course gives students the opportunity to develop a similar skill set in a more demanding language.

Prerequisite: *CSE2120: Data Structures 1*

CSE3210: SERVER-SIDE SCRIPTING 1

Students add to their ability to craft dynamic Web sites by exploring the fundamentals of server-side scripting. In the process, they add to their understanding of Internet scripting by employing databases as a repository for the information to be displayed by their sites. Students learn how to analyze, modify, write and debug algorithms and server-side scripts that use simple databases.

Prerequisites: *CSE2210: Client-side Scripting 3*
CSE2110: Procedural Programming 1
CSE2120: Data Structures 1

CSE3240: ROBOTICS PROGRAMMING 3

Students continue their work in robotics programming by adding object-oriented programming (OOP) approaches to their skill set. In the process, they learn how to adapt their older procedure-based approaches to an object-oriented approach. They learn how to use object-oriented design approaches to design and write programs that use objects associated with one another in a client/server relationship.

Prerequisites: *CSE2240: Robotics Programming 2*
CSE2110: Procedural Programming 1

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.

Prerequisites: *CSE3110: Iterative Algorithm 1*
CSE3120: Object-oriented Programming 1

CSE3320: DYNAMIC DATA STRUCTURES 1

Students learn how to design, code and debug programs using abstract data types that utilize dynamic data structures. Students explore dynamic memory allocation, in general, and as handled by their programming environment. Students concentrate on how the linked list dynamic data structure(s) can be used to implement abstract data types.

Prerequisite: *CSE3310: Recursive Algorithms 1*

CSE3330: DYNAMIC DATA STRUCTURES 2

Students enhance their knowledge of abstract data types that utilize dynamic data structures by expanding their repertoire to include stacks and queues. Students also study the unordered data structures, set and map, and learn how to incorporate them into abstract data types. As part of this work, they learn how to use linked lists to create stacks and queues.

Prerequisite: *CSE3320: Dynamic Data Structures 1*

CSE3340: DYNAMIC DATA STRUCTURES 3

Students enhance their knowledge of abstract data types that utilize dynamic data structures by expanding their repertoire to include hierarchically linked data structures. Students study general trees, binary trees, binary search trees and heaps. They learn how to use binary search trees to implement sorted sets, sorted maps and heaps to implement priority queues.

Prerequisite: *CSE3330: Dynamic Data Structures 2*

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.

Prerequisite: *None*

CSE3920: CSE PROJECT E

Students develop project design and management skills to extend and enhance competencies and skills in other CTS courses through contexts that are personally relevant.

Prerequisite: *None*

CSE3950: CSE ADVANCED PRACTICUM

Students apply prior learning and demonstrate the attitudes, skills and knowledge required by an external organization to achieve a credential/credentials or an articulation.

Prerequisite: *None*