CPRG 216 - Object-Oriented Programming 1

Description
This course explores the evolution of computer programming and examine common problem-solving techniques. Emphasis is placed on examining code reuse in functions, scoping and abstraction. Additional topics will be explored such as, version control, and you'll use classes and object-oriented programming principles to create applications using an industry-standard programming language.

3 Credits

Time Guidelines
The standard instructional time for this course is 75 hours.

Equivalents
CPRG 2160

Course Assessment

Assignments	60%
Project(s)	40%
Total:	100%

Use of Turnitin

The instructor may submit student work in this course to Turnitin's text-matching software program, to help assess the academic integrity of student work in this course. Turnitin results may be considered as one piece of evidence in academic misconduct hearings. Turnitin is an American company that keeps no unencrypted student identity data in the United States unless the students themselves choose to share this information in their submission. Students should limit their sharing of personally identifiable information by not including their names and student ID numbers within the text body of submitted assignments. SAIT has carefully reviewed this company's data management procedures.

Other Course Information
Learner Engagement:

In order to be successful, the learner is expected to be engaged in learning activities for a total of 9 to 12 learning hours per course per week, which includes both in-class and out-of-class time.

Course Learning Outcomes
1. Describe the evolution of computer programming.

Objectives:

1.1 Describe what computers do.

1.2 Define a programming language.

1.3 Illustrate the evolution of programming languages.

1.4 Describe how a computer interprets an algorithm.

1.5 Explain a program's flow of control.

1.6 Compare programming languages.

1.7 Describe the characteristics of a programming language.

2. Create solutions to programming problems.

Objectives:

2.1 Describe the rules of syntax in programming.

2.2 Apply syntax rules to writing a program.

2.3 Explain conditional statements and branching.

2.4 Demonstrate the use of conditional statements and branching.

2.5 Describe repetition structures.

2.6 Demonstrate the use of repetition structures.

3. Explain code reuse.

Objectives:

3.1 Describe functions in programming.

3.2 Describe lambda abstraction.

3.3 Apply debugging.

3.4 Apply keyword arguments and default values.

3.5 Apply a variable number of arguments.

3.6 Apply scoping of variables.

3.7 Explain the concept of functions as objects.

3.8 Describe reusing methods.

3.9 Apply global variables.

3.10 Use recursive methods.

3.11 Solve problems using a recursive function.

4. Apply classes and object-oriented programming principles in design projects.

Objectives:

4.1 Describe object-oriented programming.

4.2 Describe classes and their purpose.

4.3 Describe objects and their purpose.

4.4 Define attributes, class variables and instance variables.

4.5 Demonstrate how to create objects.

5. Apply version control.

Objectives:

5.1 Define version control.

5.2 Explain the importance of version control.

5.3 Describe Git and Github.

5.4 Explain the basics of using Github in software projects.

5.5 Demonstrate how to use Github commands.

SAIT Policies and Procedures:
Students should familiarize themselves with SAIT's Policies and Procedures, including but not limited to those in the Academic Student, Conduct, and Health, Safety, and Environment domains.