Course:	**COSC 180 - Introduction to Programming**
Instructor:	Alex Wang Rm. 348 (Regina Campus) wangxi@saskpolytech.ca
Course Description:	You will learn concepts used in object-oriented programming. You will create programs that use variables, allow for user input and output, and provide opportunities for simple decision strategies. You will also learn how to work with different variable types and how to debug programs. In addition, you will create and use strategies that involve repetition (looping) in your programs. You will create methods and work with elementary data collections (arrays). You will learn how to create object templates (classes) and create and utilize object in your programming. You will develop an understanding of inheritance and polymorphism, and you will utilize these object oriented techniques to solve problems.
Pre Requisites: **Co Requisites:**	• •
Course Hours: **Credit Units:**	• 96 • 6
Student Assessment: **Grade/Passing Grade:** **PLAR Method:**	• 3-4 Assignments (20%), 2 Midterms (20% each), Final (40%) • 50% • Evidence and Structured Interview
Learning Resources:	3 Liang, Y. Daniel. *Introduction to Java Programming: Comprehensive Version*. 12th ed. Prentice Hall, 2020.

Learning Outcomes:	1. Explain programming terminology. 2. Create a program using tools and styling conventions. 3. Perform elementary programming. 4. Use a debugging tool. 5. Create a program that uses strings and mathematical library routines. 6. Create a program that uses operators and decision statements. 7. Create a program using repetition structures. 8. Create a program using methods. 9. Use arrays to manage collections of primitive values or object references. 10. Create a program using objects and object oriented techniques. 11. Design reusable classes through inheritance and interfaces.
Prepared/Updated by: **Date:**	**Alex Wang** **August 2021**
Approved by Program Head: **Date:**	**Heath Armbruster** **August 2021**
Approved by Academic Chair: **Date:**	**Bill Walsh** **August 2021**

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
1. Explain programming terminology.	1.1. Explain what a program is. 1.2. Explain the different generation of languages: high-level, low-level and machine language. 1.3. Explain the different classifications of higher level languages: procedural programming and object oriented programming. 1.4. Explain the difference between compiling and interpreting a program. 1.5. Describe the characteristics of Java.	Lecture, reading assignment.
Assessment Tools: Written exam		

2. Create a program using tools and styling conventions.	2.1 Describe the process that is followed to create and execute a computer program. 2.2 Describe source code, bytecode and object code. 2.3 Create a program. 2.4 Compile the program using a compiler 2.5 Run the program from the command line 2.6 Run the program using an Integrated Development Environment (IDE). 2.7 Explain the difference between Semantic and Syntax errors 2.8 Describe the styling conventions. 2.9 Use the style conventions.	Lectures, Demonstrations, in-class computer activity, lab.
Assessment Tools: Assignment, quiz, written exam		

Learning Outcomes/ Assessment Tools	**Learning Steps**	**Learning Activities/Resources**
3. Elementary Programming	3.1 Identify primitive types. 3.2 Explain variables. 3.3 Create a program that declares, initializes and assigns variables. 3.4 Create a program that utilizes type casting. 3.5 Create a program that displays output 3.6 Create a program that can read variables of different types from the keyboard 3.7 Create programs that can perform standard Mathematical operations on variables.	Lecture, reading assignment, demonstration, in-class computer activity, lab.
Assessment Tools: Assignment, quiz, written exam		

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
Assessment Tools: Assignment, quiz, written exam		
4. Use a debugging tool.	4.1. Describe a debugging tool. 4.2. Explain breakpoints and watches. 4.3. Demonstrate how to step through a program. 4.4. Use debugging tool to find errors in program.	Lecture, demonstration, in-class computer activity, lab.
5. Write programs that utilize strings and mathematical libraries	5.1 Identify strings. 5.2 Demonstrate ability to concatenate strings. 5.3 Demonstrate ability to display strings. 5.4 Use standard string methods. 5.5 Describe what the math library is used for 5.6 Write a program that utilizes the math library to create random values	Lecture, reading assignment, demonstration, in-class computer activity, lab.
Assessment Tools: Assignment, quiz, written exam		

Learning Outcomes/ Assessment Tools	**Learning Steps**	**Learning Activities/Resources**
6. Create a program that uses operators and decision statements.	6.1 Identify arithmetic, increment and decrement operators. 6.2 Create a program that utilizes arithmetic, increment and decrement operators. 6.3 Utilize logical, equality and relational operators. 6.4 Solve operator expressions by following the order of precedence rules. 6.5 Create Boolean expressions combining logical, relational and equality operators. 6.6 Describe the logic of a decision statement. 6.7 Create a program with an “if statement”. 6.8 Create a program with an “if else statement”. 6.9 Create a program using nested “if statements”. 6.10 Create a program using “else-if statements”. 6.11 Create a program using a “Switch statement”. 6.12 Create a program using a Conditional Operator.	Lecture, reading assignment, demonstration, in-class computer activity, lab.
Assessment Tools: Assignment, quiz, written exam		

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
7. Create a program using repetition structures.	7.1 Describe the logic of a repetition structure. 7.2 Create a program using a for loop structure. 7.3 Create a program using an enumerated data type and the for each loop structure. 7.4 Create a program using a while loop structure. 7.5 Create a program using a do while loop structure. 7.6 Create a program using a sentinel controlled loop structure. 7.7 Create a program using nested loops.	Lecture, reading assignment, demonstration, in-class computer activity, lab.

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
Assessment Tools: Assignment, quiz, written exam		
8. Create programs using methods.	8.1 Explain methods. 8.2 Use pre-existing Java Math class methods. 8.3 Create a method that performs an action, but does not return a value or receive data. 8.4 Create a method that returns a single value. 8.5 Create a method passing in value(s). 8.6 Explain call-by-value. 8.7 Differentiate between an argument and a formal parameter. 8.8 Create an overloaded method. 8.9 Explain what a unit Test is 8.10 Create unit tests for simple methods	Lecture/Lab/Assignment in-class activity, lab
Assessment Tools: Assignment, quiz, written exam		

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
9. Use Arrays to manage collections of primitive values or object References	9.1 Define what an array is 9.2 Describe how an array is stored in memory 9.3 Use arrays to collect primitive values and object references 9.4 Access array elements 9.5 Use arrays as arguments to methods 9.6 Use multi-dimensional arrays.	Lecture/Lab/Assignment in-class activity, lab

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
10. Create a program using objects and object oriented techniques.	10.1 Create a class and an object. 10.2 Explain difference between a class and an object. 10.3 Explain public and private access modifiers. 10.4 Discuss encapsulation. 10.5 Identify characteristics of a well-encapsulated class user interface and implementation. 10.6 Explain the "this" pronoun. 10.7 Create constructors to initialize values. 10.8 Create a reference to an object. 10.9 Explain call-by-reference. 10.10 Create methods with objects as the parameter type. 10.11 Distinguish between instance and local variables. 10.12 Explain scope and duration of variables. 10.13 Create a static (class) method. 10.14 Explain wrapper class.	Lecture, reading assignment, demonstration, in-class computer activity, lab.

Learning Outcomes/ Assessment Tools	Learning Steps	Learning Activities/Resources
10. **cont'd …**	10.15 Use a wrapper class method to convert between strings and numbers. 10.16 Use wrapper class methods to perform operations on characters. 10.17 Create packages. 10.18 Create a program that imports packages.	Lecture, reading assignment, demonstration, in-class computer activity, lab.
Assessment Tools: Assignment, quiz, written exam		
11. Design reusable classes through inheritance and interfaces.	11.1 Design classes that inherit from another class. 11.2 Design abstract super classes and concrete sub classes. 11.3 Design interfaces to specify behaviors. 11.4 Explain how polymorphism makes systems extensible and maintainable. 11.5 Explain how polymorphism makes systems extensible and maintainable. 11.6 Create extensible classes that use inheritance and polymorphism.	Lecture, reading assignment, demonstration, in-class computer activity, lab.
Assessment Tools: Hand-in assignments, examinations		

Learning Outcomes/ Assessment Tools	**Learning Steps**	**Learning Activities/Resources**