THOMPSON RIVERS UNIVERSITY

Course Outline

Computing Science Department
Faculty of Science

COMP 1130 − 1
Computer Programming I (3,1,1)

Instructor:	**Phone/Voice Mail:**
Office:	**E-Mail:**

Calendar Description

Students are introduced to the use of structured problem solving methods, algorithms, structured programming, and object-oriented programming. Students use a high level programming language to learn how to design, develop, and document well-structured programs using software engineering principles. Students learn the workings of a computer as part of programming. This course is for students who plan to take further courses in Computing Science or to learn basic programming concepts.

Educational Objectives/Outcomes

Upon successful completion of the course, the student will demonstrate the ability to:

1. Understand the fundamental programming aspects and use of Java programming language.
2. Apply basic object-oriented programming concepts.
3. Design, develop, and document well-structured programs using software-engineering principles.
4. Use problem-solving skills to write software applications.

Prerequisites

None. Students with programming experience in another language should take COMP 2120.

Required Texts/Materials

1. Lewis, DePasquale and Chase; *Java Foundations, An Introduction to Program Design and Data Structures,3*rd ed, Pearson Education Inc., 2011, ISBN-13: 978-0-13-337046-1 or ISBN-10: 0-13-337046-1. An e-book/e-chapters is also available at http://www.coursesmart.com/IR/2170045/9780133449624?__hdv=6.8

2. TRU Lab/Network Computer Account.

Course Topics

Chapter 1. Introduction to Java	1 Week
Chapter 2. Data and Expressions	1.5 Weeks
Debugging (Instructor Notes)	0.5 Week
Chapter 3. Using Classes and Objects	2 Weeks
Chapter 4. Conditionals and Loops	2 Weeks
Chapter 5. Writing Classes	2.5 Weeks
Chapter 7. Arrays	2 Weeks
Chapter 17: Introduction to Recursion	1.5 Week

Syllabus - Lab Topics :

Lab Topics	Tool	Duration
Ch. 1: Introduction to the use of an appropriate IDE. Instructor to ensure students are familiar with using a Java Editor for writing and compiling Java code. Instructor to select appropriate exercise questions and programming project questions to the concepts presenting in Ch. 1.	Java Editor	2 hours
Ch. 2: Data and Expressions: Instructor to select appropriate exercise questions and programming project questions that would test student knowledge of the various concepts presented in Ch. 2.	Java Editor	4 hours
Ch. 3: Using Classes & Objects: Instructor to select appropriate exercise questions and programming project questions that would test student knowledge of the various concepts presented in Ch.3.	Java Editor	2 hours
Ch. 4: Conditionals & Loops: Instructor to select appropriate exercise questions and programming project questions that would test student knowledge of	Java Editor	4 hours

the various concepts presented in Ch.4.		
Ch. 5: Writing Classes: Instructor to select appropriate exercise questions and programming project questions that would test student knowledge of the various concepts presented in Ch.5.	Java Editor	6 hours
Ch. 7: Arrays: Instructor to select appropriate exercise questions and programming project questions that would test student knowledge of the various concepts presented in Ch.7.	Java Editor	4 hours
Ch. 17: Introduction to Recursion: Instructor to select appropriate exercise questions and programming project questions that would test student knowledge of the various concepts presented in Ch.17.	Java Editor	2 hours
Debugging and related testing concepts: On-going during most lab/seminar sessions of the semester.	Java Editor	2 hours

ACM / IEEE Knowledge Area Coverage

Knowledge Areas that contain topics and learning outcomes covered in the course

Knowledge Area	**Total Hours of Coverage**
AL/Fundamental Data Structures and Algorithms	2
PL/Object-Oriented Programming	3
PL/Basic Types Systems	0.5
SDF/Fundamental Programming Concepts	10
SDF/Fundamental Data Structures	2
SDF/Development Methods	6
PL/Functional Programming	1

Body of Knowledge coverage

KA	Knowledge Unit	Topics Covered	T1 hrs	T2 hrs	Elective hrs
AL	Fundamentals Data Structures and Algorithms	[Core-Tier1] • Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max, and mode in a list, approximating the square root of a number, or finding the greatest common divisor	2	0	0
PL	Object Oriented Programming	[Core-Tier1] • Object-oriented design ○ Decomposition into objects carrying state and having behavior • Definition of classes: fields, methods, and constructors [Core-Tier2] • Object-oriented idioms for encapsulation with privacy and visibility of class members • Using collection classes, iterators and other common library components	2	1	0
PL	Basic Types Systems	[Core-Tier1] • A type as a set of values together with a set of operations ○ Primitive types (e.g., numbers, Boolean) ○ Compound types built from other types (e.g., records, unions, arrays, lists, functions, references) • Association of types to variables, arguments, results, and fields	0.5	0	0
SDF	Fundamental Programming Concepts	• Basic syntax and semantics of a higher-level language • Variables and primitive data types (e.g., numbers, characters, Booleans) • Expressions and assignments • Simple I/O including file I/O	10	0	0

		• Conditional and iterative control structures • Functions and parameter passing • The concept of recursion			
SDF	Fundamental Data Structures	[Core-Tier1] • Arrays • Strings and string processing	2	0	0

SDF	Development Methods	[Core-Tier1]
		• Program comprehension
		• Program correctness
		  ○ Types of errors (Syntax, logic, run-time)
		  ○ Testing fundamentals and test-case generation
		  ○ Unit testing
		• Modern programming environments
		  ○ Programming using library components and their APIs
		• Debugging strategies
		• Documentation and program style
			6	0	0

PL	Functional Programming	[Core-Tier1] • First class functions (taking, returning, and storing functions)	1	0	0