ACADEMIC YEAR: 2021-2022

Course Number COMP 6481		**Course Title** Programming and Problem Solving	
Department Computer Science and Software Engineering			**Credits** 4.00
Level Graduate	**Prerequisites** MATH 204, 205 or equivalent, COMP 248 or equivalent		
Schedule **Lectures** – Mon @ 14:45 – 17:15 REMOTELY Meeting ID: 821 7226 0023 Wed @ 17:45 – 20:15 REMOTELY Meeting ID: 874 4037 7388 **Tutorials** – DD A and WW A on Tue @ 17:45 - 18:45 REMOTELY Tutor: TBA DD B and WW B on Tue @ 20:30 - 21:30 REMOTELY Tutor: TBA **Labs** – DD-I and WW I on Fri @ 11:30 - 14:30 in TBA Tutor: TBA DD-J and WW J on Fri @ 11:30 - 14:30 in TBA Tutor: TBA DD-K and WW K on Fri @ 11:30 - 14:30 in TBA Tutor: TBA DD-L and WW L on Fri @ 11:30 - 14:30 in TBA Tutor: TBA DD-M and WW M on Fri @ 11:30 - 14:30 in TBA Tutor: TBA DD-N and WW N on Fri @ 11:30 - 14:30 in TBA Tutor: TBA			
Instructor Kaustubha Mendhurwar Department of CSE	**Office Hours** (Appointments through email)	**Office** ER 1101	**Email** kaustubha.mendhurwar@concordia.ca

IMPORTANT NOTE: Due to the COVID-19 pandemic, this course is *partially* offered remotely. You must have all equipment, tools, software, etc. that are needed for a remotely delivered course, to take this course. All lectures will take place live over Zoom during class time. Additionally, all labs and exams will take place on-site at Concordia campus; so, you must be able to attend all these on-site activities to take the course.

BACKGROUND KNOWLEDGE

You should have some previous experience of programming in Java such as that provided in COMP 248 or a similar course. You should have a good understanding of expressions, statements, methods, parameters, and arrays. You should also know the basic concepts of objects, classes, and packages.

COURSE OBJECTIVES

The main objectives of this course are:

1. Understanding the constructs and the important aspects of Object-Oriented Languages and Object-Oriented Programming.
2. Gaining strong experience with designing and coding more sophisticated, robust, and modular software applications.

3. Gaining strong experience with problem analysis, problem solving, algorithms, and data structure, and additionally being able to code such algorithms/ solutions with ease.

In general, the course covers two distinct, yet quite related, subjects that can be described as: *Coding* and *Problem Analysis & Solving*! Consequently, and in relation to the first subject, course covers higher-level subjects of Object-Oriented programming, including the design of classes, inheritance, composition, polymorphism, static and dynamic binding, abstract classes, exception handling, file I/O, recursion, interfaces, inner classes, and linked lists. In relation to the second subject, the course covers data structure, abstract data types (*i.e.,* stacks, queues, double ended queues, *etc.*), trees, priority queues, dictionaries, heaps, hash tables, searching, sorting, search trees, and graphs, with a specific focus on time and space complexities and asymptotic notation.

RECOMMENDED TEXTBOOKS

During the course you will be using multiple sources; however, the two following text-books are particularly required:

1) M.T. Goodrich, R. Tamassia, Michael H. Goldwasser. *Data Structures and Algorithms in Java*, 6th edition. John Wiley & Sons, 2014. ISBN 978-1-118-77133-4. (Note: 5th edition is ok.) The book is available at the bookstore or can be rented as eTextbook. Textbook
URL: http://ca.wiley.com/WileyCDA/WileyTitle/productCd-EHEP002900.html

   From this textbook we target the coverage of these sections: 3.1 to 3.4, 4.1 to 4.3, 5.1 to 5.6, 6.1 to 6.2, 7.1 to 7.3, 8.1 to 8.4, 9.1 to 9.4, 10.1 to 10.3, 11.1 to 11.3, 12.1 to 12.4, 14.1 to 14.7, and 15.1 to 15.2. Due to time limitations, you may be required to read some of these subjects on your own. **You also need to study chapters 1 and 2 that will not be covered in class lectures**.

   Tentative Schedule based on the Goodrich et al. textbook
   The schedule is **tentative** and might change anytime.

Chapter	Topic
-	Introduction
4	Algorithm Analysis (4.1 to 4.3)
5 + 15	Recursion (5.1 to 5.6, 15.1 to 15.2)
6	Stacks and Queues (6.1 to 6.2)
3 + 7	Vectors, Lists, Iterators and Sequences (3.1 to 3.4, 7.1 to 7.3)
8	Trees (8.1 to 8.4)
9	Priority Queues and Heaps (9.1 to 9.4)
10	Maps, Dictionaries and Hash Tables (10.1 to 10.3)
11	Search Trees (11.1 to 11.3)
12	Sorting (12.1 to 12.4)
14	Graphs (14.1 to 14.7)

2) Walter Savitch, Absolute Java 6th Edition or later. Addison Wesley. The book is available in 2 formats: Hard Copy: ISBN: 978-0-13-404167-4 & Digital Copy: ISBN: 978-0-13-394783-0

Tentative Schedule based on the Savitch textbook

The schedule is **tentative** and might change anytime.

Chapter	Topic
4, 5 & 6	Review of Classes, Objects & Arrays
7	Inheritance
8	Polymorphism and Abstract Classes
9	Exception Handling
10	File I/0 & Serialization
13	Interfaces & Inner Classes
14	Generics
15 & 16	Linked Data Structures

COURSE STRUCTURE & DETAILS OF THE COURSE COMPONENTS

In addition to the lectures, the course has:

i. A Weekly Laboratory, where it is mandatory to attend. During this laboratory time (over 12 weeks), you will have:
   a. 8 mini-lab projects
      - You must attempt all 8 mini-labs. Out of the 8 mini-labs, two will be marked at random; however heavy marks will be deducted for any missing lab regardless of whether, or not, that lab is marked.
   b. 3 programming exams
      - You must attempt all 3 programming exams.
   c. Final Exam (See details below, in point # iii).

ii. Assignments – there will be 3 to 4 assignments (weight of each assignment depends on the difficulty of the assignment). These assignments play a major role in your learning of the various topics covered in this course. You will need to attempt all assignments. Each of these assignments will include a theoretical component and a programming component. Not all parts of an assignment will be marked. Each assignment will clearly specify which parts will, or will not, be marked. Nonetheless, you must attempt and submit all questions regardless of that. Full details will be specified on each of the assignments.

   For all programming components of your assignments, you need to use Java version 8.0 or later.

   Submission format: All assignment-related submissions must be adequately archived in a ZIP file using your ID(s) and last name(s) as file name. The submission itself must also contain your name(s) and student ID(s). Use your "official" name only – no abbreviations or nick names; capitalize the usual "last" name. Inappropriate submissions will be heavily penalized and only electronic submissions over Moodle will be accepted. Students will have to submit their assignments individually for the theoretical component. For the programming

component, one copy per group is to be submitted (in case groups are allowed). Assignments must be submitted in the right submission folder. Assignments uploaded to an incorrect submission folder will not be marked and result in a zero mark. No resubmissions will be allowed.
Criteria used in evaluation of assignments:

- *Correctness and Testing:* the program should conform to the specification given in the assignment. This includes the proper handling of special cases and error conditions and the providing of correct results. The submitted test cases take into consideration special cases and error conditions.
- *Design:* the program should be constructed from coherent, independent, and decoupled functions. A function should usually access only its own parameters and local variables.
- *Style:* the program should be general-purpose and well-organized.
- *Documentation and Layout:* The documentation should consist of a well annotated program and clearly formatted output. Helpful identifiers and a clear layout are part of documentation. The documentation should include the description of your design and the algorithm implemented.
- *Efficiency:* The program must implement the most appropriate method.
- *Program-User Interface:* The program should be easy to use.

IMPORTANT: For the programming part of each assignment, a demo for about 5 to 10 minutes will take place with the marker. You (or **both** members, if groups are permitted) **must** attend the demo and be able to explain your program to the marker. Different marks may be assigned to teammates based on this demo. The schedule of the demos will be determined and announced by the markers, and students must reserve a time slot for the demo (only one time slot per group). **Now, please read very carefully:**

- **If you fail to demo, a zero mark is assigned regardless of submission.**
- **If you miss a demo at a booked time, for whatever reason, you will be allowed to reschedule a second demo with a penalty of 50%.**
- **Failing to demo at the second appointment will result in zero marks and no more chances will be given under any conditions.**

iii. **Final Exam:** A final exam will take place at the end of the term. This will be an in person written exam, which will examine your understanding of the material covered during the entire term. The exam may include multiple choice questions (with negative marks for wrong answers), questions with detailed answers, short coding, and True/False questions requiring a short justification.

GRADING SCHEME

The table illustrates components of the course and their corresponding weights (%).

Component	**%**
Assignments	10%
8 Mini-Labs (2 marked at random)	15%
3 Programming Exams	45%
Final Exam	30%

Now please read carefully (rather *very* carefully): To pass the course, you must:

1) **Pass the assignments (50% or more)**
2) **Pass at least two of the three programming exams; and**
3) **Pass the final exam (50% or more).**

There are no make-ups/alternates for missed mini-labs, exams, or assignments. Finally, you must notice that there is **no fixed**, a priori relationship between the numerical percentage and the final letter grades for this course.

In the event of extraordinary circumstances beyond the University's control, the content and/or evaluation scheme in this course is subject to change.

PLAGIARISM: The most common offense under the Academic Code of Conducts plagiarism which the Code defines as **"the presentation of the work of another person as one's own or without proper acknowledgement."**

This could be material copied word for word from books, journals, internet sites, professor's course notes, etc. It could be material that is paraphrased but closely resembles the original source. It could be the work of a fellow student, for example, an answer on a quiz, data for a lab report, a paper or assignment completed by another student. It might be a paper purchased through one of the many available sources. Plagiarism does not refer to words alone - it can also refer to copying images, graphs, tables, and ideas. "Presentation" is not limited to written work. It also includes oral presentations, computer assignments and artistic works. Finally, if you translate work of another person into French/English and do not cite source, this is also plagiarism.

In this course, all submitted items that are used to evaluate your performance in this course such as exams, problems or exercises are individual and should not include any code written by someone else. While discussions between students is encouraged, one should never send or show the solution to a problem to another student and one should not use any material that he or she did not write himself/herself. In addition, use of any online resource during all the exams of this course is strictly forbidden. You are also not allowed to make use of any preexisting code in the exam even if it is your own code. If plagiarism is detected where one student copies totally or partially the solution of another student, both students will be equally punished regardless of circumstance.

Please note that plagiarism detection software will be used on all submitted material. The penalty for plagiarism may include removal from the academic program.

Simply: ***Do not copy, paraphrase, or translate anything from anywhere without saying where you obtained it!*** (Source: Academic Integrity Website:
http://provost.concordia.ca/academicintegrity/plagiarism/
http://www.concordia.ca/content/dam/concordia/offices/provost/docs/Academic-CodeConduct-2015.pdf

Please note that we will be using automated plagiarism detection software.

IMPORTANT GUIDELINES

Laptops are **STRICTLY PROHIBITED** in the lab during the lab hours. Additionally, all other communications devices, such as cellular phones, communication watches, and text/video messaging devices, tablets, pads, and similar devices are also **STRICTLY PROHIBITED**. The usage of any of these materials during the lab will result in you being asked to immediately leave the lab, and a report of academic misconduct will be filed with the University immediately.