CSE 205 Object-Oriented Programming and Data Structures

Summer 2020 Session B

Problem solving by programming with an object-oriented programming language. Introduces data structures. Overview of computer science topics.

Prerequisites: CSE110 with C or better.

Credit is allowed for only ACO102 or CSE205 or CST200

1. Staff and Logistics

1.1 Teaching and Support Staff

Instructor: Anique Tahir - Email: artahir@asu.edu - Office Hours: TBD

Note: If you cannot meet me during the office hours, I will try to accomodate any appointments. However, please try to make an appointment 3 days in advance.

Office Hours: The exact office hour scheduling will be published seperately on Canvas. There will be a constant time for office hours every week. You can join the office hours using the following link(at the appropriate time):

https://asu.zoom.us/my/anique

Since there are several students who have private queries, I will sometimes use breakout rooms to answer students questions privately. I have noticed that sometimes when projects and homeworks are due, we have a lot of traffic in the office hours. In order to reduce congestion, I would recommend coming to the office hours prepared. Please make sure that you have read the course content related to your questions before coming using the office hours. I will try to answer non-specific questions first publicly so everyone can benefit from the answers.

Teaching Assistants: TBD

1.2 Communication Other than office hours, you have several options to communicate with the course staff. If you choose to communicate using email, please make sure you use the following subject:

```
[CSE205] actual subject of the message
```

Since I receive a lot of emails, it is very important that you use the subject format otherwise I may not get your message as I plan on using a filter.

Please **avoid** using personal messages on **Canvas** to communicate with me. If you want to get in touch with me, use my email or office hours.

1.2.1 Discussion Boards One of the ways you can communicate with the entire class is by using the discussion boards on Canvas. While I recommend using these for sharing ideas and queries, please do not post any solutions to your assignments or projects. You are also not allowed to post snippets of your solutions. Any academic integrity violation will be reported.

2. Course Information

2.1 Catalogue Description Problem solving by programming with an object-oriented programming language. Introduces data structures. Overview of computer science topics. Prerequisites: CSE110 with C or better. Credit is allowed for only ACO102 or CSE205 or CST200

2.2 Textbook Java for Everyone: Late Objects , Cay S. Horstmann, 2nd Edition, © 2013, Wiley, ISBN-13: 978-1-118-06331-6. This is the **required** textbook for the course officially.

You may use any other Java book for reference as long as you think it covers all of the content. A very good Java reference(which is also free) can be found **here**.

You will also be expected to read the relevant parts of the Java API documentation while writing code. The Java API documentation can be found **here**

2.3 Course Delivery The course will be delivered in the form of regularly scheduled lectures.

2.4 Learning Objectives

1. Software Design
    - Employ proper object oriented design techniques to identify classes and objects and define the relationships among them.
    - Read simple UML (Unified Modeling Language) class diagrams to represent OO designs and implement the design in pseudocode and Java.
2. Concepts of Data Structures
    - Write programs using basic predefined Java data structures such as ArrayList.
    - Write programs to implement data structures: linked lists, queues, stacks, and trees.
    - Identify the appropriate data structures to use in a program to solve a problem.
3. Object Oriented Language Constructs

- Write programs using text files as input and output.
- Design and implement GUI (graphical user interfaces) programs.
- Write a complete Java application using OO concepts such as inheritance, interfaces, polymorphism, and abstract classes.
- Write programs that perform exception handling.

4. Introduction to Algorithmic Complexity Theory
    - Understand the definition of O ("Big Oh") and derive the Big O complexity of an algorithm.
    - Derive and explain the efficiency of sorting algorithms, e.g., selection sort, bubble sort, merge sort, quick sort.
    - Derive and explain the efficiency of searching algorithms, e.g., linear search and binary search.
    - Explain the differences between various complexity classes such as O(1), O(n), O(n2), and O(n lg n).

2.5 Major Topics

1. Object Oriented Software Development
    - Objects, classes, inheritance, abstract classes, interfaces, and polymorphism.
    - GUI-based programming.
2. Introduction to Data Structures
    - Primitive arrays and the Java ArrayList class.
    - Linked lists.
    - Stacks.
    - Queues.
    - Trees.
    - Binary search trees.
3. Introduction to Computer Algorithms
    - Recursion.
    - Searching.
    - Sorting.
    - Algorithmic complexity theory and Big O notation.
4. Input/Output Streams and Exception Processing
    - Errors, exceptions, and exception handling.
    - Text file input/output stream classes.

2.6 Weekly Plan

Week 1

- Intro to Java
- Java Software Development Tools
- Review of 1D and 2D Arrays
- Wrapper Classes

- The ArrayList Class
- Review of Text File Input/Output
- Exceptions and Exception Handling
- Review of Classes and Objects

Week 2

- Read UML class diagrams and convert the diagram into Java classes.
- Identify and implement dependency, aggregation, inheritance, and composition relationships.
- Properly use the public, private, and protected accessiblity modifiers.
- Write Java code to override and overload methods.
- Recognize when inheritance is present among classes in an OOD.
- Design and implement classes using inheritance.

Week 3

- Polymorphism
- Interfaces
- Inner classes
- GUI Programming

Week 4

- GUI Programming (concluded)
- Recursion
- Linear Search
- Binary Search
- Introduction to data structures and algorithms
- Algorithmic time complexity and Big O notation

Week 5

- Selection Sort
- Insertion Sort
- Merge Sort
- Quick Sort
- Linked Lists
- Stacks
- Queues

Week 6

- Trees
- Binary Search Trees