**San José State University**

# Computer Science Department CS151, Object Oriented Design and Programming, 07, Spring 2020

# Homework #5

Objective:

This homework’s objective is to review and understand the units on exception handling and IO programming and ways to implement them in Java. It also includes exercises on deep copy.

Details:

Exercise 1:

Implement a program that takes command line input from a user and outputs it to a file named **command\_line\_input.txt**. As long as the user keeps entering input it should keep being appended to the end of this file. When the user presses “@q” (without quotes) the command line input should end and the file should be closed and saved. Use your favorite IO programming libraries to achieve this result. Make sure to implement all necessary exception handling to account for invalid use cases. Save your program to a file named **CreateFile.java**.

Exercise 2:

Implement a program that reads in a text file “quote.txt”, included with this homework assignment, and outputs its contents to the command line screen. Use your favorite IO programming libraries to achieve this result. Make sure to implement all necessary exception handling to account for invalid use cases. Save your program to a file named **ReadFileInput.java**.

Exercise 3:

Implement a program that takes command line input from a user, expecting an integer value, convert it to hexadecimal format and output the converted value back to command line screen. Make sure to implement all necessary exception handling to account for invalid use cases. You can use any built-in Java libraries to implement the conversion. Save your program to a file named **IntToHex.java**.

Exercise 4:

Define and implement class **Course**. This class should contain the following fields: course name, course description, department, time the course starts, weekday the course is held on (for simplicity, let us assume the course only meets once a week). This class should contain getters and setters for all its attributes. This class also needs at least one constructor. Save this class and its definition into a file named **Course.java**.

Define and implement class **Student**. This class should contain the following fields: first name, last name, age, gpa, major, department. Age should be an integer value. GPA should be a floating point value. This class should also contain 4 different fields of type Course. It is up to you which details (name, description, etc.) each of those 4 courses has, as long as the student contains 4 different attributes of type Course. You might want to name the attributes by the name of the course. For example, variable name “cs151” refers to an object of type Course with the following details: CS151, Object Oriented Design and Programming, CS, 6:00pm, Tue. You might declare it as:

*private Course cs151;*

Important note: Normally, we would not be declaring a separate attribute variable for each course. It is not considered a good programming practice to hard-code things like this in your application. Normally, we would implement this using Java lists. However, we have not covered those yet so we are using this awkward workaround to add courses to a student.

Class Student should contain getters and setters for all its attributes. Implement a method called *printInfo()*, which will print the values of the Student instance. Save this class and its definition into a file named **Student.java**.

Define and implement class **StudentTest**.This class should implement *main()* method. In the body of the *main()* method you should create an instance of Student with the following information: John Smith, 20 year old, 3.6 gpa, Computer Science major, School of Computer Science department. This student will also be taking 4 courses, which you specified in the definition of the Student class. Create another instance of Student class, which should be a clone of the first student. Use *printInfo()* method call to print the values of both instances of Student. Save this class and its definition into a file named **StudentTest.java**.

Exercise 5:

Define and implement class **Employee**. This class should contain the following fields: first name, last name, employee id, hourly pay. This class should contain getters and setters for all its attributes. This class also needs at least one constructor. Implement method *computePay()* in this class. This method should accept an integer number of hours the employee worked as an input argument and return a floating point value indicating the pay the employee earned. If the number of hours is invalid (e.g. a negative number) this method should throw *NumberFormatException*. If the number of hours is over 40 then this method should throw a custom exception named *TooManyHoursWorkedException*. Remember that you will have to define your own exception class for this. Save this class and its definition into a file named **Employee.java**.

Define and implement class **EmployeeTest**.This class should implement *main()* method. In the body of the *main()* method you should create an instance of Employee with the following information: John Smith, id = 101, hourly pay = $35/hr. Use *computePay()* method call to print to command line what this employee earned for the following number of hours:

40

23

1

0

-5

45

If an exception has been thrown by *computePay()* make sure to catch it and print the appropriate error message to command line. Save this class and its definition into a file named **EmployeeTest.java**.

Submission:

In your class repo create a directory called “Assignment5” and add all the files created for this homework assignment to that directory.

This homework assignment is assigned on 02/27/2020 and is due on 03/10/2020 before 11:59pm. Email your assignment submission to me at both [Yulia.Newton@sjsu.edu](mailto:Yulia.Newton@sjsu.edu) and [yulia.newton@gmail.com](mailto:yulia.newton@gmail.com), as well as the grader at [akshay.kajale@sjsu.edu](mailto:akshay.kajale@sjsu.edu). The subject of the email should say “CS151 Assignment 5”. Add your name as it appears on the class roster and the URL to your git repo in the body of an email.

Grading:

Your code must compile and execute successfully in order to get full credit for this assignment. For each exercise, I will compile and execute xxxTestxxx.java file listed in that exercise description.

* Program with no compile errors (5 pts)
* Program executes (5 pts)
* Program outputs what is required by the exercise (5 pts)
* A total of 75 points are possible for this homework assignment.