Students may use their text book, any hand written notes, and any digital notes or programs written by the student (You may use your flash drive). **DO NOT USE THE INTERNET** 8:00am – 11:00am

**SHORT ANSWER SECTION**

1. (10 Points) Explain the difference between Procedural Programming and Object Orientated Programming.

Procedural programming is a programming paradigm, derived from structured programming, based upon the concept of the procedure call.

Object-oriented programming (OOP) is a programming language model organized around "objects" rather than "actions" and data rather than logic.

2. (10 points) Describe the difference between Internal Documentation and External Documentation when diagramming / outlining the way source code and programs operate. Give 1 example of each.

Internal Documentation- if the notes on how and why various parts of code operate is included within the source code as comments. It is often combined with meaningful variable names with the intention of providing potential future programmers a means of understanding the workings of the code. Internal documentation would be comments and remarks made by the programmer in the form of line comments and boiler plates.

Example: Java Doc Tags, comments

External Documentation- External documentation would be things like flow charts, UML diagrams, requirements documents, design documents etc.

Example: UML diagrams

3. (10 Points) Describe the difference between an Error and an Exception in Java.

Errors should not be caught or handled (except in the rarest of cases).

Exceptions are the bread and butter of exception handling

4. (10 Points) – a) Define Encapsulation and give one example

Encapsulation is the packing of data and functions into a single component. The features of encapsulation are supported using classes in most object-oriented programming languages, although other alternatives also exist.

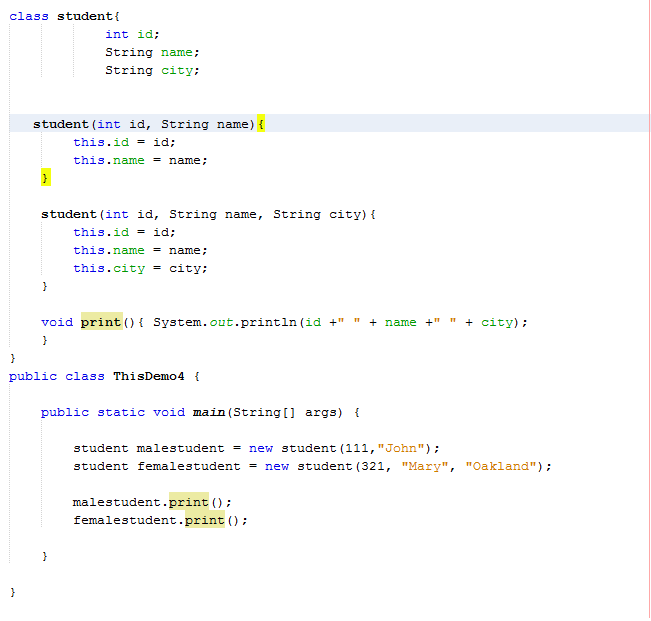
Example: Object

b) Define Polymorphism and give one example

The word 'polymorphism' literally means 'a state of having many shapes' or 'the capacity to take on different forms'. When applied to object oriented programming languages like Java, it describes a language's ability to process objects of various types and classes through a single, uniform interface

Example: interface

**DIAGRAM PROGRAMMER CREATED CLASS:** (10 Points – 2 Points Each)



References to the current object

Constructor

Object

Parameters

Class variable

**SHORT PROGRAMMING TASKS:**

(15 points) TASK 1: Try/Catch

Assignment #12 Task 2 – Add Try / Catch

Write a program that computes a single filer’s income tax burden.

|  |  |
| --- | --- |
| **TAX RATE** | **Single Filers Income** |
| 10% | Up to $6000 |
| 15% | $6,001 - $27,950 |
| 27% | $27,951 - $67,700 |
| 30% | $67,701 - $141,250 |
| 35% | $141,251 - $307,050 |
| 38.6% | $307, 051 or more |

The user should be able input her income using **new** Scanner (System.in), input method and then be returned the amount of tax owed. (Incorporate Try/Catch to deal with any unexpected user input)

All source code for solving the problem and handling user input should be created in a “programmer created class.”

Use **return** for retrieving all values from calculations or **if/else** statements etc.

**main** will be used to operate the program.

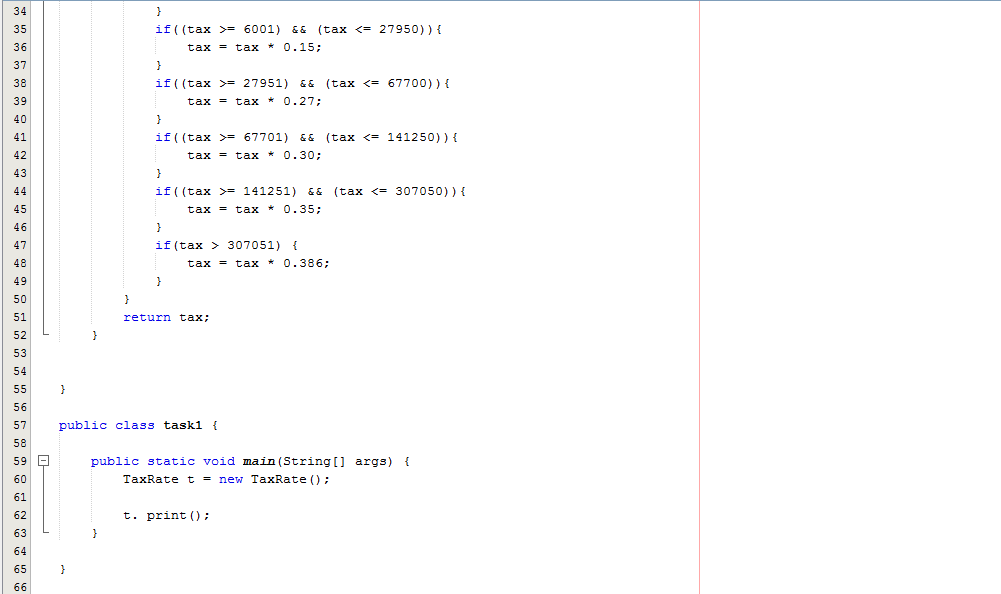
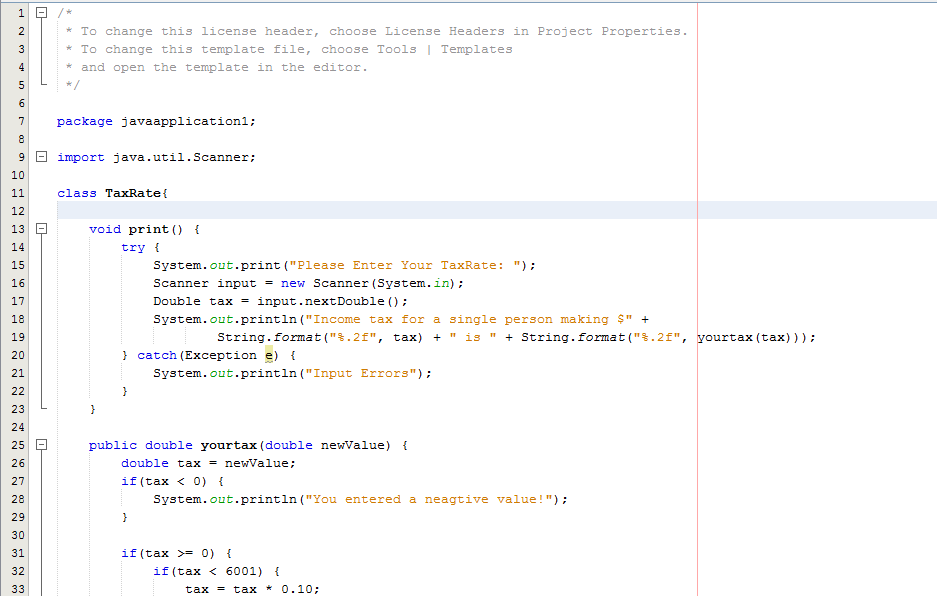
Output should have proper formatting for dollars, 2 decimal places.

**Sample Output- //**Output should have proper formatting for dollars, 2 decimal places

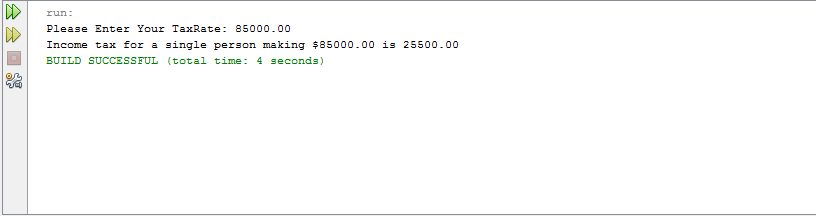
Income tax for a single person making $85000.00 is $25500.00

Income tax for a single person making $9800.00 is $1470.00

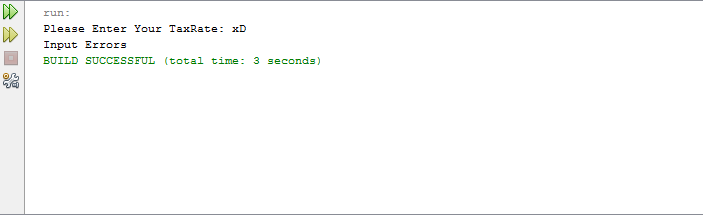
Attach snipping photos of source code and 2 runs of the program.



1 with expected user input



1 with unexpected user input



(40 points) TASK 2: Object Orientated Programming

Take your code from Assignmnet#4 Task 1 and Task 2 change the source code to reflect an object orientated model for programming.

Use Constructors, Parameters, and arguments as needed.

Write a program that has 2 methods.

1 method will convert Seconds into minutes.

1 method will convert seconds to hours, minutes, and seconds

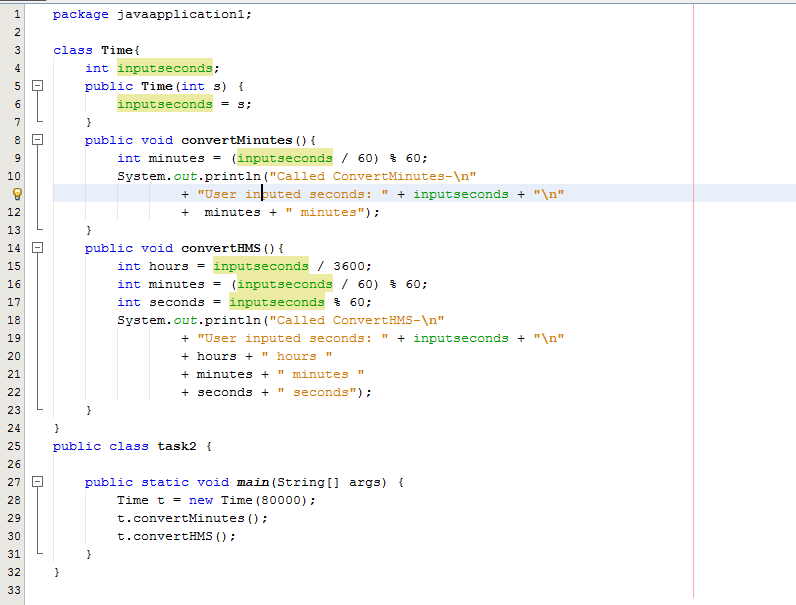
Use the modulus arithmetic symbol for part of the calculation/conversion if you like.

**Output=** Show number of seconds, conversion to minutes, and remaining seconds.

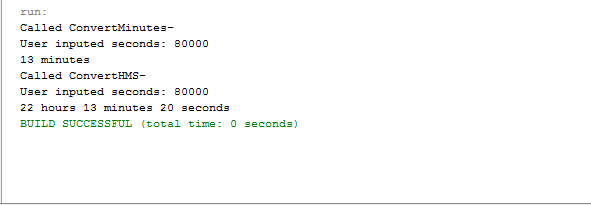
or

Show number of seconds, converted to hours, converted to minutes, and remaining seconds.

Attach Source Code photo



Attach output of both methods being run.



(15 points) TASK 3:

Use a **recursion** to **print all the numbers from an integer (n) to 100.**

The program should be able to start from any integer 0 – 100, then print the numbers between **(n)** and 100.

Example Output: Source Code ( 2 Snipping Photos: Source Code + expected output)

//Referencing recursion method created in the “program created class.”

public static void main(String[] args) {

numbers sample = new numbers();

sample.numbers(**89**);

}

**Print/Output:**

89

90

91

92

93

94

95

96

97

98

99

100

