Assignments

The assignments are useful for practicing the Java lectures topics in developement and implementation for individual and team tasks.

Assignment 01.

Deadline - YYYY/MM/dd HH:mm: Current Year/03/11 23:50 GMT+2
Problem description: "Computing Taxes Problem - The US federal personal income tax is calculated based on the filing status and taxable income. There are four filing statuses: single filers, married filing jointly, married filing separately, and head of household. The tax rates for 2009 (20xx?) are shown below:"

Marginal Tax Rate	Single	Married Filing Jointly or Qualified Widow(er)	Married Filing Separately	Head of Household
10%	$0 – $8,350	$0 – $16,700	$0 – $8,350	$0 – $11,950
15%	$8,351– $33,950	$16,701 – $67,900	$8,351 – $33,950	$11,951 – $45,500
25%	$33,951 – $82,250	$67,901 – $137,050	$33,951 – $68,525	$45,501 – $117,450
28%	$82,251 – $171,550	$137,051 – $208,850	$68,525 – $104,425	$117,451 – $190,200
33%	$171,551 – $372,950	$208,851 – $372,950	$104,426 – $186,475	$190,201 - $372,950
35%	$372,951+	$372,951+	$186,476+	$372,951+

Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. The ZIP filename rule will have JavaSE_A01_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Implement using *if-else* and *switch* statements - please see Lecture 01 for start-up.

Assignment 02.

Deadline - YYYY/MM/dd HH:mm: Current Year/03/11 23:50 GMT+2
Problem description: "Write a simple Java program that uses nested for loops to print a multiplication table – nested loops."
Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. The ZIP filename rule will have JavaSE_A02_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Implement using *for* and *while* statements - please see Lecture 01 for start-up.

Assignment 03.

Deadline - YYYY/MM/dd HH:mm: Current Year/03/11 23:50 GMT+2
Problem description: "Declare one hundred numbers in a Java program, compute their average, and find out how many numbers are above the average."
Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. The ZIP filename rule will have JavaSE_A03_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Implement using Java methods, arrays, *for* and *while* statements - please see Lecture 01 for start-up.

Assignment 04.

Deadline - YYYY/MM/dd HH:mm: Current Year/03/11 23:50 GMT+2
Problem description: "Develop in Java, 'Matrix' class and write a program that uses Matrix objects."
Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. The ZIP filename rule will have JavaSE_A04_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Implement OOP - Object Oriented Progrmming paradigms using Java class, interface, methods, arrays, *for* and *while* statements - please see Lecture 01 for start-up.

Assignment 05.

Deadline - YYYY/MM/dd HH:mm: Current Year/03/15 23:50 GMT+2
Problem description: "Develop in Java, Eclipse Java projects in BOTH: non-OOP and OOP approaches (but without the JCL - Java Collection Framework) for the following problems - also include the pics with the logic flow:

5.1. A person is tracking the gas consume of the his/her car in terms of liters and money invest per litre in every day. Find the overall consume and the average consume in both money and liters.

5.2. A software developer is staying a certain number of the minutes in front of the screen per day and he/she is counting them for 3 weeks. Find out the biggest and the smallest amount in minutes in front of the display and in which days happened.

5.3. In a students class we have m students and n assignments mark for each student. We have to find the average mark for each student and the average mark of the entire class. Extend the example to calculate the average per class and total for the entire year with minimum 5 groups/classes.

5.4. In a rent-car company, there are n cars and they are monitored m days in terms of the km per day. Find out the average per m days for each car. In addition find out the max and min of the km per day for each car.

5.5. In a parking lot with OCR cameras and access cards there are n places for the cars. In each parking place, there are parking different cars which are staying the different number of minutes. Each parking place is monitored for 5 working days. Find out the max and min time spent by the car for each place."

Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. Also the ZIP bundle must contain the JPEG small pics of the data flows exported from http://draw.io or MS Visio. The ZIP filename rule will have JavaSE_A05_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Implement business logic flow and OOP - Object Oriented Progrmming paradigms using Java class, interface, methods, arrays, *for* and *while* statements - please see Lecture 01 for start-up.

Assignment 06.

Deadline - YYYY/MM/dd HH:mm: Current Year/03/22 23:50 GMT+2
Problem description: Implement in Java Robocode (robocode-1.9.0.0), the following tasks:
6.1 Using just the ahead and turnRight/turnLeft methods create a robot that travels in a complete square once, beginning from its starting position. Make the robot travel 150 units for each side of the square.
6.2 Using a for loop create a robot that travels in a complete square 10 times.
6.3 Adapt the code so that it uses a loop that repeats four times to control the forward movement and turns.
6.4 Using a while loop create a robot that travels in a square forever (or until the round ends anyway).
6.5 Alter your robot so that it counts the number of squares it has travelled and writes this to the console.
6.6 Alter the above robot so that it incorporates an if statement such that it travels first clockwise and then anti-clockwise (hint: x % 2 == 0 for even numbers of x, and turnRight and turnLeft can accept negative degrees). Also have the robot print out whether it's currently travelling clockwise or anti-clockwise.
6.7 Create a new method called moveInSquare. The method should return void, and accept a single parameter, of type int called lengthOfSide. Alter the code from exercise 5.5 above so that the statements that describe how to move in a square are now in this method. Alter the run method so this method is called to make the robot move.
6.8 Experiment with adding additional attributes that control your robots behaviour:
a. a boolean attribute called aggressive can indicate whether the gun should fire at each corner
b. a boolean attribute called scanForRobots can control whether the robot rotates its radar at each corner
c. other attributes can control the size of the square, the number of degrees the gun and radar are turned, the direction of their turn, etc.
Also, create into the robot code with an 'onScannedRobot' method. Consult the documentation for the parameters and return types of the method. Implement the method to do something useful when it receives the event, e.g:
a) Write out the name of the robot you've seen, its current energy, and its distance
b) Fire on it!
Note: this extends the previous exercise so that its got all the other behaviour. It just adds this one extra event handler.

Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. The ZIP filename rule will have JavaSE_A06_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Implement *if-else, switch, for* and *while* statements inside a AI Game framework already developed by a third-party entity - please see Lecture 01 and 02 for start-up.
Hints: Download Robocode 1.9.0.0 from http://sourceforge.net/projects/robocode/files/latest/download?source=files. Be sure that you are testing the "game" using a ***network connection available***. Please take a look into the following sections for hints.
- Unzip/extract the kit in /home/stud/javase/lectures
- Go to into the extracte directory and change permision for *.sh files from the directory of robocode (chmod 755 *.sh).
- Run (double click) on 'robocode.sh' file
- Click in main menu on 'Robot->Source editor' and create the source code of the robot ('File->New->Robot' - Package: eu.ase.jrobot) in order to implement the exercices.
- Implement the requirements in 'public void run()' method from the your class robot, which extends 'robocode.Robot' class.
- Compile the created source code ('Compiler->Compile').
- Start new battle from the main window of the Robocode ('Battle->New')

Assignment 07.

Deadline - YYYY/MM/dd HH:mm: Current Year/04/15 23:50 GMT+2
Problem description: Implement the following tasks using **Java Multithreading API**:
7.1 Parallel addition for two large vectors - benchmark with running time on various configurations (e.g. millis consumed in 8 threads on 2 cores, 4 threads on 4 cores, etc.)
7.2 Parallel dot product between two large vectors - benchmark
7.3 Parallel addition for two large matrixes - the two input matrix are read from file stream (text file) and the output is written also in file stream (text file) + benchmark
7.4 Parallel addition of the items from each row of a matrix object - benchmark
Upload and packiging: "The ZIP file has the source code files + *.sh/*.bat file for compiling and running the sample. The ZIP filename rule will have JavaSE_A07_SURNAME_FirstName1_FirstName2.zip". The upload is available through SAKAI, please see E-Learning Solution section from this page.
Objective: Use Java API multi-threading for concurrency and parallelism.