# CMPE 221_223_242
# Fall 2022
# Programming Homework 3

This assignment is due by 23:59 on Sunday, 18 Dec 2022.

You are welcome to ask your HW related questions. You should use only one of these options:

1. Moodle Homework **Question Forum**: HW Question-and-Answer (Q&A) Forum on Moodle is always available. Use the "Forum" link at the course Moodle page.
2. Homework **RECITATION HOURS**: There will be two Q&A RECITATION HOURs on the following days:

   CMPE223-HW3-OfficeHour1: 11 Dec, 18:00-20:00 PM, Zoom ID: 986 024 5410

   CMPE223-HW3-OfficeHour2: 15 Dec, 18:00-20:00 PM, Zoom ID: 986 024 5410

Note: Please make sure that you have read the HW document well before participating. However, no HW related questions will be accepted except from the above options.

Note 2: This HW must be done in groups of 2 and only one of the group members should submit the documents to the system. In assignments submitted by two people from the same group, points are deducted from both students.

## PROGRAMMING TASK

### Q1:

In this question, you are expected to implement a simple Inventory Tracking System **using one of the search trees**. You must use your own implementation of tree data structure by taking inspiration from your textbook or web. You are not allowed to use any external library or .jar file.

In your implementation, for each product, you must keep ID, name and piece in tree data structure.

You should implement three basic operations for your systems, which are:

1. Create product
2. Is the product available
3. Quit

**Create product:** In this function you should create product and store it in your search tree data structure.

**Is the product available:** This function, if the product is in stock, prints to the console how many are left. If the product is not found, the message " The product is out of stock!!!" must be displayed.

**Quit:** This function must stop your program.

**Example Input/Output:**

Below is an example of input/output. Do not forget to test your program with different test codes as well. You can check VPL in the LMS page for more examples.

Inputs must be read from an input.txt file, not from the console but your output must appear in the console. You can also find the input.txt file that gives the following output in LMS.

```
--------------- WELCOME TO ITS ---------------
Create Product:

                        ID: 23
                        Name: sweatshirt
                        Piece: 1
Create Product:

                        ID: 56
                        Name: skirt
                        Piece: 2
Create Product:

                        ID: 95
                        Name: hat
                        Piece: 6
There are 1 products

There are 2 products

The product is out of stock!!!

Thank you for using ITS, Good Bye!
```

## Q2:

In this question, a chauffeur-driven rental company (CDRC) asks you to write a management system code in java **using binary search trees** for their captain's salary increase. You must use your own implementation of binary search trees by taking inspiration from your textbook or web. You are not allowed to use any external library or .jar file.

These chauffeurs should be kept in a captain database. In this captain database, for each captain, you should keep ID, name, available, and rating star.

In your implementation, you must keep the captain entries in a binary search tree by the ID of the captain. No two captains have the same ID!
Your system will have the following functionalities; the details of these functionalities are given below (see the end of this document for sample inputs/outputs):

1. Add a captain
2. Delete the captain
3. Print a captain's information
4. Print all captains' information
5. Rent a car
6. Finish the ride
7. Quit

**Add a captain**: This function generates a BST node for each captain in the tree. The function is followed by two values: the captain ID and the captain's name. The default value for the rating stars will be "0" and the default value of available is "true".

**Delete the captain**: This function has one value that represents the captain's ID. The command will search for this captain in the tree then remove his/her node from the BST, and it will output the "The captain left CDRC" message. If the specified captain is not found, it will output the "Couldn't find any captain with ID number " message.

**Print a captain's information:** This function has one value which is the captain ID. It will output the name and the rating stars for the specified captain. If the specified captain is not found, it will output the "Couldn't find any captain with ID number " message.

**Print all captain's information:** This function prints all captain's IDs, names, available, and rating stars.

**Rent a car**: This function has one value which is the captain ID. The command will book a ride with the specified captain by changing the available to "false". If the available for the specified captain is already "false", then output the "The captain is not available. He is on another ride!" message. If the specified captain is not found, it will output the "Couldn't find any captain with ID number " message.

**Finish the ride:** This function has two value which is the captain ID, and the rider satisfaction (0 or 1). This command will make the specified captain available again by changing the available to "true". The rating stars will affect the captain rating stars. The rating stars will increase by one if the rider is satisfied and decrease by one otherwise. (Note: the rating stars is a value between 0 and 5 only). If the available for the specified captain is already "true", then output the "The captain is not in a ride!” message. If the specified captain is not found, it will output the “Couldn’t find any captain with ID number ” message.

**Quit:** This function must stop your program.

**Example Input/Output:**

Below is an example of input/output. Do not forget to test your program with different test codes as well. You can check VPL in the LMS page for more examples.

Inputs must be read from an input.txt file, not from the console but your output must appear in the console. You can also find the input.txt file that gives the following output in LMS.

```
--------------- WELCOME TO CDRC SYSTEM ---------------
Add Captain: Add a new captain record in the System

                       ID: 801
                       Name: Burak
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Add Captain: Add a new captain record in the System

                       ID: 802
                       Name: Ahmet
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Add Captain: Add a new captain record in the System

                       ID: 803
                       Name: Ali
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Add Captain: Add a new captain record in the System

                       ID: 804
                       Name: Can
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Add Captain: Add a new captain record in the System

                       ID: 805
                       Name: Yasir
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Add Captain: Add a new captain record in the System

                       ID: 806
                       Name: Pelin
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Add Captain: Add a new captain record in the System

                       ID: 807
                       Name: Adem
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
IsAvailable: Reserve a new Ride with captain 801

----------------------------------------------------------------
IsAvailable: Reserve a new Ride with captain 802

----------------------------------------------------------------
IsAvailable: Reserve a new Ride with captain 803

----------------------------------------------------------------
IsAvailable: Couldn't find any captain with ID number 814

----------------------------------------------------------------
IsAvailable: The captain Ali is not available. He is on another ride!

----------------------------------------------------------------
Display Captain: Couldn't find any captain with ID number 820

----------------------------------------------------------------
Display Captain:
                       ID: 802
                       Name: Ahmet
                       Available: False
                       Rating star: 0

----------------------------------------------------------------
Finish: Finish ride with captain 801

                       ID: 801
                       Name: Burak
                       Available: True
                       Rating star: 0
----------------------------------------------------------------
Finish: Finish ride with captain 802

                       ID: 802
                       Name: Ahmet
                       Available: True
                       Rating star: 1
----------------------------------------------------------------
Finish: The captain Adem is not in a ride
----------------------------------------------------------------
Finish: Couldn't find any captain with ID number 811
----------------------------------------------------------------
Delete Captain:The captain Can left CCR
----------------------------------------------------------------
Delete Captain: Couldn't find any captain with ID number 814
----------------------------------------------------------------
----------ALL CAPTAINS----------
--CAPTAIN:

                       ID: 801
                       Name: Burak
                       Available: True
                       Rating star: 0

--CAPTAIN:

                       ID: 802
                       Name: Ahmet
                       Available: True
                       Rating star: 1

--CAPTAIN:

                       ID: 803
                       Name: Ali
```

```
                        Available: False
                        Rating star: 0

--CAPTAIN:

                        ID: 805
                        Name: Yasir
                        Available: True
                        Rating star: 0

--CAPTAIN:

                        ID: 806
                        Name: Pelin
                        Available: True
                        Rating star: 0

--CAPTAIN:

                        ID: 807
                        Name: Adem
                        Available: True
                        Rating star: 0

Thank you for using CDRC, Good Bye!
```

## WHAT TO HAND IN

A zip file for both parts containing:

- The Java sources for your program.
- The Java sources should be WELL DOCUMENTED as comments, as part of your grade will be based on the level of your comments.
- You should test your Java source files on (if) available Moodle VPL environment to ensure your code solution's correctness before submitting. VPL simply tests your program's output by checking against given sample input. You should pass that task's VPL test case successfully.
- A **maximum-3 pages** PDF report document that explains your own answers for programming task in a clearly readable PA report format (refer to **PA REPORT FORMAT** section). Include this report in your zip file and upload it to the PA Report Submission screen in LMS.
- For given task, only code or report submission will not be graded. In other words, you should submit both correct code solution and its related report for the task in order to be graded.

A programming assignment report is a self-description of a programming assignment and your solution. The report must not be hand-written. You may use a word processor or the on-line editor of your choice and prepare as a PDF document. The report must be grammatically correct and use complete English sentences. Each report should include the following sections, in the order given:

**Information (%2.5)**: This section includes your ID, name, section, assignment number information properly.

**Problem Statement and Code Design (%15)**: Include a brief summary of the problem and/or your sub-tasks to be completed in this assignment. You should show your modular design rationale by creating a structure chart that indicates your top-down, stepwise refinement of the problem solution. You may create the structure chart using available graphical tools like MS PowerPoint, SmartDraw etc.

**Implementation, Functionality (%20)**: Since you have modular source code, you should describe each sub-module (program) in this section. Each sub-module should include names and types of any input/output parameters as well as the pseudocode algorithm that used for completing its task. By this way, you give meaning to each chart boxes from the previous section.

**Testing (%7.5)**: You should provide a tester class that is able to identify key test points of your program. This class should be able to generate additional (apart from the given sample input/output) test data for the purpose of being clear on what aspects of the solution are being tested with each set. This section should also include a description of any program *bugs* that is, tests which has incorrect results. You should write these to describe your tests, summarize your results, and argue that they cover all types of program behavior.

**Final Assessments (%5)**: In this final section, you should briefly answer the following questions:

- What were the trouble points in completing this assignment?
- Which parts were the most challenging for you?
- What did you like about the assignment? What did you learn from it?

## GRADING:

- Codes ( %50, Q1:25, Q2:25)
  - Available test cases evaluation on VPL: %15, (Q1:7.5, Q2:7.5)
  - Hidden test cases evaluation: %15, (Q1:7.5, Q2:7.5)
  - Approach to the problem: %20, (Q1:10, Q2:10)
- Report ( %50)
  - Information: %2.5
  - Problem Statement and Code design: %15
  - Implementation, Functionality: %20
  - Testing: %7.5
  - Final Assessments: %5

Submitting report without codes will not be evaluated!!

## IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:59 on Sunday, Dec 18th.

2. You should upload your homework to Moodle before the deadline. No hardcopy submission is needed. You should upload files and any additional files if you wrote additional classes in your solution as a single archive file (e.g., zip, rar).

3. The standard rules about late homework submissions apply (20 points will be deducted for each late day). Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. You ARE NOT ALLOWED to modify the given method names. However, if necessary, you may define additional data members and member functions.

5. Your classes' name MUST BE as shown in the homework description.

6. The submissions that do not obey these rules will not be graded.

7. To increase the efficiency of the grading process as well as the readability of your code, you have to follow the following instructions about the format and general layout of your program.

8. Do not forget to write down your id, name, section, assignment number or any other information relevant to your program in the beginning of your Java files. Example:

```
//----------------------------------------------------------
// Title: Scheduler tester class
// Author: Name/Surname
// ID: 2100000000
// Section: 1
// Assignment: 1
// Description: This class tests the …
//----------------------------------------------------------
```

9. Since your codes will be checked without your observation, you should report everything about your implementation. Add detailed comments to your classes, functions, declarations etc. Make sure that you explain each function in the beginning of your function structure. Example:

```
void setVariable(char varName, int varValue)
//----------------------------------------------------------
// Summary: Assigns a value to the variable whose
// name is given.
// Precondition: varName is a char and varValue is an
// integer
// Postcondition: The value of the variable is set.
//---------------------------------------------------------
{
      // Body of the function
}
```

10. Indentation, indentation, indentation...

11. This homework will be graded by your TA, Elif ÜNAL. Thus, you may ask her your homework related questions through HW forum on Moodle course page.