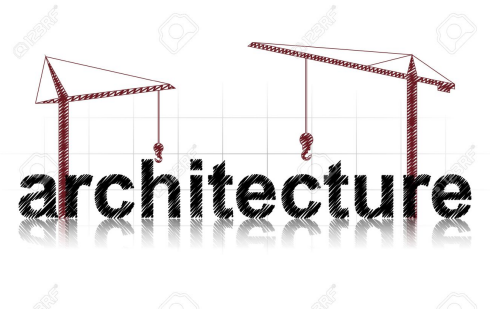


Audition for KUArch

Contents

1 Introduction

1.1 Submission

Submit a folder that contains **only** your Java source file (*.java) at BlackBoard.

- Please write your name into the Java source file where it is asked for.

- You should use comments to explain the purpose of the methods and any specific line of code that needs explanation. Please note that code without comments will be penalized.

1.2 Academic Honesty

Koç University's *Statement on Academic Honesty* holds for all homework given in this course. Failing to comply with the statement will be penalized accordingly. If you are unsure whether your action violates the code of conduct, please consult with your instructor.

1.3 Aim of The Project

This homework is designed for you to feel comfortable creating and using methods. In some cases, you have to implement the obligated methods while you need to implement your own, in others.

2 Plan

2.1 Overview

In this project, you are expected to implement a program that simulates a architecture design audition. KUArch Audition aims the select and eliminate candidates for the architecture design competition to find the best architect in Koç University. Let us go deeper into it:

In the audition part, contestants are tested by their **design knowledge**, their **design experience**, **the years of architecture education** they received, **the number of different types of projects** they worked on and **number of awards** they have received. To spice up the audition part, we flip a coin for every contestant to decide whether they are given **recycle building material** to boost the acceptance of their design as environmental friendly. According to these factors, contestants gain individual total points which determine the triple passing to the finale.

2.2 Variables

KUArch is an audition for non-professional architects. Therefore, top architects and, obviously, people who cannot design are not welcome. So, the qualities of the contestants are required to be in some restricted intervals. (The boundaries are also included.)

Knowledge:

Design Education: An architecture education takes **4-6 years** of degree and it is a must to have for attending KUArchitect.

Design Experience: The first step to great building design is great design experience. KUArch demands KUArchitects to have experience between **integers 2 and 10**.

Experience:

Number of Projects: No one would want to work with a designer who never worked on real project before. KUArch demands KUArchitects to have worked on projects between the **integer values 4 to 15**.

Project Types: By designing different types of buildings, you also gain expertise. Knowing **2 to 5 different types** of projects makes you experienced designer.

Award: Every KUArchitect is required to have won **1 to 5 awards** to enter the KuArch.

int totalPointOfKUArchitect1, totalPointOfKUArchitect2, totalPointOfKUArchitect3, totalPointOfKUArchitect4: Defined as integer variables to keep total points of the four contestant in the first part.

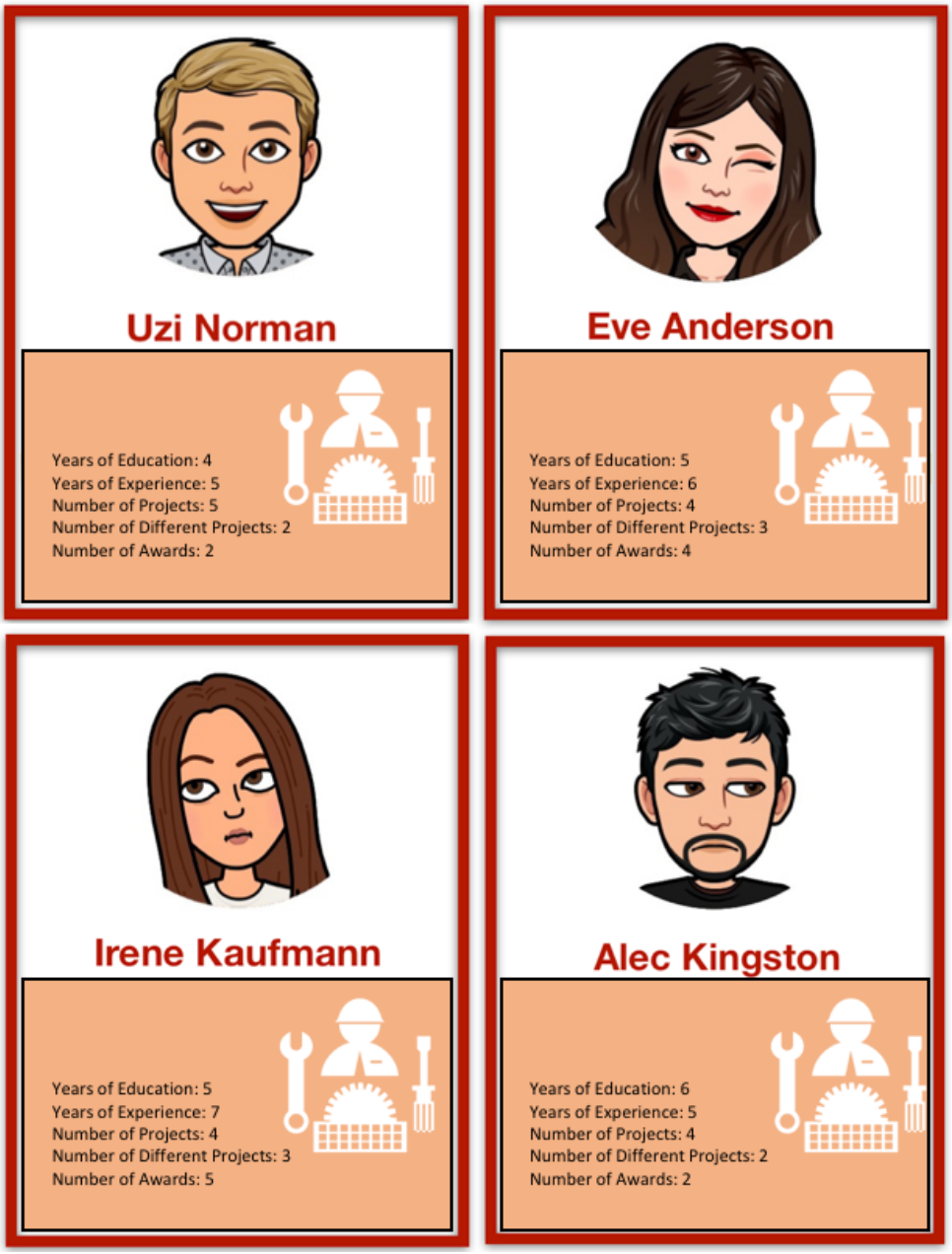
int firstID, secondID, thirdID: Defined as integer variables to keep the digit numbers of the first three architects passing to the finale.

int firstPoint, secondPoint, thirdPoint: Defined as integer variables to keep total points of the KUArchitect with first, second and third maximum points of Part 1.

int currentKUArchitectID: Declared to store the ID number of the current architect under consideration. This is initialized to 1 to indicate the ID of the KUArchitect 1.
(**Hint:** Think of it as a counter.)

2.3 Contestants

Today, we are seeking to find the architect star of Koç University out of four people. Let's meet the team!



2.4 Constants

In the code given to you, we have created 1 constant value for you to work with. Feel free to create more constant variables if needed. However, note that you SHOULD NOT create constants which replace the ones defined below.

CONTESTANT_NUM: Used to store the number of contestants in the game. It is set to 4.

3 Code

We have provided a sample code for you to work with. It consists of the following parts and you are asked to fill in the missing sections in the code.

3.1 Part I

This is the audition part. In this part, you are required to calculate total points and compare these points to find the first three contestants. **We have created the signature of 9 methods for you to work with. You are asked to implement the details of these method using the signature details provided below.** (Along with the methods below, you may implement any method you want.)

3.1.1 Methods

1) double askKnowledge(): Prompts user to enter two integer values for number of years of experience and the number of years of education and reads the entries **until a valid input is given.** Then, it calculates the knowledge point of the KUArchitect by making a call to calculateKnowledge method and returns the result.

2) double calculateKnowledge(int yExp, int yEduc): Takes years of experience and years of knowledge. It calculates and returns the knowledge point using the following formula:

$$KnowledgePoint = (\frac{yEduc + yExp}{5})^{(yExp-2)} + fibonacci(yEduc) \quad (1)$$

Example:

$$KnowledgePoint = (\frac{4+6}{5})^4 + fibonacci(4) \quad (2)$$

$$= (2)^4 + 3 \quad (3)$$

$$= 19 \quad (4)$$

(**Hint 1:** Implementing a fibonacci method which gives the input's element of fibonacci series may be useful. Fibonacci series goes like this: 1 1 2 3 5 8 13 21... Ex: fibonacci(1)=1, fibonacci(2)=1; fibonacci(7)=13)

(**Hint 2:** It is recommended to use **Math.pow(double,double)** method.)

3) int askExperience(): Prompts user to enter two integer values for the number of projects and number of different building projects worked on and reads the entries **until a valid input is given.** Then, it calculates the experience point by using calculateExperience method and returns the answer.

4) int calculateExperience(int numProjects, int numDifProjects): Takes number of projects and number of different projects and returns the experience point using the following formula:

$$ExperiencePoint = numProjects \times numDifProjects! \quad (5)$$

(**Hint:** Implementing a factorial method will be needed.)

5) int askAwards(): Prompts the number of KUArchitects's specials from the user and reads the entry **until a valid input is given** and returns it.

6) boolean hasRecycledMaterial(): Determines whether a KUArchitect is given recycled building materials or not randomly. KUArchitect who is given recycled building material will have more advantage in the competition since their design will be appreciated more. The probability to gain a recycle material is 50%.

(**Hint:** You may want to use RandomGenerator.)

7) int pointCalculator(double knowledge, int experience, int awards, boolean hasRecycledMaterial): Calculates the total point of a KUArchitect using this partial function formula:

$$TotalPoint = \begin{cases} \sqrt{knowledge \times experience} + 1.3 \times knowledge, & hasRecycledMaterials = true \\ \sqrt{knowledge \times experience} + 0.9 \times knowledge, & hasRecycledMaterials = false \end{cases}$$

The total point calculated by the formula must be ROUNDED to the closest integer before it is returned. If the decimal part is greater than or equal to 0.5, then it should be rounded above and conversely if it is smaller than 0.5, it should be rounded below. (Ex: rounded(0.7)=1, rounded(2.3)= 2, rounded(4.5)=5)

(**Hint:** It is recommended to use **Math.sqrt(double)** method.)

8) void assignPoint(int p): Takes an integer value which represents the total point calculated for a KUArchitect and assigns it to the variable belongs to the KUArchitect determined by currentArchitectID.

(**Hint:** For this purpose, there are already defined variables as totalPointOfKUArchitect1, totalPointOfKUArchitect2, totalPointOfKUArchitect3, totalPointOfKUArchitect4 and currentArchitectID.)

9) void comparator(): Compares the total points of all contestants.

• Prints out the KUArchitect with maximum total point as #1 and assigns its ID number to firstID, its total point to firstPoint.

• Prints out the KUArchitect with second maximum total point as #2 and assigns its ID number to secondID, its total point to secondPoint.

• Prints out the KUArchitect with maximum total point as #3 and assigns its ID number to thirdID, its total point to thirdPoint.

Example:

totalPointOfKUArchitect1 = 80
totalPointOfKUArchitect2 = 20

totalPointOfKUArchitect3 = 100
totalPointOfKUArchitect4 = 40

Then, KUArchitect 2 is eliminated.

firstID = 3 (The ID number of the architect with highest maximum point.)
secondID = 1 (The ID number of the architect with second maximum point.)
thirdID = 4 (The ID number of the architect with third maximum point.)

firstPoint = 100 (The total point of the architect with ID number firstID.)
secondPoint = 80 (The total point of the architect with ID number secondID.)
thirdPoint = 40 (The total point of the architect with ID number thirdID.)

3.2 PART II

This is the finale part. The champion of KUArchitect is going to be found out soon.

3.2.1 Final Game

KUArchietcts make duos for one-to-one combat. For the sake of simplicity, we call KUArchitect in first position= architect1, second position architect =architect2, and the third position arhitect = architect3. In the first combat, architect1 and archecitect2 compete against each other. In the second combat, architect2 and archecitect3 compete against each other. In the third combat, architect1 and archecitect3 compete against each other.

- In each combat, the KUArchitects make NTIMES matches.
- In every match, architects have winning probability against each other depending on their total points.
- The winning of each architects is calculated after 3 combats.
- Each architect plays NTIMES x 2 matches.
- The winner of the game is decided by the total number of matches won after completion of 3 combats.

Example:

First Combat Between arhitect1 and architect2:

architect1Point = 80
architects2Point = 50

- The winning probability of architect1 and architect2 against each other are 80/130 and 50/130, respectively. (**Hint:** Using RandomGenerator is a way to do this)

3.3 Run Method

You should fill the run method by using the methods you created.

4 Sample Outputs

(In these samples, contestants' values in part 2.3 are tested.) However keep in mind that your result can vary since the result is determined by the random distribution.

```
NEW CONTESTANT:
Years of education of KUArchitect#1: 5
Years of experience of KUArchitect#1: 6
Number of projects completed by KUArchitect#1: 4
Number of different projects of KUArchitect#1: 3
Number of awards of KUArchitect#1 4
KUArchitect#1 has reached 27 points.
_____


NEW CONTESTANT:
Years of education of KUArchitect#2: 5
Years of experience of KUArchitect#2: 7
Number of projects completed by KUArchitect#2: 4
Number of different projects of KUArchitect#2: 3
Number of awards of KUArchitect#2 5
KUArchitect#2 has reached 35 points.
_____


NEW CONTESTANT:
Years of education of KUArchitect#3: 6
Years of experience of KUArchitect#3: 5
Number of projects completed by KUArchitect#3: 4
Number of different projects of KUArchitect#3: 2
Number of awards of KUArchitect#3 2
KUArchitect#3 has reached 41 points.
_____


NEW CONTESTANT:
Years of education of KUArchitect#4: 4
Years of experience of KUArchitect#4: 5
Number of projects completed by KUArchitect#4: 3
Enter a valid number of projects by KUArchitect#4: 5
Number of different projects of KUArchitect#4: 2
Number of awards of KUArchitect#4 2
KUArchitect#4 has reached 12 points.
_____


KUArchitect#3 becomes #1 with 41 points.
KUArchitect#2 becomes #2 with 35 points.
KUArchitect#1 becomes #3 with 27 points.

The game is now completed and the scores are as below:

KUArchitect#2 won 97 times in 100 games.
KUArchitect#1 won 88 times in 100 games.
KUArchitect#3 won 115 times in 100 games.

CONGRATULATIONS KUArchitect#3!! YOU ARE THE WINNER OF KUArchitect.
```

5 End of Project

Your project ends here. You may continue to tinker with the code to implement any desired features and discuss them with your section leader. However, **do not** include any additional features that you implement after this point in to your submission.

6 For Questions

For questions about the project you may contact with Ayça Tüzmen Yıldırım and Section Leader Coordinators. Note that it may take up to 24 hours before you receive a response; so, please ask your questions before it is too late. No questions will be answered when there is less than two days left for the submission.

Final Warning: Do not include anything beyond this point to your submission.