A Vicious Competition: KUisine

Submission Date: 28 October 11:59 PM

Contents

1 Introduction

1.1 Submission

Submit a folder that contains **only** your Java source file (*.java) to Comp130 homework folder.

Full path: **F:\COURSES\UGRADS\COMP130\Homework\**

Note: Comp 131 students, please submit to the folder under the COMP 130 directory.

Please use the following naming convention for the submitted folders:

YourPSLetter_CourseCode_Surname_Name_HWNumber_Semester

Example folder names:

- PSA_COMP130_Orcun_Suheyla_Ezgi_HW2_F18

- PSB_COMP131_Kanil_Ayda_HW2_F18

Important notes:

- Using the naming convention properly is important, failing to do so may be **penalized**.

- **Do not** use Turkish characters when naming files or folders.

- Submissions with unidentifiable names will be **disregarded** completely. (ex. ”homework2, ”project” etc.)

- 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. For this project, you are expected to implement a program that simulates a cooking competition which consists of two parts, audition and finale.

2 Plan

2.1 Overview

KUisine is the competition to find the best cook in Koç University. Let us go deeper into it:

In the audition part, contestants are tested by their **knife skills**, their **cooking skills**, **the years of culinary education** they received, **the number of different types of cuisines** and **special dishes** they are able to prepare. To spice up the audition part, we flip a coin for every contestant to decide whether they are given **a recipe book** to boost their knowledge or not. According to these factors, contestants gain individual total points which determine the triple passing to the finale.

In the finale, the finalists make duos to cook against each other. The contestant who cooks better wins the combat. For every two winnings, the contestant earns a point. These points are used to determine the winner.

2.2 Variables

KUisine is a competition for non-professional cooks. Therefore, top chefs and, obviously, people who cannot cook are not welcome. So, the qualities of the contestants are required to be in some restricted intervals. (The boundaries are also included.)

Ability:

Knife Skill: The first step to great food is great knife skills. KUisine demands KUchefs to have knife skills between **integers 2 and 6**.

Cooking Skill: No one would want to have a raw or burnt meat. So, chefs should master their cooking skills to a point. It is bounded between the **integer values 5 to 15**.

Knowledge:

Education: A culinary education takes **2-4 years** of degree and it is a must to have for attending KUisine.

Cuisine: By tasting food, you also taste culture. Knowing **5 to 10 different types** of cuisines makes you sophisticated enough.

Special: Every wanna-be chef wants to have their own famous special someday. Having **1 to 5 potential dishes** is required for KUisine.

int totalPointOfKUchef1, totalPointOfKUchef2, totalPointOfKUchef3, totalPointOfKUchef4: Defined as integer variables to keep total points of the four contestant in the first part.

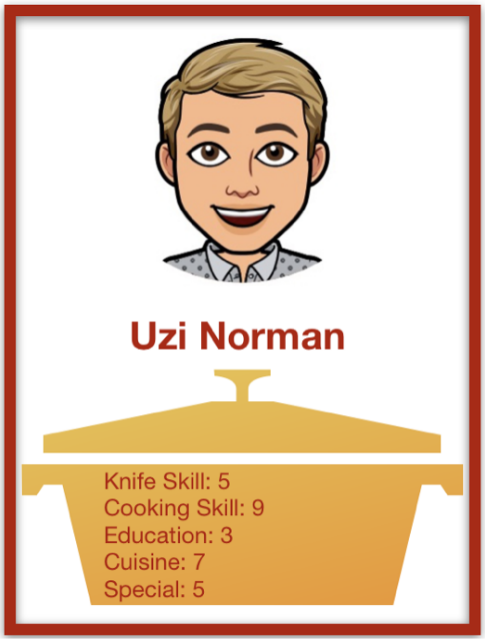
int firstID, secondID, thirdID: Defined as integer variables to keep the digit numbers of the first three chefs passing to the finale.

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

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

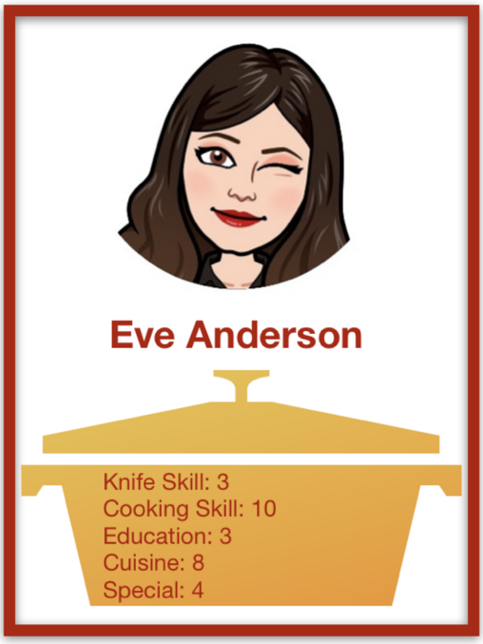
2.3 Contestants

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



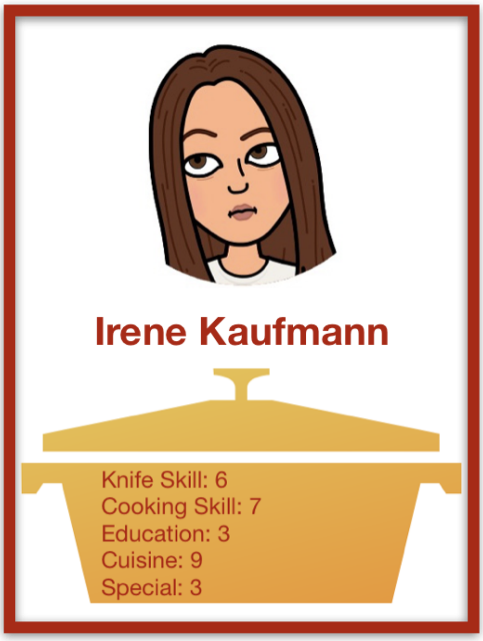
Uzi Norman

Knife Skill: 5
Cooking Skill: 9
Education: 3
Cuisine: 7
Special: 5



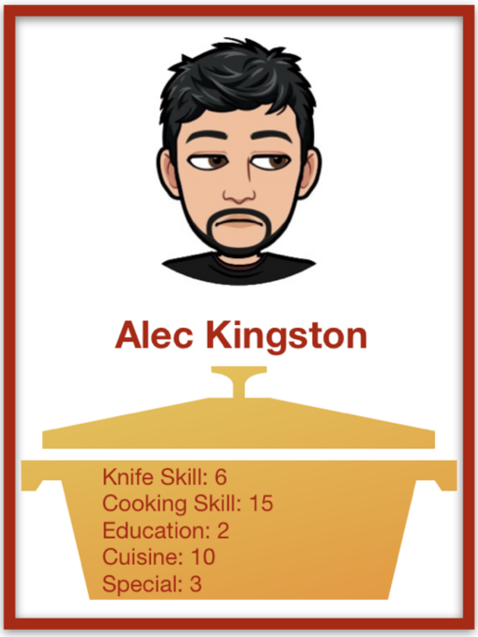
Eve Anderson

Knife Skill: 3
Cooking Skill: 10
Education: 3
Cuisine: 8
Special: 4



Irene Kaufmann

Knife Skill: 6
Cooking Skill: 7
Education: 3
Cuisine: 9
Special: 3



Alec Kingston

Knife Skill: 6
Cooking Skill: 15
Education: 2
Cuisine: 10
Special: 3

2.4 Constants

In the code given to you, we have created 2 constant values 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.

NTIMES: Set as 10000 to be the match number in Part 2.

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 askAbility(): Prompts user to enter two integer values for knife skill and cooking skill and reads the entries **until a valid input is given.** Then, it calculates the ability point of the KUchef by making a call to calculateAbility method and returns the result.

2) double calculateAbility(int ks, int cs): Takes knife skill and cooking skill. It calculates and returns the ability point using the following formula:

$$AbilityPoint = (\frac{KnifeSkill + CookingSkill}{10})^{(KnifeSkill)} + fibonacci(CookingSkill) \tag{1}$$

Example:

$$AbilityPoint = (\frac{6+7}{10})^{6} + fibonacci(7) \tag{2}$$

$$= (1.3)^{6} + 13 \tag{3}$$

$$= 17.826809 \tag{4}$$

(**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... Ex: fibonacci(1)=1, fibonacci(2)=1; fibonacci(6)=8)

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

3) int askKnowledge(): Prompts user to enter two integer values for the years of education and number of different cuisines and reads the entries **until a valid input is given.** Then, it calculates the knowledge point by using calculateKnowledge method and returns the answer.

4) int calculateKnowledge(int e, int c): Takes years of education and number of different cuisines and returns the knowledge point using the following formula:

$$KnowledgePoint = c \times e! \tag{5}$$

(**Hint:** Implementing a factorial method may be useful.)

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

6) boolean hasBook(): Determines whether a KUchef gains a recipe book or not randomly. The probability to gain a book is 50%.

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

7) int pointCalculator(double ability, int knowledge, int special, boolean hasBook): Calculates the total point of a KUchef using this partial function formula:

$$TotalPoint = \begin{cases} \sqrt{ability \times special} + 1.5 \times knowledge, & hasBook = true \\ \sqrt{ability \times special} + 1 \times knowledge, & hasBook = false \end{cases}$$

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 KUchef and assigns it to the variable belongs to the KUchef determined by currentChefID.

(**Hint:** For this purpose, there are already defined variables as totalPointOfKUchef1, totalPointOfKUchef2, totalPointOfKUchef3, totalPointOfKUchef4 and currentChefID.)

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

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

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

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

Example:

totalPointOfKUchef1 = 80
totalPointOfKUchef2 = 20
totalPointOfKUchef3 = 100
totalPointOfKUchef4 = 40

Then, KUchef 2 is eliminated.

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

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

3.2 Part II

This is the finale part. The champion of the KUisine is going to be found out soon. **(You are required to write your own helper methods.)**

3.2.1 Distribution

In the final game, it is important in which order the contestants participate. So, to have a fair game, you need to assign KUchefs in a randomly selected order from all possible placements. As you know, 3 people have 3!=6 different distributions.

3.2.2 Final Game

KUchefs make duos for one-to-one combats. In a limited time, they have to cook attentively specified dishes for them. The general rules that hold for the game can be found below: (For the sake of simplicity, chef 1, chef 2 and chef 3 are used to understand the context.)

Combat: A single one-to-one competition
Match: The competition that continues until a chef wins two combats
Game: The competition consists of NTIMES matches

- In the first combat, chef 1 and chef 2 compete against each other. In the second combat, chef 2 and chef 3 compete against each other. In the third combat, chef 1 and chef 3 compete against each other. In the fourth combat, chef 1 and chef 2 compete against each other again and it continues in this pattern.

- In every single combat, chefs have winning probabilities against each other depending on their total points.

Example :

First Combat: (Chef 1 vs Chef 2)

firstPoint= 80
secondPoint = 50

The winning probabilities of chef 1 and chef 2 against each other are 80/130 and 50/130, respectively. (**Hint:** Using RandomGenerator is a way to do this.)

- When a chef wins **two combats**, s/he gets one point.

- When a chef gets a point, a match is completed.

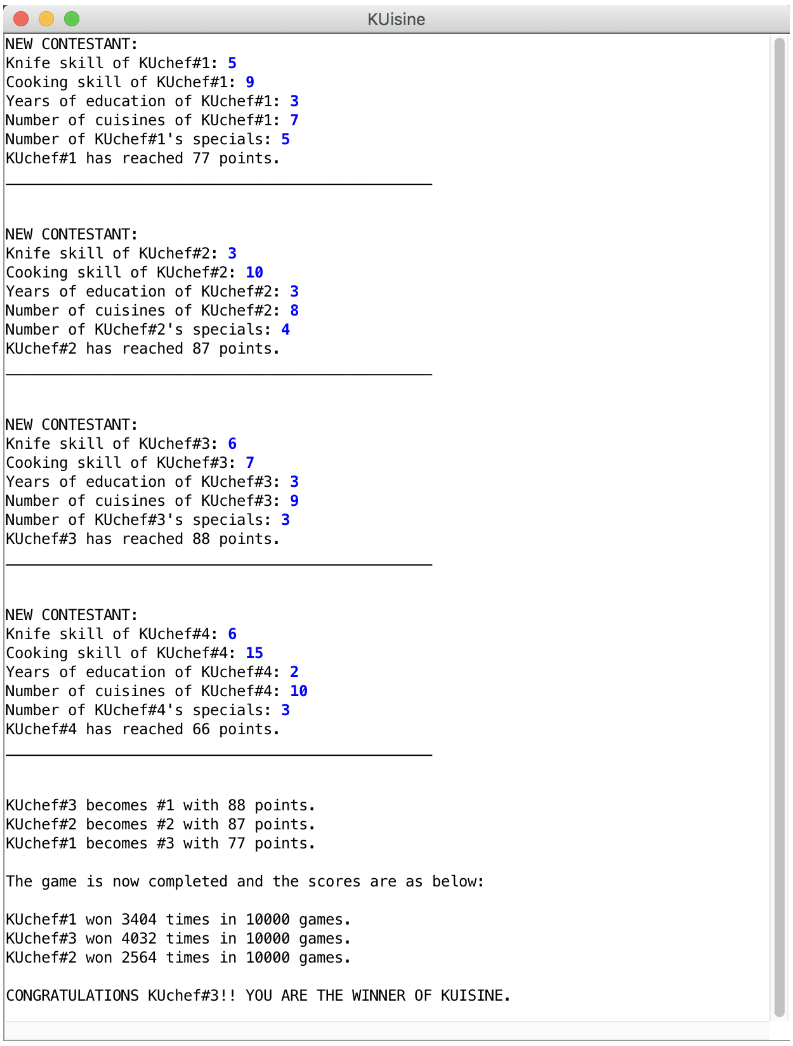
- Chefs make NTIMES matches and the most winning chef wins the game.

3.3 Run Method

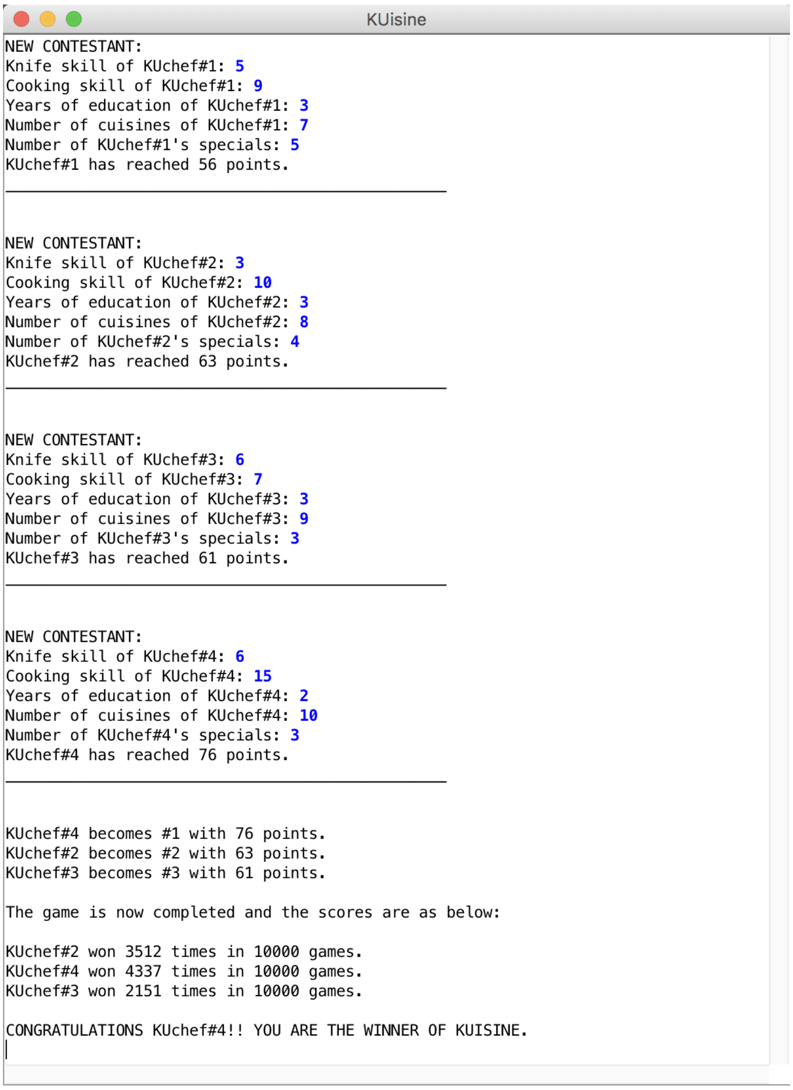
You should fill the run method by using the methods you created.

4 Sample Outputs

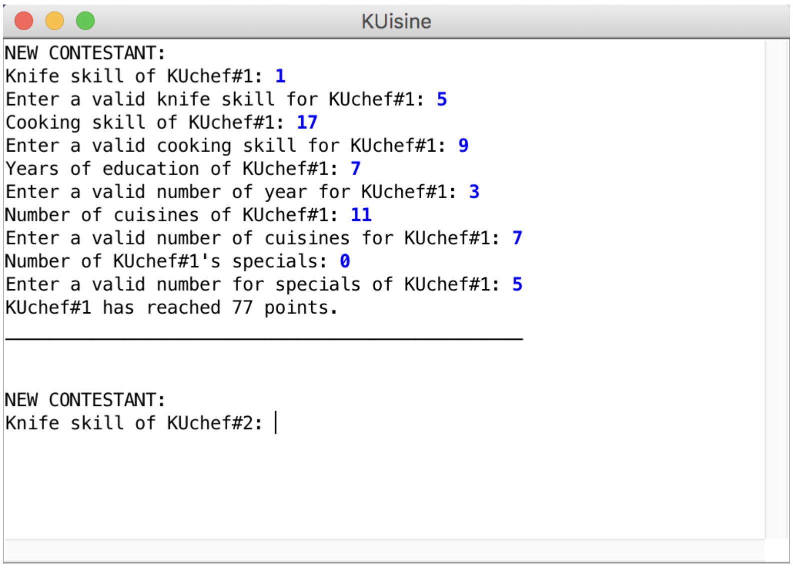
(In these samples, contestants' values in part 2.3 are tested.)



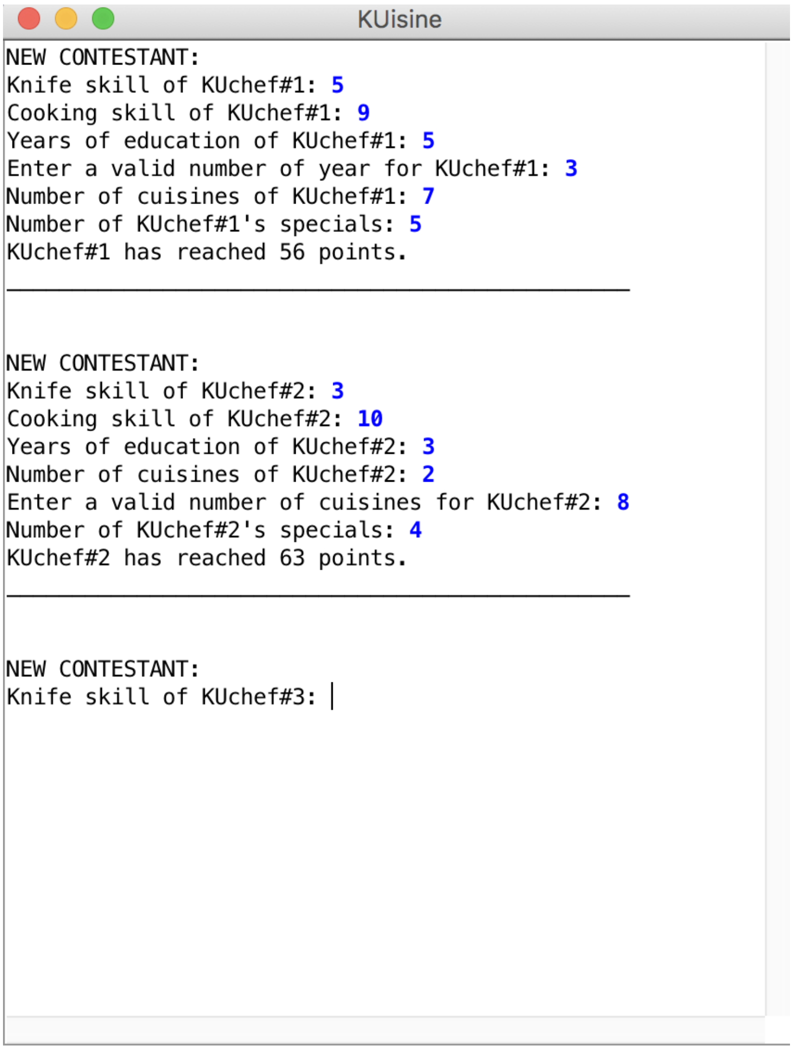
```
NEW CONTESTANT:
Knife skill of KUchef#1: 5
Cooking skill of KUchef#1: 9
Years of education of KUchef#1: 3
Number of cuisines of KUchef#1: 7
Number of KUchef#1's specials: 5
KUchef#1 has reached 77 points.
________________________________________________


NEW CONTESTANT:
Knife skill of KUchef#2: 3
Cooking skill of KUchef#2: 10
Years of education of KUchef#2: 3
Number of cuisines of KUchef#2: 8
Number of KUchef#2's specials: 4
KUchef#2 has reached 87 points.
________________________________________________


NEW CONTESTANT:
Knife skill of KUchef#3: 6
Cooking skill of KUchef#3: 7
Years of education of KUchef#3: 3
Number of cuisines of KUchef#3: 9
Number of KUchef#3's specials: 3
KUchef#3 has reached 88 points.
________________________________________________


NEW CONTESTANT:
Knife skill of KUchef#4: 6
Cooking skill of KUchef#4: 15
Years of education of KUchef#4: 2
Number of cuisines of KUchef#4: 10
Number of KUchef#4's specials: 3
KUchef#4 has reached 66 points.
________________________________________________


KUchef#3 becomes #1 with 88 points.
KUchef#2 becomes #2 with 87 points.
KUchef#1 becomes #3 with 77 points.

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

KUchef#1 won 3404 times in 10000 games.
KUchef#3 won 4032 times in 10000 games.
KUchef#2 won 2564 times in 10000 games.

CONGRATULATIONS KUchef#3!! YOU ARE THE WINNER OF KUISINE.
```



```
NEW CONTESTANT:
Knife skill of KUchef#1: 5
Cooking skill of KUchef#1: 9
Years of education of KUchef#1: 3
Number of cuisines of KUchef#1: 7
Number of KUchef#1's specials: 5
KUchef#1 has reached 56 points.
______________________________________________


NEW CONTESTANT:
Knife skill of KUchef#2: 3
Cooking skill of KUchef#2: 10
Years of education of KUchef#2: 3
Number of cuisines of KUchef#2: 8
Number of KUchef#2's specials: 4
KUchef#2 has reached 63 points.
______________________________________________


NEW CONTESTANT:
Knife skill of KUchef#3: 6
Cooking skill of KUchef#3: 7
Years of education of KUchef#3: 3
Number of cuisines of KUchef#3: 9
Number of KUchef#3's specials: 3
KUchef#3 has reached 61 points.
______________________________________________


NEW CONTESTANT:
Knife skill of KUchef#4: 6
Cooking skill of KUchef#4: 15
Years of education of KUchef#4: 2
Number of cuisines of KUchef#4: 10
Number of KUchef#4's specials: 3
KUchef#4 has reached 76 points.
______________________________________________


KUchef#4 becomes #1 with 76 points.
KUchef#2 becomes #2 with 63 points.
KUchef#3 becomes #3 with 61 points.

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

KUchef#2 won 3512 times in 10000 games.
KUchef#4 won 4337 times in 10000 games.
KUchef#3 won 2151 times in 10000 games.

CONGRATULATIONS KUchef#4!! YOU ARE THE WINNER OF KUISINE.
```



```
NEW CONTESTANT:
Knife skill of KUchef#1: 1
Enter a valid knife skill for KUchef#1: 5
Cooking skill of KUchef#1: 17
Enter a valid cooking skill for KUchef#1: 9
Years of education of KUchef#1: 7
Enter a valid number of year for KUchef#1: 3
Number of cuisines of KUchef#1: 11
Enter a valid number of cuisines for KUchef#1: 7
Number of KUchef#1's specials: 0
Enter a valid number for specials of KUchef#1: 5
KUchef#1 has reached 77 points.
______________________________________________


NEW CONTESTANT:
Knife skill of KUchef#2:
```



```
NEW CONTESTANT:
Knife skill of KUchef#1: 5
Cooking skill of KUchef#1: 9
Years of education of KUchef#1: 5
Enter a valid number of year for KUchef#1: 3
Number of cuisines of KUchef#1: 7
Number of KUchef#1's specials: 5
KUchef#1 has reached 56 points.
______________________________________________


NEW CONTESTANT:
Knife skill of KUchef#2: 3
Cooking skill of KUchef#2: 10
Years of education of KUchef#2: 3
Number of cuisines of KUchef#2: 2
Enter a valid number of cuisines for KUchef#2: 8
Number of KUchef#2's specials: 4
KUchef#2 has reached 63 points.
______________________________________________


NEW CONTESTANT:
Knife skill of KUchef#3: |
```

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, Ata Kadirioğlu, Ece Azizoğlu, İpek Köprülülü, Utku Noyan. 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.