CSE2DES/CSE5DES - 2019 - Assignment - Part 1

Due Date: 10.00 am, Wednesday 18 September 2019

Assessment: This part is worth 50% of the total assignment mark, i.e. it is worth 15% of the final marks.

Assignment Submission: An electronic copy in **pdf format**, containing answers to both Task 1 and Task 2, is to be submitted electronically via **latcs8** using the submit command:

```
submit DES <file name>
```

You may need to transfer files from Windows to Unix, in the labs or from home. In this case, we must learn and practise how to do this *well in advance of the due date.*

This is an individual assignment. You are not permitted to work as a group when writing this assignment.

Copying, Plagiarism: Plagiarism is the submission of somebody else's work in a manner that gives the impression that the work is your own. The Department of Computer Science and Computer Engineering treats plagiarism very seriously. When it is detected, penalties are strictly imposed. Students are referred to the Department of Computer Science and Computer Engineering's Handbook and policy documents with regard to plagiarism.

No extensions will be given: Penalties are applied to late assignments (5% is deducted per day, accepted up to 5 days after the due date only). If there are circumstances that prevent the assignment being submitted on time, an application for special consideration may be made. See the departmental Student Handbook for details. Note that delays caused by computer downtime cannot be accepted as a valid reason for a late submission without penalty. Students must plan their work to allow for both scheduled and unscheduled downtime.

Return of Assignments: Students are referred to the departmental Student Handbook for details. It is planned that the assignments will be returned within three weeks of of the due date.

Objectives: To learn to identify the relevant use-cases for a given application, describe the use cases, and develop an object-oriented domain model.

Problem Statement

International Figure Skating (IFS) is an organization that organizes figure skating competitions at international level. Competitions are organized several times a year in different countries.

IFS needs a system to organize and maintain records in a single competition. The system, in essence, needs to keep information on the skaters, their competition entries, the judges, and the information necessary to organizing the performance sessions, and entering the judges' scores and and processing results.

Judges and Skaters. Each person (a judge or a skater) is identified by an id. Also recorded are the person's name, phone number, email address and the person's nationality. In addition, a skater also has their gender and date of birth recorded.

A look-up table for nationalities is available, which lists the nationality three-letter codes and names.

Competition Disciplines. There are three competition disciplines:

- Ladies' Singles (for female skaters),
- Mens' Singles (for male skaters) and
- Pair Skating (for pairs of a female and a male skater).

Thus, a female skater, for example, can make an entry for competing in Ladies' Singles or Pair Skating (with a partner) or both.

Disciplines can be identified by the codes "LS", 'MS" and "PS" for Ladies' Singles, Mens' Singles and and Pair Skating, respectively. Each competition entry is identified by a unique id.

Competition Programs. There are two competition programs:

- Technical program and
- Free program.

For each entry of competition, the skater or the pair must undertake two performances: a short performance for the technical program, and a longer performance for the free program. For the technical program, a performance is scored on a number of technical requirements. For the free program, a performance is scored on both technical and artistic qualities.

Programs can be identified by the codes "TP" and "FP" for technical program and free program, respectively.

Competition Sessions. The performances are allocated to sessions. Each session is for one particular discipline (e.g. Ladies' Singles) and one program (e.g. technical program). Each session has a unique session number, the session's date, the starting time and the ending time. Each session has a panel of nine judges, who each comes from a different country.

Scoring. After each performance, each of judges gives it a score between 0 and 6 inclusive with decimals to one place allowed as intermediate values. The score scale is:

- 0 = not skated
- 1 = bad, very poor
- 2 = poor
- 3 = average
- 4 = good
- 5 = excellent
- 6 = perfect

For each performance, the score of each individual judge must be recorded. The scores of the judges are added up to be the performance's score. The technical program performance's score and free program performance's score of an entry are added up to be the total score of the entry (of a skater or a pair of skaters). The total scores are used for the final ranking for each of the three competition disciplines.

Tasks

For Part 1 of the assignment, you are required to do the tasks described below.

Task 1 – Use Case Modeling

For the purpose of building a prototype (which you will do for Part 2), the following use cases have been selected for the initial focus.

1. Add a skater

   The purpose of this use case is to enter the details about a skater. It is not concerned with entering the details about the skater's entry or entries.

2. Add a judge

   To enter the details about a judge.

3. Add a Ladies' Singles entry

4. Add a Mens' Singles entry

5. Add a Pair Skating entry

6. Add a session

   The purpose of this use case is to enter the initial details about a session, to start the session arrangement. It is not concerned with entering the details about the judges and competition entries assigned to the session.

7. Assign a performance of an entry to a session

8. Assign a judge to a session

9. Enter the score given by a judge for a performance

Your task is to **describe the use cases listed above**.

- The descriptions must be written from the **pure text-based interaction viewpoint**. (See Chapter 3, Part 2, Section 4.2, and Lab 3, pp.4-6)

- There are two kinds of validations: state-dependent and state-independent. The validation that a person id is unique is a state-dependent validation: the outcome depends on the set of id's we currently have in the system. By contrast, the validation of a gender value entered for a person is state-independent: the outcome, for a particular value entered, remains the *same* throughout the operation of the system.

  In describing use cases, state-dependent validations are *mandatory*, and state-independent validations are *optional*, i.e. they can be *implicitly* assumed.

- Clearly state any assumption you make. Your assumptions must not be inconsistent with what are given the description above.

  In addition, *you must **not** assume that identifiers such as person id's and session numbers are automatically generated.*

Task 2 - Structural Domain Modeling

Construct a structural domain model for the application. (The structural domain model is also known as the "domain class model" or simply the "domain model" for short)

- Your model **must follow the modeling choices** described below.
- If you want to, you can present the structural domain model in more than one class diagram to avoid "congestion" in some diagram.
- The use of Enterprise Architect to draw the class diagrams is strongly recommended.
- You should include your class diagrams in your pdf document, rather than in separate files.
- Diagrams that are drawn by hand will have 50% of its marks deducted.

Modeling Choice 1 - Regarding competition disciplines

Consider the following extract from the description above:

> *There are three competition disciplines: Ladies' Singles (for female skaters), Mens' Singles (for male skaters) and Pair Skating (for pairs of a female and a male skater).*

The above description may prompt us to model competition discipline concepts by an inheritance hierarchy, with a superclass and three subclasses. In fact, we have three possible choices:

1. To represent the disciplines by one superclass and three subclasses

   (Point to ponder: In this case, how many instances does each of the subclass have? Do the instances necessarily need to have any attributes?)

2. To represent them by one class only, namely `Discipline`, which has one attribute to indicate the one of the three specific disciplines

   (Point to ponder: In this case, how many instances does the class have?)

3. To represent them by an enumerated type with three values (instead of using the non-specific string type)

Any of these choices can do the job, which is to *indicate* if a particular entry is a Ladies' Singles entry, a Mens' Singles entry or Pair Skating entry.

> In the interest of simplicity, we will adopt the third solution, in which we represent disciplines by an enumerated type. *Your model must use this modeling choice.*

Modeling Choice 2 – Regarding the technical and free programs

> *For each entry of competition, the skater or the pair must undertake two performances: a short performance for the technical program, and a longer performance for the free program.*

As for the previous case, we have three possible choices:

1. To represent the programs by one superclass and two subclasses
2. To represent them by one class only, namely `Program`, which has an attribute to indicate the specific program
3. To represent them by an enumerated type with two values

Similar to the previous case, each of these choice can do the required job of indicating if a performance for an entry is for a technical program or a free program.

> We will adopt the third choice, in which we represent the programs by an enumerated type. *Your model must use this modeling choice.*

Modeling Choice 3 – Regarding the competition entries

We have several choices:

1. To use an inheritance hierarchy with superclass `Entry` and three subclasses, `Ladies' Single Entry`, `Mens' Single Entry` and `Pair Skating Entry`
2. To use an inheritance hierarchy with superclass `Entry` and two subclasses, `Single Entry` and `Pair Entry`
3. To use one class, `Entry`

The third choice makes it difficult to clearly model relationship between entries and skaters because the different types of `Entry` have different number of associations with `Skater`. That leaves us with the first two choices. There is not much to choose between them. We will make a somewhat arbitrary decision to choose option 2.

> We will adopt the second choice, in which we have superclass `Entry` and two subclasses `Single Entry` and `Pair Entry`. *Your model must use this modeling choice.*

Mark Allocations

- **Task 1: 50 marks**

  You are required to describe all the use cases listed. But about 5 use cases will be selected for marking.

- **Task 2: 50 marks**

- You must include your
  - Student ID,
  - First name,
  - Surname, and
  - Whether the subject is CSE2DES or CSE5DES

  on first page, with clear indication as to which is your first name and which is your surname. Failure to do this will have 5 marks deducted.

■