IT2154 Assignment 1: Total 100 marks (25%)

Instructions:

1. This is an individual assignment. Student who is caught cheating and practice ***plagiarism*** will face the following penalty according to **NYP policy**.
    a. First time in any assessment will fail the entire module.
    b. Second time in any assessment will fail all modules in that semester.
    c. Third time in any assessment will be removed from the Polytechnic.
2. Download the MushroomPocket.zip file from Brightspace.
3. You are responsible for the correct and complete submission of your assignment.
4. Include your name and admin number in the beginning of each file.
5. Zip up the whole application folder, rename it as “<admin_no>_<name>_ASN1.zip” and submit via Brightspace.

Objective of the assignment

Apply C# programming knowledge such as OOP concept & Entity Framework in a scenario-based console application.

(Refer to **Appendix 1** for the Assessment Rubrics).

Scenario of the assignment

This assignment is to create a **Mushroom Pocket** console application to let Super Mario’s player keep, view the characters they played and check if they can transform to a higher role character to protect the Mushroom Kingdom. If the characters in the pocket are ready to transform, the player can transform their characters. Each time when players played a character, they can add the character details to this Mushroom Pocket program.

The character transformation criteria in this application are imaginary version which means that it was totally different from the actual Super Mario game play. The transformation of a character here is based on the number of the same character that the player has played.

The current program has 1 class object, **MushroomMaster** class with attributes that describe the transformation criteria such as the name of the character, number of the same character to be transformed and what it will transform to. This class has 3 objects – **Waluigi**, **Daisy** and **Wario**.

Create a **Character** class and implement subclasses for these 3 characters. This means that the players can only add these 3 Mushroom characters to the pocket. You can add more Mushroom characters subclasses to make your program more interesting if you want to. The Mushroom class should have the following attributes:

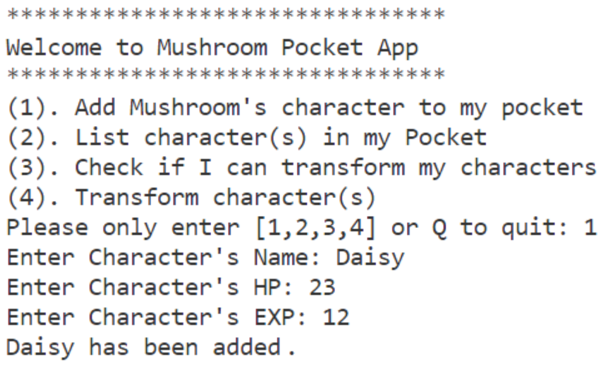
- Name of character,
- HP,
- EXP,
- Skill

The following is the skill description for each Mushroom character.

Mushroom Characters	**Skill**
Waluigi	Agility
Daisy	Leadership
Wario	Strength
Luigi	Precision and Accuracy
Peach	Magic Abilities
Mario	Combat Skills

Specifications for the Mushroom Pocket Program

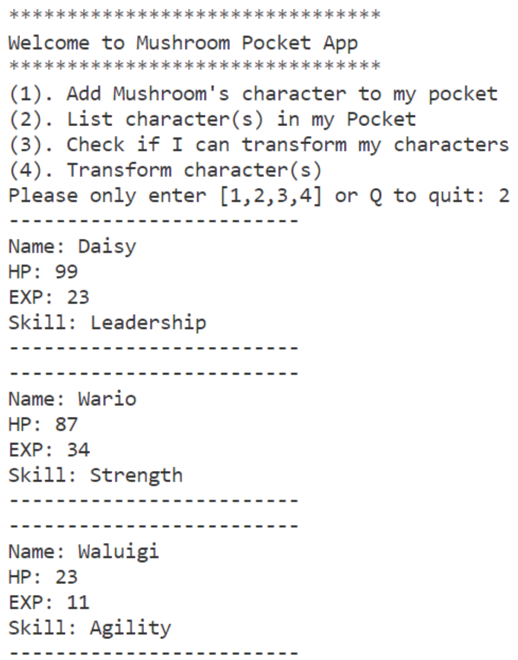
1. Repeatedly display a menu for players to enter their choices until ‘Q’ or ‘q’ is being entered to exit the program. Input validation is needed to prevent run-time exception.



```
********************************
Welcome to Mushroom Pocket App
********************************
(1). Add Mushroom's character to my pocket
(2). List character(s) in my Pocket
(3). Check if I can transform my characters
(4). Transform character(s)
Please only enter [1,2,3,4] or Q to quit: 1
Enter Character's Name: Daisy
Enter Character's HP: 23
Enter Character's EXP: 12
Daisy has been added.
```

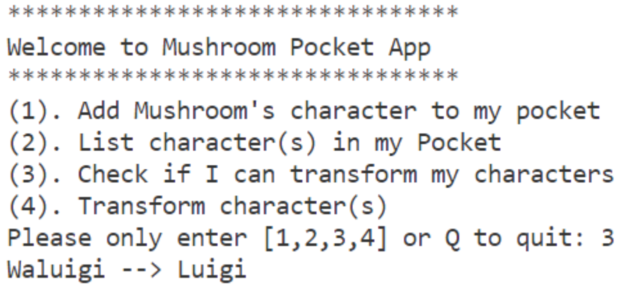
2. For implementation of option (1) in the menu, prompt user to enter data of the character they played and store it in the Mushroom Pocket. Input validation is needed to prevent run-time exception.

3. For implementation of option (2), list down the characters that the user entered and sort the list by HP in *descending* order.



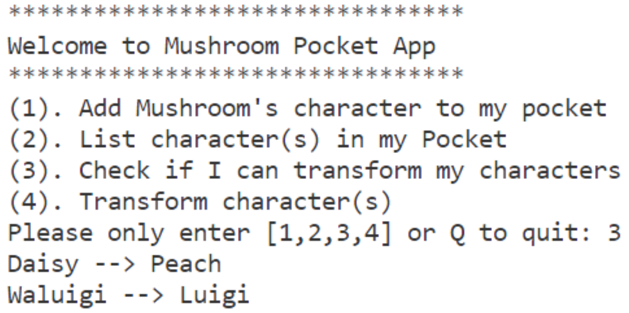
```
********************************
Welcome to Mushroom Pocket App
********************************
(1). Add Mushroom's character to my pocket
(2). List character(s) in my Pocket
(3). Check if I can transform my characters
(4). Transform character(s)
Please only enter [1,2,3,4] or Q to quit: 2
-------------------------
Name: Daisy
HP: 99
EXP: 23
Skill: Leadership
-------------------------
-------------------------
Name: Wario
HP: 87
EXP: 34
Skill: Strength
-------------------------
-------------------------
Name: Waluigi
HP: 23
EXP: 11
Skill: Agility
-------------------------
```

4. For implementation of option (3), list character(s) that can be transformed in the following format.



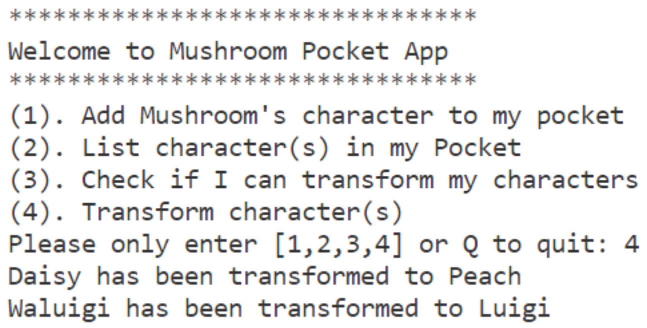
```
********************************
Welcome to Mushroom Pocket App
********************************
(1). Add Mushroom's character to my pocket
(2). List character(s) in my Pocket
(3). Check if I can transform my characters
(4). Transform character(s)
Please only enter [1,2,3,4] or Q to quit: 3
Waluigi --> Luigi
```

If there are 2 and more characters can be transformed, it will display in the following format.

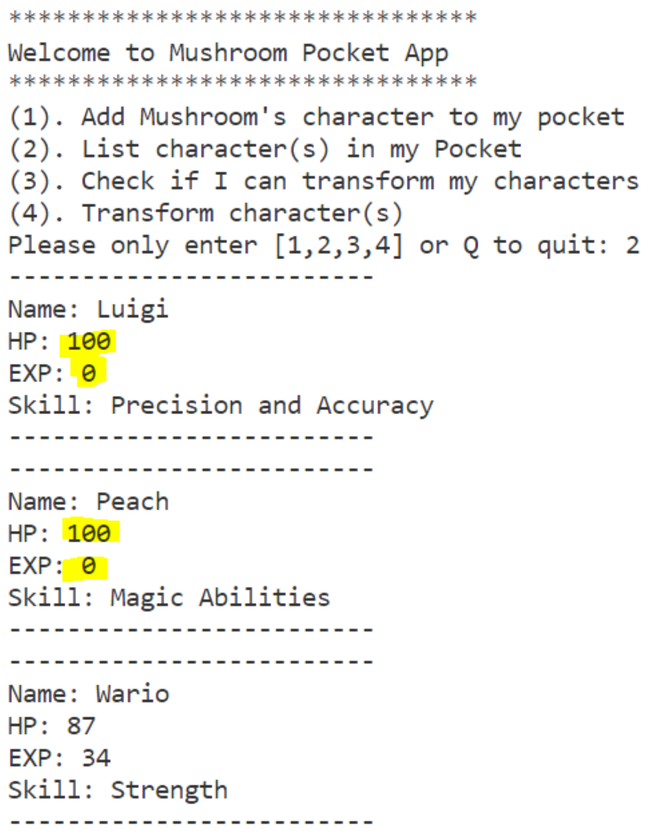


```
********************************
Welcome to Mushroom Pocket App
********************************
(1). Add Mushroom's character to my pocket
(2). List character(s) in my Pocket
(3). Check if I can transform my characters
(4). Transform character(s)
Please only enter [1,2,3,4] or Q to quit: 3
Daisy --> Peach
Waluigi --> Luigi
```

5. For implementation of option (4), transform character(s). The newly transformed character will have its HP=100, EXP=0 and Skill= [based on the newly transformed abilities].



```
********************************
Welcome to Mushroom Pocket App
********************************
(1). Add Mushroom's character to my pocket
(2). List character(s) in my Pocket
(3). Check if I can transform my characters
(4). Transform character(s)
Please only enter [1,2,3,4] or Q to quit: 4
Daisy has been transformed to Peach
Waluigi has been transformed to Luigi
```



```
********************************
Welcome to Mushroom Pocket App
********************************
(1). Add Mushroom's character to my pocket
(2). List character(s) in my Pocket
(3). Check if I can transform my characters
(4). Transform character(s)
Please only enter [1,2,3,4] or Q to quit: 2
-------------------------
Name: Luigi
HP: 100
EXP: 0
Skill: Precision and Accuracy
-------------------------
-------------------------
Name: Peach
HP: 100
EXP: 0
Skill: Magic Abilities
-------------------------
-------------------------
Name: Wario
HP: 87
EXP: 34
Skill: Strength
-------------------------
```

--End of Assignment—

Appendix 1: IT2154 Assignment 1 Rubrics: Total 100 marks (25%)

Criteria	Not Ready (Below D) 0-40 marks	Beginning (D/D+) < 60 marks	Developing (C/C+) < 70 marks	Functional (B/B+) < 80 marks	Advanced (A/A+) 80-100 marks
Implementation (55%)	Does not implement at least 1 basic feature as required for the assignment. The program does not work and has many errors.	Successfully implemented at least 2 basic features required. Program works with minimum errors but does not produce correct results nor display correctly. No validation and exceptional handling implemented.	Successfully implemented at least 3 basic features as required. Program works with minimum non-critical errors and most of the time produces expected outcomes. Little validation and exceptional handling implemented.	Besides successful implementation of all basic features as required, at least 1 additional simple feature with good usability is correctly implemented. Program works well with minimum errors and consistently produces expected outcomes. Adequate validation and exceptional handling implemented.	Besides successful implementation of all basic features as required, at least 1 additional comprehensive & creative feature with good usability is well implemented. Program works well with no errors and consistently produces expected outcomes. Full validation and exceptional handling implemented.
Coding Practices (15%)	No demonstration of any good coding practice (e.g., naming convention, readable alignment, etc)	The program lacks organization and logic.	The program has some organization and logic.	The program is organized and logical.	The program is particularly well-organized, logical, and well-debugged.

OOP Concepts & Usage of Entity Framework (30%)	No demonstration of OOP concept	The program code shows poor understanding and implementation of applying OOP in C# programming and how the objects work together. No problem-solving skill in C# observed in the code.	The program code shows little understanding and implementation of applying OOP in C# programming and how objects work together. It demonstrates little problem-solving skill in C# observed in the code.	The program code shows understanding and implementation of applying OOP in C# programming and how the objects work together to meet a goal. It demonstrates adequate problem-solving skill in C# observed in the code.	The program code shows advanced understanding and implementation of applying OOP and Entity Framework in C# programming. It demonstrates good problem-solving level in C# observed in the code.

--End of Assignment 1 Rubrics--