Self-Practice Exercises - B

Use of Dictionary and Function

You have earlier learnt how dictionary data structure can be used to store information using the key-value pair relationship. The property of a dictionary is that the key has to be immutable (i.e. cannot be changed). For more sophisticate information, the key itself can be further extended to become a **composite key** that contains multiple elements by using a tuple. Similarly, you can also extend the value to consist of multiple elements such as by using list data structure.

Ex 1: The following table shows records of students' quizzes results based on different groups (i.e. FSA to FSC).

Group	Student ID	Grades	
		LQ1	EQ1
FSA	1	80	63
FSA	2	75	81
FSA	3	85	60
FSA	4	65	75
FSB	1	65	70
FSB	2	60	73
FSB	3	70	70
FSB	4	80	73
FSC	1	80	90
FSC	2	70	75
FSC	3	75	73
FSC	4	60	80

We can create a dictionary with composite key in a program to store (some of) the entries using the following definitions.

```
grades_dict = {('FSA', 1): [80, 63],
               ('FSA', 2): [75, 81],
               ('FSA', 3): [85, 60],
               ('FSB', 1): [65, 70],
               ('FSB', 2): [60, 73],
               ('FSC', 1): [80, 90] }
```

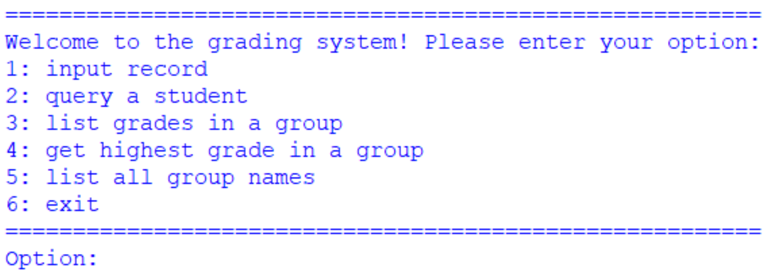
Indexing can then be used to select a specific item within the key tuple or value within the list, i.e. key[0] and key[1], and value[0] and value[1].

In this exercise, you are to develop a grading system consisting of the following functionalities:

1. Allow the user to input the record information as shown in the table above.
2. Query the grade of a student in a certain group
3. List all the grades in a lab group
4. Get the highest grade in a lab group
5. List all the groups available in the system
6. Exit the system

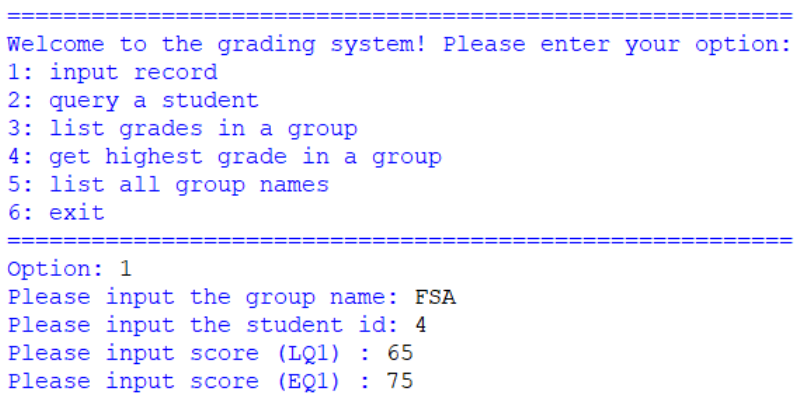
You are required to code each functionality in the form of function, and call them from your main program based on user's selection.

Your program should hence first prompt the user with a menu to select the specific function. Example of such a menu is as follows.

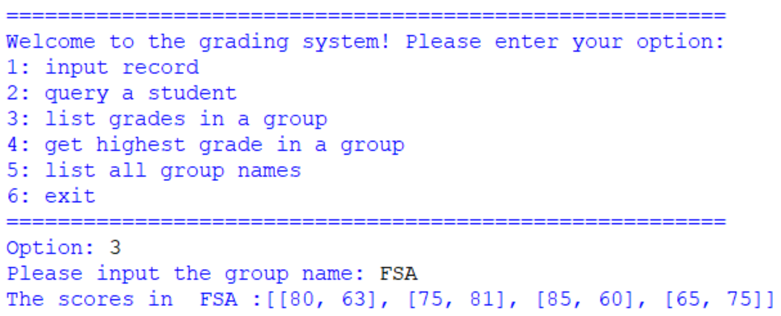


```
========================================================
Welcome to the grading system! Please enter your option:
1: input record
2: query a student
3: list grades in a group
4: get highest grade in a group
5: list all group names
6: exit
========================================================
Option:
```

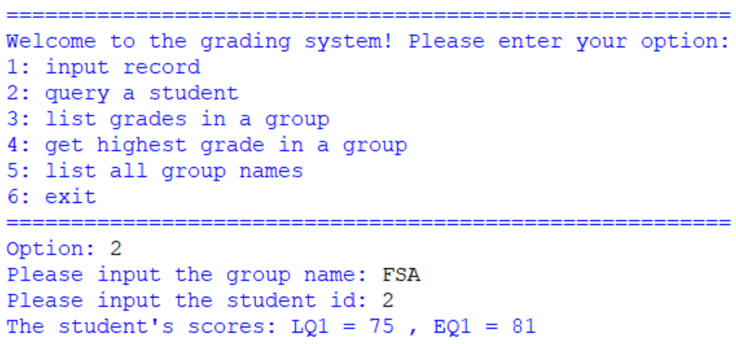
The following screenshots show examples of selecting the respective options.



```
========================================================
Welcome to the grading system! Please enter your option:
1: input record
2: query a student
3: list grades in a group
4: get highest grade in a group
5: list all group names
6: exit
========================================================
Option: 1
Please input the group name: FSA
Please input the student id: 4
Please input score (LQ1) : 65
Please input score (EQ1) : 75
```

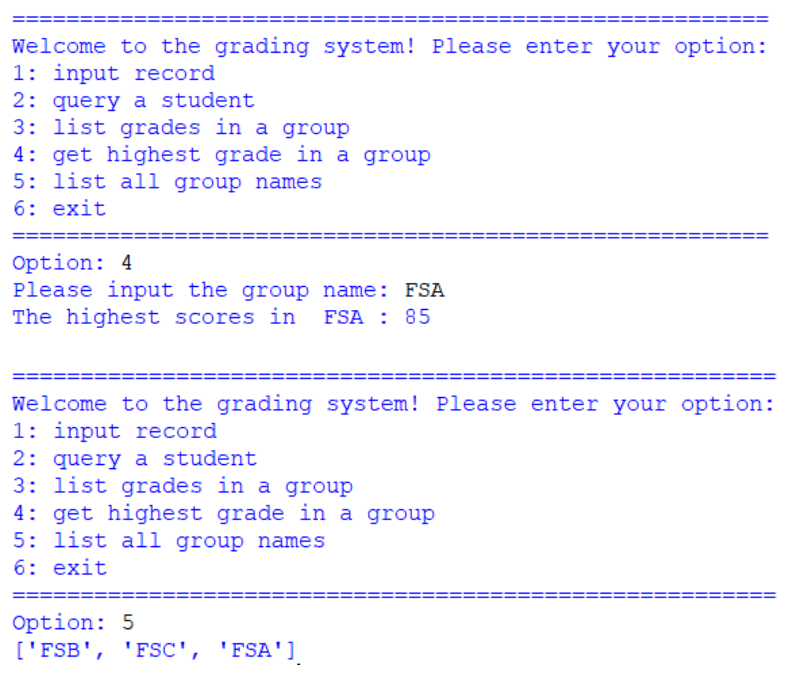


```
========================================================
Welcome to the grading system! Please enter your option:
1: input record
2: query a student
3: list grades in a group
4: get highest grade in a group
5: list all group names
6: exit
========================================================
Option: 3
Please input the group name: FSA
The scores in  FSA :[[80, 63], [75, 81], [85, 60], [65, 75]]
```



```
========================================================
Welcome to the grading system! Please enter your option:
1: input record
2: query a student
3: list grades in a group
4: get highest grade in a group
5: list all group names
6: exit
========================================================
Option: 2
Please input the group name: FSA
Please input the student id: 2
The student's scores: LQ1 = 75 , EQ1 = 81
```

Self-Practice Exercises - B



```
========================================================
Welcome to the grading system! Please enter your option:
1: input record
2: query a student
3: list grades in a group
4: get highest grade in a group
5: list all group names
6: exit
========================================================
Option: 4
Please input the group name: FSA
The highest scores in  FSA : 85

========================================================
Welcome to the grading system! Please enter your option:
1: input record
2: query a student
3: list grades in a group
4: get highest grade in a group
5: list all group names
6: exit
========================================================
Option: 5
['FSB', 'FSC', 'FSA']
```

The following code snippet shows how the main part of the program can be coded.

```
###########################################################
grades_dict = {  #preload some records into the dictionary
    ('FSA', 1): [80,63],
    ('FSA', 2): [75,81],
    ('FSA', 3): [85,60],
    ('FSB', 1): [65,70],
    ('FSB', 2): [60,73],
    ('FSC', 1): [80,90]
}

#### Define available options in the grading system: ######
INPUT = 1
QUERY = 2
LIST = 3
MAX = 4
LISTALLGROUPS = 5
EXIT = 6

###########################################################
option = INPUT     # default option
while option != EXIT:
     showMenu()      # display the selection menu
     option = int(input("option: "))   # get option
     if option == INPUT:
          :
```

Self-Practice Exercises - B

The following are prototypes (i.e. the 'definition') of the functions together with the parameters(s) that are to be used in your program.

i. **`inputRecord(dictionary, group, id, score)`** to input the record into your program, where `dictionary` is the dictionary implemented for the data structure, `group` is a string representing a group name, `id` is a student's id number (positive integers ranging from 1 to 40), and `score` is the grade of the student.

ii. **`query(dictionary, group, id)`** to get the score of a student in a lab group. (Use `try/except` to check for valid/invalid group-id selection.)

iii. **`listGrades(dictionary, group)`** to get all the students' grades in a group. (Extract the individual grade tuple using the `items` method, and combine them into a list by using the `append` method. An alternative is to use a one-line list comprehension.)

iv. **`maxGrade(dictionary, group)`** to get the highest grade in a group.
(Extract individual marks into a list, change the 2-element list to a single-element list using the `sum()` function, then apply the `max()` function.)

v. **`listGroups(dictionary)`** to list all the group in the system.
(Use `key[0]` to extract all the groups available in the dictionary, apply `set()` function to remove duplicate groups. Apply `sort` method to sort them in alphabetical order.)

Self-Practice Exercises - B

Ex 2. You can also group a collection of dictionaries into one single dictionary to form a nested dictionary. The following table shows the used car prices based on their brands, models and years of registration.

Entry	Brand	Model	Year	Price
1	Honda	Civic	2010	35000
2	Toyota	Corolla	2015	70000
3	VW	Golf	2012	45000
4	Honda	City	2013	36000
5	Toyota	Corolla	2012	45000
6	VW	Jetta	2013	62000
7	Honda	Civic	2015	56000
8	Toyota	Vios	2017	63000
9	VW	Jetta	2011	40000
10	Toyota	Vios	2013	42000
11	VW	Golf	2014	65000
12	Hyundai	Elantra	2012	35000

A nested dictionary such as the following can then be used to store these information as a 'database'.

```
cars={1:{'brand':'Honda','model':'Civic','year':2010,'price':50000},
      2:{'brand':'Toyota','model':'Corolla','year':2015,'price':70000},
      3:{'brand':'VW','model':'Golf', 'year':2012,'price':45000}}
```

By doing so, you will be able to query the 'database' based on keys used in the nested dictionary.

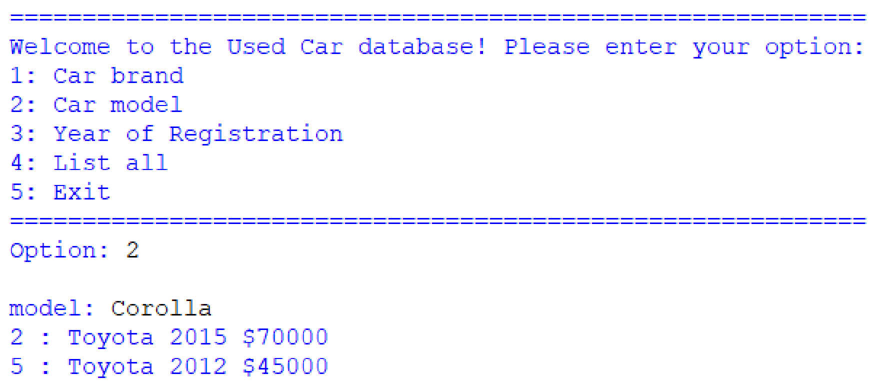
Code a program using the nested dictionary and function that allows user to query for the car availability based on the different keys. You should try using the `items()` method to extract the dictionary key:value information, such as:

```
        for car_id, car_info in cars.items():
```

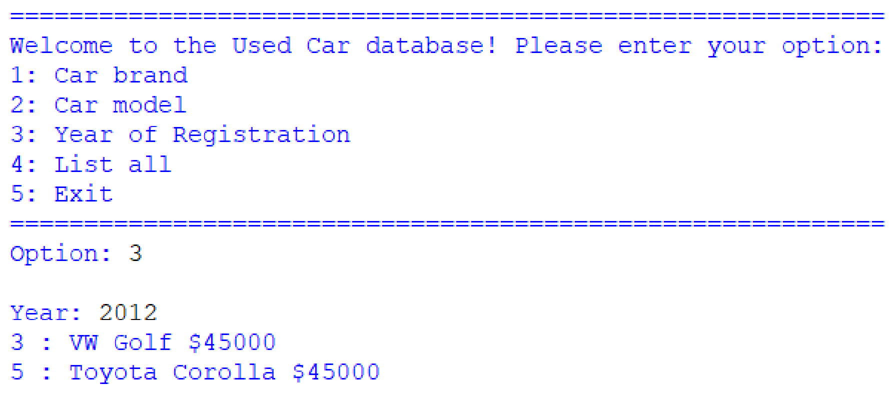
The followings are screenshot of the program output with a partially filled database with entries for the table shown above.

```
===========================================================
Welcome to the Used Car database! Please enter your option:
1: Car brand
2: Car model
3: Year of Registration
4: List all
5: Exit
===========================================================
Option: 1

Car brand: Toyota
2 : Corolla 2015 $70000
5 : Corolla 2012 $45000
```

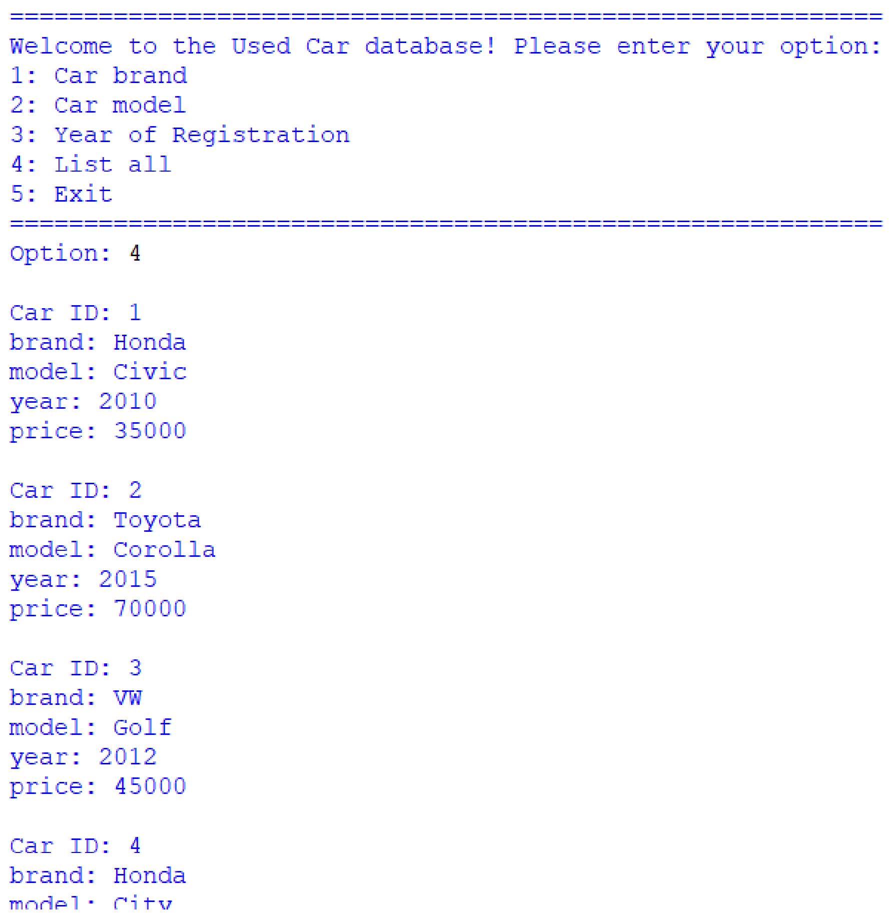
Self-Practice Exercises - B



```
===========================================================
Welcome to the Used Car database! Please enter your option:
1: Car brand
2: Car model
3: Year of Registration
4: List all
5: Exit
===========================================================
Option: 2

model: Corolla
2 : Toyota 2015 $70000
5 : Toyota 2012 $45000
```



```
===========================================================
Welcome to the Used Car database! Please enter your option:
1: Car brand
2: Car model
3: Year of Registration
4: List all
5: Exit
===========================================================
Option: 3

Year: 2012
3 : VW Golf $45000
5 : Toyota Corolla $45000
```



```
===========================================================
Welcome to the Used Car database! Please enter your option:
1: Car brand
2: Car model
3: Year of Registration
4: List all
5: Exit
===========================================================
Option: 4

Car ID: 1
brand: Honda
model: Civic
year: 2010
price: 35000

Car ID: 2
brand: Toyota
model: Corolla
year: 2015
price: 70000

Car ID: 3
brand: VW
model: Golf
year: 2012
price: 45000

Car ID: 4
brand: Honda
model: City
```