Data Structures

Solve the following exercises and upload your solutions to Moodle until the specified due date. Make sure to use the *exact filenames* that are specified for each individual exercise. Unless explicitly stated otherwise, you can assume correct user input and correct arguments.

Exercise 1 – Submission: `ex1.py` 10 Points

Write a program that reads a string as input. This string is the processed character by character. Every lowercase character is added to a list, every uppercase letter is added to another list and all remaining/other characters are also added to yet another list. In addition, collect all unique characters in a set. Print all data structures afterwards-

Example input:

```
Enter string: This 12 IS an ExamPLE SENTEnce.
```

Example output (order of `unique` might differ):

```
lowercase: ['h', 'i', 's', 'a', 'n', 'x', 'a', 'm', 'n', 'c', 'e']
uppercase: ['T', 'I', 'S', 'E', 'P', 'L', 'E', 'S', 'E', 'N', 'T', 'E']
other: [' ', '1', '2', ' ', ' ', ' ', ' ', ' ', '.']
unique: {'e', 'n', 'i', 'N', '2', 'c', 'E', 'h', 'a', 'I', 'x', '.', 'S',
         's', 'm', 'L', 'T', ' ', '1', 'P'}
```

Exercise 2 – Submission: `ex2.py` 10 Points

You are given the following list of filenames:

```
fnames = ["file7.txt", "file1.png", "file3.txt", "file2.txt",
          "file7.txt", "file1.txt", "file3.txt", "file4.png",
          "file4.png", "file5.txt", "file0.txt", "file7.dat"]
```

Write a program that drops all duplicate entries and only collects files ending with `".txt"` into a sorted list. Print this list afterwards.

Output:

```
['file0.txt', 'file1.txt', 'file2.txt', 'file3.txt', 'file5.txt', 'file7.txt']
```

Exercise 3 – Submission: `ex3.py` 10 Points

Write a program that counts characters in a user specified string using a dictionary. Before counting, all characters must be transformed to lowercase. Print the dictionary afterwards.

Example input:

```
Enter string: This 12 IS an ExamPLE SENTEnce.
```

Example output (order might differ):

```
{'t': 2, 'h': 1, 'i': 2, 's': 3, ' ': 5, '1': 1, '2': 1, 'a': 2, 'n': 3,
 'e': 5, 'x': 1, 'm': 1, 'p': 1, 'l': 1, 'c': 1, '.': 1}
```

Exercise 4 – Submission: `ex4.py` 35 Points

Write a program that calculates the row sums, the column sums and the total sum of a 2D matrix that was entered by the user. The program should work as follows:

- The 2D matrix should be implemented as a nested list of integer numbers.
- To extract the matrix from the console input, the following actions must be performed:
  - Until `"x"` is entered, the user can enter entire rows.
  - Such an entire row input must follow the format `"int_1 int_2 ... int_n"`, where `int_i` are integer numbers separated by a single space character (you can assume correct user input w.r.t. the data types).
  - The individual integers must then be extracted and stored in a (row) list.
  - If the user enters rows with different sizes, extend all shorter rows with 0, so that all rows are equally sized afterwards, i.e., apply 0-padding to the end of shorter rows.
  - All row lists must then be collected inside another list, which will then be the matrix/nested list.
- Using the matrix, calculate the row sum, i.e., for each row, compute the sum and store the result in a list (row sum list).
- Using the matrix, calculate the column sum, i.e., for each column, compute the sum and store the result in a list (column sum list).
- Using the matrix, calculate the total sum of all elements.
- Print a nicely formatted matrix of the form

  ```
  [[r1c1 r1c2 ... r1cn]
           ...
   [rmc1 rmc2 ... rmcn]]
  ```

  where `ricj` indicates the element in the i-th row and j-th column.
- Print the row sums, the column sums and the total sum.

Example input:

```
Enter row: 1 2 3 4
Enter row: 5
Enter row: 0
Enter row: x
```

Example output:

```
[[1 2 3 4]
 [5 0 0 0]
 [0 0 0 0]]
row sums: [10, 5, 0]
column sums: [6, 2, 3, 4]
total sum: 15
```

Exercise 5 – Submission: `ex5.py` 35 Points

In an earlier exercise, we already talked about powerlifting and the three weight lifts: squat, bench press and deadlift. Write a program that can manage lifters and their weights as follows (see the example below for more details):

- Create a console user interface as shown in the example program execution below. The available actions are `"a"` to add a new lifter, `"r"` to remove an existing lifter, `"u"` to update an existing lifter, `"v"` to view all lifters and `"x"` to exit the program. All other input is invalid and leads to an error message (see example).
- The add action `"a"` must do the following:
  - The user must enter a new lifter name.
  - A corresponding entry is created in the dictionary, which then maps to a further dictionary with three default entries that all contain empty lists: `"squat"`, `"bench press"`, `"deadlift"`.
  - If the dictionary already contains the name, an error message is displayed and the action is aborted.
- The remove action `"r"` must do the following:
  - The user must enter an existing lifter name.
  - If the dictionary contains the name, the entry is removed. Otherwise, an error message is displayed and the action is aborted.
- The add action `"u"` must do the following:
  - The user must enter an existing lifter name.
  - If the name does not exist, an error message is displayed and the action is aborted.
  - The user must then enter which lift to update (must be one of `"squat"`, `"bench press"`, `"deadlift"`), followed by float numbers separated by single space characters indicating the lifted weights (you can assume correct user input in both cases).
  - The corresponding list must then be extended by this entered weight (e.g., if the user entered `"squat"` and then `"120.5 130 140"`, this lifter's squat list must be extended with `[120.5, 130, 140]`).
- The view action `"v"` should display all lifters and all their weights for each of the three lifts.

Example program execution:

```
Welcome to Powerlifting Data Collector!

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: y
Invalid action 'y'. Try again!
```

```
Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: v

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: a
Enter new lifter name: Jake

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: a
Enter new lifter name: Jake
Lifter 'Jake' already exists!

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: a
Enter new lifter name: Tina

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: v
------------------------------
Name: Jake
squat: []
bench press: []
deadlift: []
------------------------------
Name: Tina
squat: []
bench press: []
deadlift: []
```

```
Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: u
Enter lifter name to update: TINA
Lifter 'TINA' does not exist!

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: u
Enter lifter name to update: Tina
Enter lift (one of 'squat', 'bench press', 'deadlift'): squat
Enter weight(s): 120.5 130 140

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: u
Enter lifter name to update: Tina
Enter lift (one of 'squat', 'bench press', 'deadlift'): deadlift
Enter weight(s): 195.5

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: v
------------------------------
Name: Jake
squat: []
bench press: []
deadlift: []
------------------------------
Name: Tina
squat: [120.5, 130.0, 140.0]
bench press: []
deadlift: [195.5]

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
```

```
x - Exit the program
Enter action: u
Enter lifter name to update: Tina
Enter lift (one of 'squat', 'bench press', 'deadlift'): deadlift
Enter weight(s): 200 201

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: v
------------------------------
Name: Jake
squat: []
bench press: []
deadlift: []
------------------------------
Name: Tina
squat: [120.5, 130.0, 140.0]
bench press: []
deadlift: [195.5, 200.0, 201.0]

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: r
Enter lifter name to remove: TEMP
Lifter 'TEMP' does not exist!

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: r
Enter lifter name to remove: Jake

Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: v
------------------------------
Name: Tina
squat: [120.5, 130.0, 140.0]
bench press: []
deadlift: [195.5, 200.0, 201.0]
```

```
Available actions:
a - Add lifter
r - Remove lifter
u - Update lifter
v - View lifters
x - Exit the program
Enter action: x
Bye!
```