L06 Tasks

L06-T1: Reversed list
L06-T2: Removing duplicate elements from a list
L06-T3: Menu-based program for shopping list processing
L06-T4: Matrix
L06-T5: Matrix 2

Submission:

- Submit L06-T3, L06-T4, and L06-T5 to CodeGrade via Moodle before the deadline

Note:

- Be especially careful with spaces so that your output follows the sample output. Note that also each input-string in the program is ending with '\n'. The reason for this is that it makes the output in CodeGrade more readable.

- If you see some "Code Structure Tests" hidden in CodeGrade but they are not mentioned in the task description, you don't need to worry about them. They are just there to make sure the code is ok.

L06-T1: Reversed list

Write a function that reverses a list of integers, **without** using the reverse() method or slicing.

In the main program ask for integers using the function `input_integers()` given in the lecture. You can find this function in the file “L6E4.py”.

Example run 1:

```
Give integers separated by comma:
1,2,3,4,5
Reversed list: [5, 4, 3, 2, 1]
```

L06-T2: Removing duplicate elements from a list

Implement a function that takes a list of integers and returns the same list with all duplicate elements removed, preserving the **original** order. Print the original list and the new list to the screen.

In the main program ask for integers using the function `input_integers()` given in the lecture. You can find this function in the file “L6E4.py”.

Example run 1:

```
Give integers separated by comma:
3,2,3,1,6,2,3,4,5
Original List: [3, 2, 3, 1, 6, 2, 3, 4, 5]
List with duplicates removed: [3, 2, 1, 6, 4, 5]
```

Example run 2:

```
Give integers separated by comma:
3,2,3,1,6,2,3,4,5
Original List: [3, 2, 3, 1, 6, 2, 3, 4, 5]
List with duplicates removed: [3, 2, 1, 6, 4, 5]
```

Example run 3:

```
Give integers separated by comma:
1,4,3,1,5,7,4,-4,-12,-12,-1,-4,-4
Original List: [1, 4, 3, 1, 5, 7, 4, -4, -12, -12, -1, -4, -4]
List with duplicates removed: [1, 4, 3, 5, 7, -4, -12, -1]
```

L06-T3: Menu-based program for shopping list processing

(Submit this task to CodeGrade on Moodle)

Write a program that maintains a shopping list. You can add products to the shopping list and remove products from it.

The program should work in such a way that the user enters information in the menu on whether he wants to add or remove a product or stop.

- If the user wants to add something, the program asks for the name of the product and adds it to the list.
- If the user wants to delete something, the program asks for the location of the product in the list.
- The "End" selection tells the user what he is going to buy and ends the program.

The implementation of the shopping list is done as a list. The program should print the list after each operation.

Perform the menu handling as a separate function.

If the selection is not recognized, inform the user with the notification “Unknown selection”

Example run 1:

```
Your shopping list contains the following products:
[]
You can choose from the functions below:
1) Add
2) Remove
0) End
Your choice:
1
Enter the product to be added:
Milk

Your shopping list contains the following products:
['Milk']
You can choose from the functions below:
1) Add
2) Remove
0) End
Your choice:
1
Enter the product to be added:
Tea

Your shopping list contains the following products:
['Milk', 'Tea']
You can choose from the functions below:
1) Add
2) Remove
0) End
Your choice:
5
Unknown selection.

Your shopping list contains the following products:
['Milk', 'Tea']
You can choose from the functions below:
1) Add
2) Remove
0) End
Your choice:
2
You have 2 items in your shopping list.
Enter the location of the product to be removed:
1

Your shopping list contains the following products:
['Tea']
```

```
You can choose from the functions below:
1) Add
2) Remove
0) End
Your choice:
0
You are going to buy the following products:
['Tea']
```

L06-T4: Matrix

(Submit this task to CodeGrade on Moodle)

Write a function `create_matrix(rows, cols)` that takes the number of rows and columns as input and prompts the user to enter the elements row by row separated by whitespaces. Validates the number of elements in each row. If the number of elements is not correct, ask the elements again. The function **returns** the resulting matrix.

Note that when you use the `split()` method in Python without providing any parameter, it will split the string using whitespace characters (spaces, tabs, and newlines) as the default delimiter.

After that write a function `print_matrix(matrix)` that prints `matrix` **row by row.** Separate element by using tabulator. See example run below. Note that the method `` `print` `` does not number of rows and columns as parameters, but the method can print any matrix. This function **does not return** anything to the main program.

Ask the number of rows and columns in the main program.

Note: make sure that your code is having these functions: `create_matrix(rows, cols)` and `print_matrix(matrix)` with **exact names and parameters**. This will be tested in the "Code structure tests" in CodeGrade. If these structure tests fail, the remaining tests will not be graded.

Example run 1:

```
Enter the number of rows:
2
Enter the number of columns:
3
Give row 1:
4 5
Error: Invalid number of elements in the row. Please try again.
Give row 1:
1 2 3
Give row 2:
4 5 6
|1   2    3|
|4   5    6|
```

Example run 2:

```
Enter the number of rows:
3
Enter the number of columns:
2
Give row 1:
1
Error: Invalid number of elements in the row. Please try again.
Give row 1:
1 2
Give row 2:
3 4
Give row 3:
5 6

|1      2|

|3      4|

|5      6|
```

L06-T5: Matrix 2

(Submit this task to CodeGrade on Moodle)

For a given matrix M with dimensions m×n (m rows and n columns), its transpose, denoted as MT, will have dimensions n×m (n rows and m columns). The element at row i and column j in the original matrix becomes the element at row j and column i in the transposed matrix. Write a function **transpose(matrix)** which takes a matrix as input and **returns** its transpose. Hint: Generate first a matrix of the correct size by filling it with zeros. Then add the correct elements.

Note: make sure that your code is having these functions: transpose(matrix) with **exact names and parameters**. This will be tested in the "Code structure tests" in CodeGrade. If these structure tests fail, the remaining tests will not be graded**.**

Extend your solution to Task 4 so that after the original matrix is printed, the transpose(matrix) function is called. After that, the transpose is printed. See example below.

Example run 1:

```
Enter the number of rows:
3
Enter the number of columns:
2
Give row 1:
2 4
Give row 2:
3 6
Give row 3:
4 8
```

```
The original matrix:
|2      4|
|3      6|
|4      8|
Its transpose:
|2      3       4|
|4      6       8|
```

Example run 2:

```
Enter the number of rows:
2
Enter the number of columns:
4
Give row 1:
1
Error: Invalid number of elements in the row. Please try again.
Give row 1:
1 2 3 4
Give row 2:
5 6 7 8
The original matrix:
|1      2       3       4|
|5      6       7       8|
Its transpose:
|1      5|
|2      6|
|3      7|
|4      8|
```