

Assignment 6 – Lists and Tuples

Lists

A list, also known as array is a collection of items known as elements.

Creating a List

1. Create a new python file.
2. Code the following:

```
1  #!/usr/bin/env python3
2
3  # create an empty list
4  items = []
5
6  # create a list with repeating values
7  counts = [0] * 3      # same as [0, 0, 0]
8
9  # create a list with items loaded
10 cars = ["Porsche", "Ford", "Mercedes"]
11
```

Retrieving Elements from a List

3. Modify as follows:
 - a. Lists use indexes to access element positions in the list and since these lists are zero based (i.e., the first position of the list is zero), accessing the cars list with an index 1 will return the second item in the list.
4. Test

```
9  # create a list with items loaded
10 cars = ["Porsche", "Ford", "Mercedes"]
11 print("The second car in the list is a " + cars[1])
12
```

Run: blackjack blackjack test

C:\Users\Saddleback\AppData\Local\Programs\Python\Python38\python.exe

The second car in the list is a Ford

Process finished with exit code 0

Replacing Elements in a List

5. Modify to update the second car in the list
6. Test again.

```

9      # create a list with items loaded
10     cars = ["Porsche", "Ford", "Mercedes"]
11     print("The second car in the list is a " + cars[1])
12
13     # update the second car in the list
14     cars[1] = "BMW"
15     print("After update, the second car in the list is now a "
16           + cars[1])
17
Run: test (1) x
C:\Users\Saddleback\PycharmProjects\Final\venv\Scripts\python.
The second car in the list is a Ford
After update, the second car in the list is now a BMW
Process finished with exit code 0

```

Append an Element to the End of a List

7. Modify as follows to *append* (add to the end) a new car to the list.
 - a. Note the *len()* function which returns the length (number of elements) in the list. When using this as an index, we must subtract 1 since the index is zero based.
8. Test once again...

```

15     print("After update, the second car in the list is now a "
16           + cars[1])
17
18     # add a new car to the end of the list
19     cars.append("Lexus")
20     print("A new car was added to the end of the list: "
21           + cars[len(cars)-1])
22
Run: lists x
C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\py
The second car in the list is a Ford
After update, the second car in the list is now a BMW
A new car was added to the end of the list: Lexus
Process finished with exit code 0

```

Insert an Element to the middle of a List

9. Modify as follows to *insert* a new car into the 2nd position in the list.
10. Test once again...

```

21         + cars[len(cars)-1])
22
23     # insert a new car into the 2nd position of the list
24     cars.insert(1, "Honda")
25     print("After insert, the second car in the list is now a "
26           + cars[1])
27
28

```

Run: movie-1 x movie-1 x

```

C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe "C:/Users/Saddleback/AppData/Local/Programs/Python/Python36/python.exe"
The second car in the list is a Ford
After update, the second car in the list is now a BMW
A new car was added to the end of the list: Lexus
After insert, the second car in the list is now a Honda

Process finished with exit code 0

```

Remove an Elements from a List

By Value

11. Modify as follows to remove the Porsche from the list:
12. Test to show which car is now the second car in the list.

```

25     print("After insert, the second car in the list is now a "
26           + cars[1])
27
28     # remove a car from the list = by value
29     cars.remove("Porsche")
30     print("The 1st element was deleted so the second car in the list is now a "
31           + cars[1])
32

```

Run: movie-1 x movie-1 x

```

C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe "C:/Users/Saddleback/AppData/Local/Programs/Python/Python36/python.exe"
The second car in the list is a Ford
After update, the second car in the list is now a BMW
A new car was added to the end of the list: Lexus
After insert, the second car in the list is now a Honda
The 1st element was deleted so the second car in the list is now a BMW

Process finished with exit code 0

```

By Index

13. Modify as follows to remove a car by index:

14. Test...

```

30 print("The 1st element was deleted so the second car in the list is now
31     + cars[1])
32
33 # remove a car from the list = by index
34 cars.pop(1)
35 print("The car in the second position was deleted...")
36 print("The car in the second position is now a " + cars[1])
37

```

Run: movie-1 x movie-1 x

C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe "C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe"

The second car in the list is a Ford
 After update, the second car in the list is now a BMW
 A new car was added to the end of the list: Lexus
 After insert, the second car in the list is now a Honda
 The 1st element was deleted so the second car in the list is now a BMW
 The car in the second position was deleted...
 The car in the second position is now a Mercedes

Process finished with exit code 0

Processing Items in a List*Search to See if Item is in List*

15. Modify as follows to test to see if a car is in the list.

16. Test:

```

36 print("The car in the second position is now a " + cars[1])
37
38 print()
39
40 # process to find a car in the list
41 car = "Honda"
42 if car in cars:
43     print("There is a " + car + " in the list")
44

```

Run: movie-1 x movie-1 x

C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe "C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe"

The second car in the list is a Ford
 After update, the second car in the list is now a BMW
 A new car was added to the end of the list: Lexus
 After insert, the second car in the list is now a Honda
 The 1st element was deleted so the second car in the list is now a BMW
 The car in the second position was deleted...
 The car in the second position is now a Mercedes

There is a Honda in the list

Process finished with exit code 0

Print Entire List – For Loop

17. Modify as follows to display the cars in the list with a *for* loop:

18. Test...

```
43     print("There is a " + car + " in the list")
44
45     print()
46     print("The list contains " + str(len(cars)) + " items in the list")
47
48     # process to display all cars in the list
49     print("Processing with a for loop")
50     for car in cars:
51         print("\t",car)
52
```

Run: movie-1 x movie-1 x

C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe "C:~
The second car in the list is a Ford
After update, the second car in the list is now a BMW
A new car was added to the end of the list: Lexus
After insert, the second car in the list is now a Honda
The 1st element was deleted so the second car in the list is now a BMW
The car in the second position was deleted...
The car in the second position is now a Mercedes

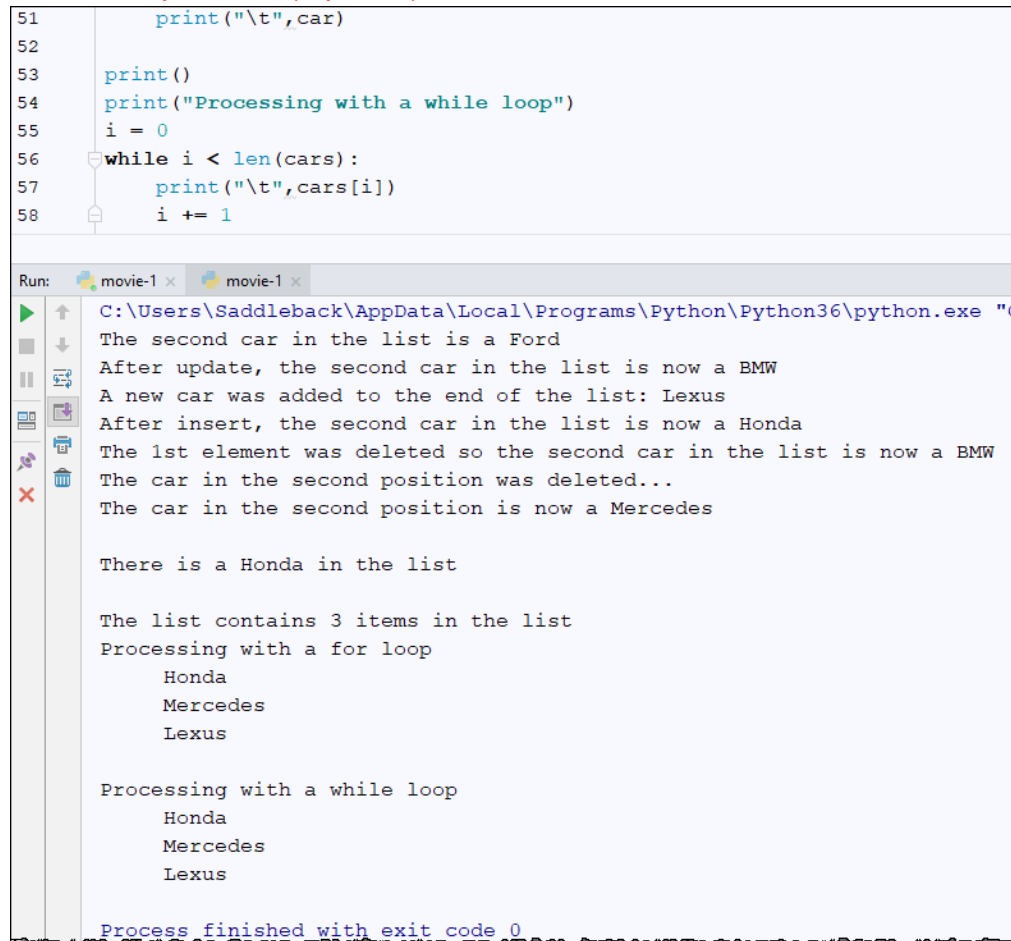
There is a Honda in the list

The list contains 3 items in the list
Processing with a for loop
 Honda
 Mercedes
 Lexus

Process finished with exit code 0

Print Entire List – While Loop

19. Modify as follows to add processing the list with a *while* loop.
20. Run one final time:
 - a. Make sure your screen capture includes all the print lines.

Screen Capture #1 (4 points)

The screenshot shows a Python IDE with a code editor and a run console. The code in the editor is as follows:

```
51     print("\t",car)
52
53     print()
54     print("Processing with a while loop")
55     i = 0
56     while i < len(cars):
57         print("\t",cars[i])
58         i += 1
```

The run console shows the following output:

```
Run: movie-1 x movie-1 x
C:\Users\Saddleback\AppData\Local\Programs\Python\Python36\python.exe "
The second car in the list is a Ford
After update, the second car in the list is now a BMW
A new car was added to the end of the list: Lexus
After insert, the second car in the list is now a Honda
The 1st element was deleted so the second car in the list is now a BMW
The car in the second position was deleted...
The car in the second position is now a Mercedes

There is a Honda in the list

The list contains 3 items in the list
Processing with a for loop
    Honda
    Mercedes
    Lexus

Processing with a while loop
    Honda
    Mercedes
    Lexus

Process finished with exit code 0
```

The Movie List Program

In this application, we'll create a list of movies that we can display, add to and delete from.

We will start by breaking the application into pieces. Make sure to save this file as we will continue to enhance this Movie List application again in this and future assignments.

Create a new python file named **movie-1.py**. (Make sure you attach this file when submitting your assignment.)

Code Validation – movie-1.py (3 points)

Menu

First, we will create a function for the menu. The Menu should look something like this:

```
C:\Users\Saddleback\AppData\Local\B
COMMAND MENU
list - List all movies
add  - Add a movie
del  - Delete a movie
exit - Exit program

Process finished with exit code 0
```

```
movie-1.py x
1  #!/usr/bin/env python3
2
3
4  def display_menu():
5      print("The Movie List program")
6      print()
7      print("COMMAND MENU")
8      print("list - List all movies")
9      print("add  - Add a movie")
10     print("del  - Delete a movie")
11     print("exit - Exit program")
12     print()
13
```

main()

Now we will create the main function and call the menu function to display the menu items. Don't forget to add the if statement that calls the main() function.

```

12     print()
13
14
15     def main():
16         display_menu()
17
18
19     if __name__ == "__main__":
20         main()
21

```

The List

We'll be using a list that is created in the main and then passed between the functions. Go ahead and create a list for the movie titles and add a few titles so we have something in the list.

```

15     def main():
16         # Create and Initialize the movie list
17         movie_list = ["Deadpool",
18                       "Casino Royale",
19                       "Goodfellas"]
20
21         display_menu()
22

```

Display Movie List

Add a function to display the movie list.

```

12     print()
13
14
15     def list_movies(movie_list):
16         i = 1
17         for movie in movie_list:
18             print(str(i) + ". " + movie)
19             i += 1
20     print()
21

```


Code the Menu Options

In the main, after the menu is displayed, we will need to prompt for the user selection from the menu and process that selection. Additionally, this should be a loop until the user selects the exits.

```
29     display_menu()
30
31     while True:
32         command = input("Command: ")
33         if command.lower() == "list":
34             list_movies(movie_list)
35         elif command.lower() == "exit":
36             break
37         else:
38             print("Not a valid command. Please try again.\n")
39
40     print("Bye!")
41
42
43 if __name__ == "__main__":
```

*Test "List all movies"***Screen Capture #2 (1 point)**

```
C:\Users\Saddleback\AppData
The Movie List program

COMMAND MENU
list - List all movies
add  - Add a movie
del  - Delete a movie
exit - Exit program

Command: list
1. Deadpool
2. Casino Royale
3. Goodfellas
```

Add Movie to the List

Add a function to Add a movie to the list. In the add function, you will need to first prompt for a movie title, append it to the list, and finally display that the movie was added.

```

20     print()
21
22
23 def add(movie_list):
24     movie = input("Movie: ")
25     movie_list.append(movie)
26     print(movie + " was added.\n")
27
28
29 def main():
30     # Create and Initialize the menu

```

You will also need to add the menu option code to call the new add function.

```

39     if command.lower() == "list":
40         list_movies(movie_list)
41     elif command.lower() == "add":
42         add(movie_list)
43     elif command.lower() == "exit":
44         break

```

*Test "Add a movie"***Screen Capture #3 (1 point)**

```

C:\Users\Saddleback\AppData
The Movie List program

COMMAND MENU
list - List all movies
add  - Add a movie
del  - Delete a movie
exit - Exit program

Command: list
1. Deadpool
2. Casino Royale
3. Goodfellas

Command: add
Movie: Monsters, Inc.
Monsters, Inc. was added.

Command: list
1. Deadpool
2. Casino Royale
3. Goodfellas
4. Monsters, Inc.

```

Delete Movie from the List – By Index

Add a function to Delete a movie from the list. In the delete function, you will need to first prompt for a movie number, validate that the number is in range, delete the movie and finally prompt the user to let them know the movie was deleted.

```

26     print(movie + " was added.\n")
27
28
29     def delete(movie_list):
30         number = int(input("Number: "))
31         if number < 1 or number > len(movie_list):
32             print("Invalid movie number.\n")
33         else:
34             movie = movie_list.pop(number - 1)
35             print(movie + " was deleted.\n")
36
37
38     def main():

```

Again, you will also need to add the menu option code to call the new delete movie function.

```

50     elif command.lower() == "add":
51         add(movie_list)
52     elif command.lower() == "del":
53         delete(movie_list)
54     elif command.lower() == "exit":

```

*Test "Delete a movie" - Number***Screen Capture #4 (1 point)**

```

C:\Users\Saddleback\AppData\Local\Microsoft\Windows\CurrentVersion\Exe\cmd.exe
The Movie List program

COMMAND MENU
list - List all movies
add  - Add a movie
del  - Delete a movie
exit - Exit program

Command: list
1. Deadpool
2. Casino Royale
3. Goodfellas

Command: del
Number: 4
Invalid movie number.

Command: del
Number: 2
Casino Royale was deleted.

Command: list
1. Deadpool
2. Goodfellas

```

Lists of Lists

A list is a simple array of elements, a list of lists is two dimensional, like a spreadsheet with rows and columns.

Creating a List of Lists

21. Create a python file.
22. Code the following:

```
1  #!/usr/bin/env python3
2
3  movies = [
4      ["Deadpool", "2016", "8.0"],
5      ["Casino Royale", "2006", "8.1"],
6      ["Star Wars: A New Hope", "1977", "8.6"]
7  ]
8
9  movie = ["Monsters, Inc", "2001", "8.1"]
```

Adding to the List of Lists

23. Code the following to add a movie to the movies list.

```
1  #!/usr/bin/env python3
2
3  movies = [
4      ["Deadpool", "2016", "8.0"],
5      ["Casino Royale", "2006", "8.1"],
6      ["Star Wars: A New Hope", "1977", "8.6"]
7  ]
8
9  movie = ["Monsters, Inc", "2001", "8.1"]
10
11 movies.append(movie)
```

Accessing Items in the List of Lists

24. Code the following to display some elements from the list:

```
9  movies.append(movie)
10
11 # The first number is the row,
12 # the second number is the column
13 print("Item in movies position [0][0]: " + movies[0][0])
14 print("Item in movies position [3][2]: " + movies[3][2])
15
```

Run: more_lists x

```
C:\Users\Saddleback\PycharmProjects\Final\venv\Scripts\python.exe
Item in movies position [0][0]: Deadpool
Item in movies position [3][2]: 8.1

Process finished with exit code 0
```

Processing Items in the List of Lists

25. Code the following to display all the elements in the movies list:

Screen Capture #5 (1 points)

```

1  #!/usr/bin/env python3
2
3  movies = [["Deadpool", "2016", "8.0"],
4            ["Casino Royale", "2006", "8.1"],
5            ["Star Wars: A New Hope", "1977", "8.6"]]
6
7  movie = ["Monsters, Inc", "2001", "8.1"]
8
9  movies.append(movie)
10
11 # The first number is the row,
12 # the second number is the column
13 print("Item in movies position [0][0]: " + movies[0][0])
14 print("Item in movies position [3][2]: " + movies[3][2])
15
16 print()
17 # Display all elements in the movies list
18 # We will need to use a nest for loop to do this
19
20 # The first loop goes through the row
21 for movie in movies:
22     # The second loop goes through the columns
23     for item in movie:
24         print(item, end=" | ")
25     print()
26

```

26. Run one final time:

- Make sure your screen capture includes all the print lines.

Screen Capture #6 (1 points)

```

C:\Users\Saddleback\AppData\Local\Programs\Python\Python38-32\python.exe
Item in movies position [0][0]: Deadpool
Item in movies position [3][2]: 8.1

Deadpool | 2016 | 8.0 |
Casino Royale | 2006 | 8.1 |
Star Wars: A New Hope | 1977 | 8.6 |
Monsters, Inc | 2001 | 8.1 |

```

The Movie List Program (Continued)

In this version, we will update the movies application (movie-1.py) to use 2D lists to include the year the movie was released.

The functions we'll need to modify will include:

- Main – We will need to update the movie list to be 2 dimensional and add the years for the movie.
- List – We will need to be able to display both the name and year in the output.
- Add – We will need to input both the movie name and year.
- Delete – Update to get the movie title being deleted based off a two-dimension list.

Start by making a copy of the movie-1.py file and naming it **movie-2.py**. Make sure you include this file when submitting the assignment.

Code Validation – movie-2.py (3 points)

Update the movies list to be 2D:

```
def main():  
    movie_list = [ ["Deadpool", 2016],  
                   ["Casino Royale", 2006],  
                   ["Goodfellas", 1990]]
```

Update the list_movies to display the 2D list:

```
def list_movies(movie_list):  
    if len(movie_list) == 0:  
        print("There are no movies in the list.\n")  
        return  
    else:  
        i = 1  
        for row in movie_list:  
            print(str(i) + ". " + row[0] + " (" + str(row[1]) + ")")  
            i += 1  
        print()
```

Update the add to be able to add both a movie title and year to the 2D list:

```
def add(movie_list):
    name = input("Name: ")
    year = input("Year: ")
    movie = (name, year)
    movie_list.append(movie)
    print(movie[0] + " was added.\n")
```

Update the delete to be able to delete a movie from the 2D list:

```
def delete(movie_list):
    number = int(input("Number: "))
    if number < 1 or number > len(movie_list):
        print("Invalid movie number.\n")
    else:
        movie = movie_list.pop(number-1)
        print(movie[0] + " was deleted.\n")
```

Add a Movie

Screen Capture #7 (1 point)

```
C:\Users\Saddleback\AppData\Local\Programs\Python\Python38-32\python.exe
The Movie List program

COMMAND MENU
list - List all movies
add  - Add a movie
del  - Delete a movie
exit - Exit program

Command: list
1. Deadpool (2016)
2. Casino Royale (2006)
3. Goodfellas (1990)

Command: add
Name: Monsters, Inc.
Year: 2001
Monsters, Inc. was added.

Command: list
1. Deadpool (2016)
2. Casino Royale (2006)
3. Goodfellas (1990)
4. Monsters, Inc. (2001)

Command:
```

Delete a Movie

Screen Capture #8 (1 point)

```

Command: list
1. Deadpool (2016)
2. Casino Royale (2006)
3. Goodfellas (1990)
4. Monsters, Inc. (2001)

Command: del
Number: 3
Goodfellas was deleted.

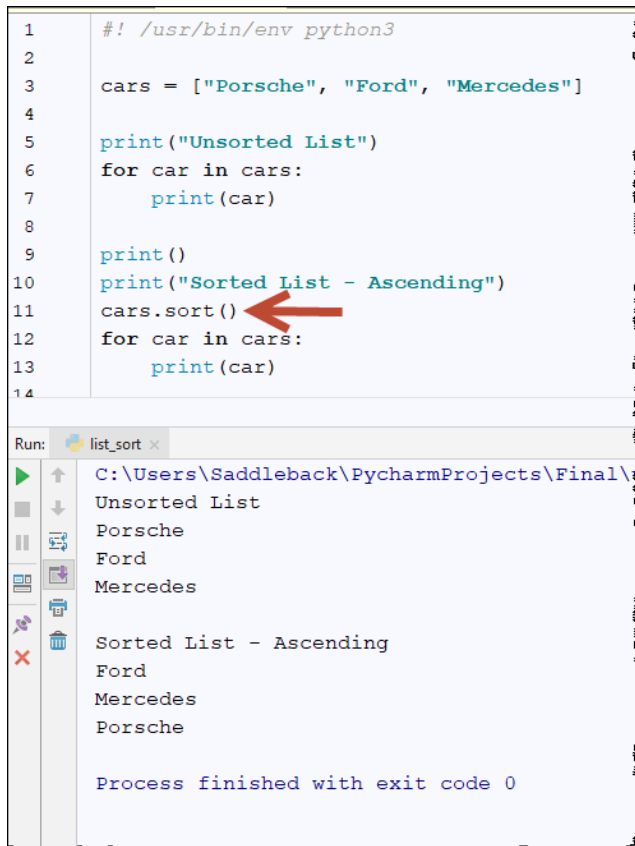
Command: list
1. Deadpool (2016)
2. Casino Royale (2006)
3. Monsters, Inc. (2001)

Command:

```

Sorting a List - Ascending Order

27. Create a python file named **list_sorting.py**.
28. Code the following list and sort in ascending order.
29. Test:



```

1  #!/usr/bin/env python3
2
3  cars = ["Porsche", "Ford", "Mercedes"]
4
5  print("Unsorted List")
6  for car in cars:
7      print(car)
8
9  print()
10 print("Sorted List - Ascending")
11 cars.sort()
12 for car in cars:
13     print(car)
14

```

Run: list_sort x

```

C:\Users\Saddleback\PycharmProjects\Final\
Unsorted List
Porsche
Ford
Mercedes

Sorted List - Ascending
Ford
Mercedes
Porsche

Process finished with exit code 0


```


Sorting a List – Descending Order

30. Add the sort in descending order.

31. Test

```
12     for car in cars:
13         print(car)
14
15     print()
16     print("Sorted List - Descending")
17     cars.sort()
18     cars.reverse()
19     for car in cars:
20         print(car)
21
```



Run: list_sort x

C:\Users\Saddleback\PycharmProjects\Fi

Unsorted List

Porsche

Ford

Mercedes

Sorted List - Ascending

Ford

Mercedes

Porsche

Sorted List - Descending

Porsche

Mercedes

Ford

Process finished with exit code 0

Count

32. Update the cars list and print the number of occurrences of a value in a list.

```

1  #!/usr/bin/env python3
2
3  cars = ["Porsche", "Ford", "Mercedes"]
4
5  print("Unsorted List")
6  for car in cars:
7      print(car)
8
9  print()
10 print("Sorted List - Ascending")
11 cars.sort()
12 for car in cars:
13     print(car)
14
15 print()
16 print("Sorted List - Descending")
17 cars.sort()
18 cars.reverse()
19 for car in cars:
20     print(car)
21
22 cars = ["Porsche", "Ford", "Mercedes", "Porsche", "BMW"]
23
24 print()
25 print("The Porsche occurs " + str(cars.count("Porsche")) + " times")
26

```

33. Test:

Screen Capture #9 (1 point)

```

C:\Users\Saddleback\AppData\Local\Pro
Unsorted List
Porsche
Ford
Mercedes

Sorted List - Ascending
Ford
Mercedes
Porsche

Sorted List - Descending
Porsche
Mercedes
Ford

The Porsche occurs 2 times

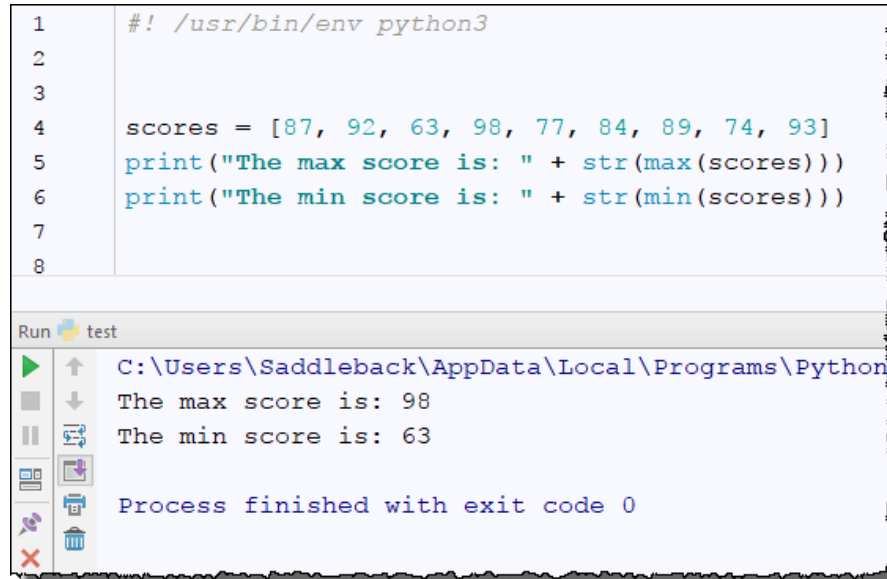
```

Min and Max

Min and Max will return the minimum and maximum values in a list.

34. Code as follows:

Screen Capture #10 (1 point)



```
1  #!/usr/bin/env python3
2
3
4  scores = [87, 92, 63, 98, 77, 84, 89, 74, 93]
5  print("The max score is: " + str(max(scores)))
6  print("The min score is: " + str(min(scores)))
7
8
```

Run test

C:\Users\Saddleback\AppData\Local\Programs\Python\Python38\python.exe

The max score is: 98

The min score is: 63

Process finished with exit code 0

Tuples

Tuples are very similar to lists except a tuple is immutable (i.e., it cannot be changed).

35. Code as follows:

Screen Capture #11 (1 point)



```
1  #!/usr/bin/env python3
2
3  # Creating a tuple
4  stop_light = ("Red", "Yellow", "Green")
5
6  # Accessing with the index
7  print(stop_light[1])
8
```

Run test

C:\Users\Saddleback\AppData\Local\Programs\Python\Python38\python.exe

Yellow

Process finished with exit code 0

Extra Credit

To get full points for each extra credit, you must include screen captures of the running output as well as the python (.py) code files.

Extra Credit #1 – Prime Number Checker (+1 Extra Credit)

Complete a program that checks whether a number is a prime number and displays its factors if it is not a prime number.

```
Prime Number Checker

Please enter an integer between 1 and 5000: 5
5 is a prime number.

Try again? (y/n): y

Please enter an integer between 1 and 5000: 6
6 is NOT a prime number.
It has 4 factors: 1 2 3 6

Try again? (y/n): y

Please enter an integer between 1 and 5000: 200
200 is NOT a prime number.
It has 12 factors: 1 2 4 5 8 10 20 25 40 50 100 200

Try again? (y/n): n

Bye!
```

Specifications:

- A prime number is divisible by two factors (1 and itself). For example, 7 is a prime number because it is only divisible by 1 and itself.
- If the user enters an integer that is not between 1 and 5000, the program should display an error message.
- If the number is a prime number, the program should display a message.
- If the number is not a prime, the program should display a message. Then, it should display the number of factors for the number and a list of those factors.
- Store the factors for each number in a list.
- Use functions to organize the code for this program.

Extra Credit #2 – Tic Tac Toe (+1 Extra Credit)

Create a two-player Tic Tac Toe game.

```

Welcome to Tic Tac Toe

+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+

X's turn
Pick a row (1, 2, 3): 1
Pick a column (1, 2, 3): 1

+---+---+---+
| X |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+

O's turn
Pick a row (1, 2, 3): 1
Pick a column (1, 2, 3): 2

...
...

X's turn
Pick a row (1, 2, 3): 3
Pick a column (1, 2, 3): 3

+---+---+---+
| X | O | O |
+---+---+---+
|   | X |   |
+---+---+---+
|   |   | X |
+---+---+---+

X wins!

Game over!

```

Specifications:

- Use a list of lists to store the Tic Tac Toe grid.
- If the user picks an invalid row or column or a cell that's already taken, display an error message.
- If there is a winner, the game should display an appropriate message and end. Otherwise, it should continue until the grid is full and end in a tie.