QUEENSBOROUGH COMMUNITY COLLEGE
The City University of New York
Department of Engineering Technology

Programming Exercises - Lists

1. A – C, determine the output displayed by the lines of code. Save your code as *PE4_1.py*.

A	`print("Python")`
	`print("Python"[0])`
	`print("Python"[-1])`
	`print("Python"[:])`
		B	`str = "Python 123"`
			`print(str)`
			`print(str[0])`
			`print(str[-1])`
			`print(str[:])`

Output		Output	

C	`strNum = "0, 1, 2, 3, 4, 5, 6, 7, 8, 9"`
	`print(strNum[1], strNum[-1], len(strNum))`
	`print(strNum[:len(strNum)])`
	`print(strNum[1]+strNum[-3])`		

Output			

2. Use list methods to code below. Save the code as *PE4_2.py*.
   a) Create an empty list called *n*.
   b) Add 2 and 4 into the list.
   c) Print the list.
   d) Add 0, 1 and 3 in proper order.
   e) Print the list.
   f) Add 5 in proper order.
   g) Print the list.
   h) Remove 0 from the list.
   i) Print the list.
   j) Remove and print 2 from the list.
   k) Print the list.
   l) Remove and print 4 from the list.
   m) Print the list.
   n) Add all the removed numbers and print the sum.
   o) Change the first item to 100 and last item to 9.9.
   p) Copy the list *n* to a *newNum* list.
   q) Clear the list *n*.
   r) Print the original list, *n* and the *newNum* list.
   s) Delete the list n.

Example Output

```
[2, 4]

[0, 1, 2, 3, 4]

[0, 1, 2, 3, 4, 5]

[1, 2, 3, 4, 5]

[1, 3, 4, 5]
2

[1, 3, 5]
4

Sum of all removed numbers = 6

[100, 3, 9.9]

Original list = []
New list = [100, 3, 9.9]
```

3. Write the codes and precisely produce the output format below. Save the code as *PE4_3.py*.
   a) Create a list called *courses* containing the names of your current courses.
   b) Print the list of courses.
   c) Use the *len* method to print "I am taking X courses" where X is the number of courses in the list.
   d) Using indexes to print the first and last item from the list.
   e) Using slicing to print the first four classes.
   f) Using slicing to print the last four classes.
   g) Using slicing to print the classes except for the first and last classes.

Example Output

```
['ET123', 'ET456', 'ET789', 'ENGL101', 'MA321']
I am taking 5 courses.
ET123   MA321
['ET123', 'ET456', 'ET789', 'ENGL101']
['ET456', 'ET789', 'ENGL101', 'MA321']
['ET456', 'ET789', 'ENGL101']
```

4. Create/Make *PE4_4.py* to do the following:
   a) Create an **empty** list named *grades*.
   b) Add any five grades **one** at a time to *grades*.
   c) Print the current list.
   d) Compute the total of these grades using the **indexing** to reference each number in grades.
   e) Compute the average of these grades using the *len* () function.
   f) Print the average with a precision of **two** decimal places.
   g) Use **two different methods** to remove all failing grades (lower than 60) **one** at a time from the list.
   h) Print the updated list.
   i) Use the built-in functions, *sum* () and *len* () to compute the average with **three** decimal places and print the updated result in one statement.

   Five grades can be any numbers (create two grades less than 60). Just make sure to precisely match the output format below.

```
Example Output
Current List: [92, 51, 83, 37, 72]
Average: 67.00

Updated list: [92, 83, 72]
Updated Average: 82.333
```

5. Requests a name from the user. Save the code as *PE4_5.py*.
   a) Use *Input*() to prompt and request a **full name**.
   b) After the user types the full name and presses the `Enter (or Return)` key, display the **first name** and **last name** in two **separate** lines.

   Input text can be any content. Just make sure to precisely match the output format below.

```
Example Output 1:
Enter the full name of your favorite US president:
george washington
First Name: George
Last Name: Washington

Example Output 2:
Enter the full name of your favorite US president:
Franklin Delano Roosevelt
First Name: Franklin
Last Name: Roosevelt
```

6. A – H, determine and **explain** the output displayed by the lines of code. Save your code as *PE4_6.py*.
A - F, using the given list, `n = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`

A	`print(n, n[:])`
	`print(n[0], n[-10])`
	`print(n[9], n[-1])`
	`print(n[3:])`
	`print(n[:5])`
	`print(n[-5:-1])`
	`print(n[4:8]`

Output	

B	`print(n[-1] + n[-2])`
	`print(n[9] - n[1])`
	`print(n[2] * n[5])`
	`print(n[8] / n[2])`
	`print(len(n), n[:len(n)], sep = '\n')`
	`print(min(n), max(n), type(n), sep = '\t')`

Output	

C	`n[0], n[9] = "apple", 'cherry'`
	`n.insert(3, "banana")`
	`n.insert(-1, "kiwi")`
	`print(n)`
	`print(f"Do you like {n[0].upper()} or {n[-1].upper()}?")`

Output	

D	`n.append(-11)`
	`n.append("orange")`
	`n[0], n[1] = n[-1], n[-2]`
	`print(n+n)`
	`print(n*2)`

Output	

E	`item1 = n.pop(0)`
	`print(f"{item1} is removed.")`
	`item2 = n.pop()`
	`print(f"{item2} is removed.")`
	`print('n = ', n)`
	`print(f'Removed items: {item1} & {item2}')`

Output	

F	`n.insert(6, 'pear')`
	`del n[-1]`
	`del n[0]`
	`print(n)`
	`n.remove("pear")`
	`n.remove(6)`
	`print('n = ', n)`
	`n.clear()`
	`print(f'n = {n}')`

Output	

G	`fruits = ['kiwi', 'pear', 'orange', 'apple', 'cherry']`
	`fruits.sort()`
	`print(fruits[0], fruits[-1])`
	`fruits.sort(reverse=True)`
	`print(fruits[0], fruits[-1])`

Output	

H	`fruits = ['kiwi', 'pear', 'orange', 'apple', 'cherry']`
	`print(sorted(fruits))`
	`print(fruits[0], fruits[-1])`
	
	`print(sorted(fruits, reverse=True))`
	`print(fruits[0], fruits[-1])`

Output	

7. A – C, identify the errors and **rewrite** the statements in the correct syntax. Save your code as *PE4_7.py*.

A	`myList = ['apple','banana','cherry']` `print(myList[3])`
Debug	
B	`print(myList[-1:-4])`
Debug	

C	`word = 'sea'`
	`word[0] = 'p'`
	`print(word)`

Debug	
D	`n = [1, "two", 'three', 4]` `print(" ".join(n))`
Debug	