

Lab 4 (Due: Sep 16)

PYTHON PROGRAMMING FOR DATA SCIENCE - COSC 3360

Department of Computer Science and Electrical Engineering

Fall Semester, 2022

Exercises to practice

(No need to submit)

Exercise 1

Write the following program

```
str1 = 'Catherine'
li = [2, 3, 5]

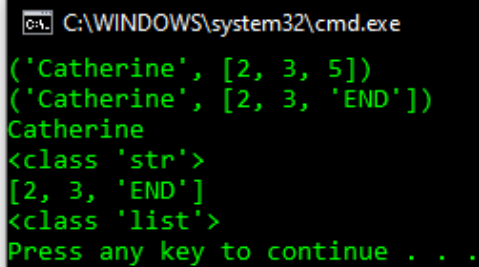
tuple1 = (str1, li)
print(tuple1)

#modifies last element of list:
tuple1[1][-1] = 'END'
print(tuple1)

liStr, liNumANDstrs = tuple1 #unpack sequence

print(liStr)
print(type(liStr))

print(liNumANDstrs)
print(type(liNumANDstrs))
```



```
C:\WINDOWS\system32\cmd.exe
('Catherine', [2, 3, 5])
('Catherine', [2, 3, 'END'])
Catherine
<class 'str'>
[2, 3, 'END']
<class 'list'>
Press any key to continue . . .
```

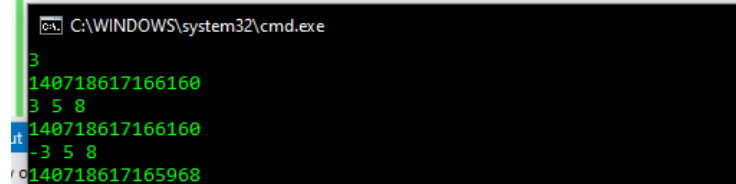
Exercise 2

Write the following program

```
def myF():
    x = 3
    y = 5
    z = 8
    return (x, y, z) #it returns a tuple (parenthesis can be omitted)

tuple1 = myF() #similar to the line below
print(tuple1[0])
print(id(tuple1[0]))
x, y, z = myF() #similar to the line above: unpacking a tuple
print(x, y, z)
print(id(x))

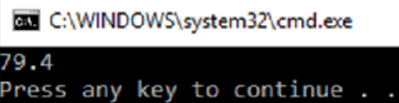
x = -3 # x is not the same variable as x above
print(x, y, z)
print(id(x))
```

**Exercise 3**

Write the following program

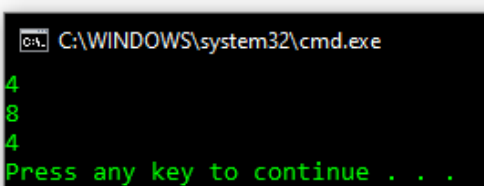
```
def average(*args):
    return sum(args) / len(args)

grades = [88, 75, 96, 55, 83]
result = average(*grades)
print(result)
```

**Exercise 4**

Write the following program

```
numbers = [3, 7, 1, 4, 2, 8, 5, 6, 2]
print(numbers.index(2))
print(numbers.index(2, 5))
print(numbers.index(2, 1, 6))
```



Exercise 5

Write the following program

```
def greaterThanFive(x):  
    print('In function')  
    return x > 5  
  
numbers = [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]  
  
result = filter(greaterThanFive, numbers)  
print(type(result))  
print(result) #prints address of object filter  
result = list(filter(greaterThanFive, numbers)) #function is called here  
print(result)
```

```
C:\WINDOWS\system32\cmd.exe  
<class 'filter'>  
<filter object at 0x00000128B58BA608>  
In function  
In function  
In function  
In function  
In function  
In function  
In function  
In function  
In function  
In function  
[10, 7, 9, 8, 6]  
Press any key to continue . . .
```

Exercise 6

Write the following program

```
print('First lambda')  
result = lambda x: x*x #x*x is the only expression allowed  
print(result(8))  
  
print('Second lambda:', end='...') #does not go to new line  
result2 = lambda x, y: x + y  
print(result2(10, 20))  
  
print('Third lambda:', end=' ') #does not go to new line  
result3 = lambda x, y, z: x + y + z  
print(result3(10, 20, 50))
```

```
C:\WINDOWS\system32\cmd.exe  
First lambda  
64  
Second lambda:...30  
Third lambda: 80  
Press any key to continue . . .
```

Exercises to submit

Create a **New Project** for every exercise. Take a screenshot of the source code along with its output and place the **source code** and the **screenshot** in a **zipped folder** named **LastNameFirstName_Lab4**

Exercise 1

Given the following list: `numbers = [9, -3, 7, 2, 1, 2, 9, 9, 8, 2, 0, 0, 9, 2]`, ask user to enter a number. Pass list along with the number entered into function **myFind()** and find any occurrences of the number in the list. Append the index/indices of the number found into list **listIndex**. Return **listIndex** in *main* program and print list

Exercise 2

Given the following string: `str1 = 'Computer Science'`, use a **filter** to filter out consonants, i.e., your output *list* should contain any vowels found

Exercise 3

Given the following string: `str1 = 'Computer Science'`, use a **filter** to filter out any lower case characters, i.e., your output *list* should contain any upper case characters only

Exercise 4

Create a function named *greetingFunction* that takes in two arguments, the name of the user and a greeting, and prints out the greeting along with user's name; both arguments are given as user input. Re-write the the functionality of the *greetingFunction* using a **lambda** expression

Exercise 5

Given the following list: `grades = [90, 74, 87, 80]`, use a **lambda** expression to compute the average grade. Your algorithm should compute the average grade regardless of the *length* of the list. Use only the **sum** and **len** built-in functions

Note: Submit through **Canvas**