

CSX3001/ITX3001  
FUNDAMENTALS OF COMPUTER  
PROGRAMMING

CLASS 12 MUTABLE AND IMMUTABLE DATA TYPES

PYTHON

## SCOPE OF VARIABLES

Scope of a variable is the portion of a program where the variable is recognized. Parameters and variables defined inside a function is not visible from outside. Hence, they have a local scope.

```
def exFunc():
    x = 100
    print('Value of x inside function: ', x)

x = 50
print('Value of x outside function(Before): ', x)
exFunc()
print('Value of x outside function(After): ', x)
```

## GLOBAL VARIABLES

In Python, a variable declared outside of the function or in global scope is known as global variable. This means, global variable can be accessed inside or outside of the function.

```
aStr = 'global'
def doSth():
    print('aStr in a function: ', aStr)

print('aStr outside a function: ', aStr)
```

## LOCAL VARIABLES

A variable declared inside the function's body or in the local scope is known as local variable.

```
def doSth():
    aStr = 'local'
    print('aStr in a function: ', aStr)

print('aStr outside a function: ', aStr)
```

From above example, you may notice that aStr cannot be used outside function because it is a local variable for a function, doSth().

## GLOBAL KEYWORD

Global keyword is a keyword that allows a user to modify a variable outside of the current scope. It is used to create global variables from a non-global scope *i.e.* inside a function. Global keyword is used inside a function only when we want to do assignments or when we want to change a variable.

```
x = 1
def doSth():
    global x
    x = 20+5
    print('Value of x inside a function: ', x)

print('Value of x outside a function: ', x)
doSth()
print('Value of x outside a function: ', x)
```

What is the output of the above example?

---

---

What will be the output of above example if you remove a line which contains the keyword global?

---

---

## PASSING PARAMETER

Passing parameter(s) to a function in Python is passed by object reference. If parameter(s) is/are mutable object(s) and updated within a function, the value of the argument(s)(passed in parameter(s)) will be changed accordingly.

The mutable object which you have studied so far is list.

Class	Description	Immutable?
<b>bool</b>	Boolean value	✓
<b>int</b>	integer (arbitrary magnitude)	✓
<b>float</b>	floating-point number	✓
<b>list</b>	mutable sequence of objects	
<b>tuple</b>	immutable sequence of objects	✓
<b>str</b>	character string	✓
<b>set</b>	unordered set of distinct objects	
<b>frozenset</b>	immutable form of set class	✓
<b>dict</b>	associative mapping (aka dictionary)	

```
def mutableList(aList):
    aList.append(10)
    print('Value of List aList inside a function: ', aList)

a = [1,2,3]
print('Value of List a outside a function: ', a)
mutableList(a)
print('Value of List a outside a function: ', a)
```

```
def immutableInt(b):
    b += 10
    print('Value of b inside a function: ', b)

a = 100
print('Value of a outside a function: ', a)
immutableInt(a)
print('Value of a outside a function: ', a)
```

### ♦ TRY THESE EXERCISES

1) Given a function, namely

**def isOdd(n):,**

complete this function which will return a Boolean value 'True' when a passing parameter is an odd number, otherwise this function returns 'False'. Write a complete program to ask for an input and test this function.

2) Given a function, namely

**def mid(x, y, z):,**

complete this function which will return a middle value of the three input parameters. Write a complete program to ask for three integers and test this function.

3) Given a function, namely

**def combination(n, r):,**

complete this function which will return the combination of  $n$  chooses  $r$ .

Run the following Python function and observe the result.

```
def removeDuplicate(li):
    newli = []
    seen = set()
    for item in li:
        if item not in seen:
            seen.add(item)
            newli.append(item)

    print(li)
    return newli

li = [1,1,3,5,4,5,5,9,4,8,6]
print(li)
li = removeDuplicate(li)
print(li)
```

Note that a variable *seen* is defined as a set data type, which are unordered collection, indexing has no meaning. Hence the slicing operator `[:]` does not work. To add a value to a set, `.add()` is used which is different from `.append()` that is used to add a value to a list.

- 4) Modify the above function (change the name of the function to `MoDuplicate(int_list)`) to replace odd duplicated integer(s) with 10 times of that duplicated integers, and even duplicated integer(s) with 20 times for that duplicated integers. For example, 1 will be replaced by 10 and 2 will be replaced by 20. Note that the first appearance of an integer remains unchanged.

```
For example, int_list = [1,1,3,5,4,5,4,9,4,8,6]
out_li = MoDuplicate(int_list)
print(out_li)
```

The expected output: `[1, 10, 3, 5, 4, 50, 80, 9, 80, 8, 6]`

The following python code encrypt a message by shifting the characters' position by a number (shift) you enter. The following code can correctly encrypt alphabets (numeric and punctuation marks are not supported).

```
letter = input("Enter the message you want to encrypt (only alphabet and no space): ")
shift = int(input("The number you want to shift by: "))

base = ord("a") #ord() returns an integer representing Unicode character
              # For example, ord("a") returns the integer 97

lower_case = letter.lower() # convert all characters in variable "letter" to lowercase letters
new_strs = [""] #initialize an empty list of characters

for character in lower_case:
    c = (ord(character) - base + shift) % 26 # here we % 26 because there are 26 characters
    new_strs.append(chr(c + base)) #chr() returns a character from integer e.g. chr(97) returns 'a'
                                # append() appends a character to new_strs which is still a List

print ("".join(new_strs)) # "".join --> joins all characters in the List to a string
```

```
Enter the message you want to encrypt (only alphabet and no space): Dragonball
The number you want to shift by: 3
gudjrqedoo
```

- 5) Convert the above code into function, namely `easyencrypt(letter, shift)`, that takes two inputs; letter and shift. The function should return encrypted text and then you can print it outside the function. Expected output is as follows:

```
letter = input("Enter the message you want to encrypt (only alphabet and no space): ")
shift = int(input("The number you want to shift by: "))

encrypted_text = easyencrypt(letter, shift)

print (encrypted_text)
```

```
Enter the message you want to encrypt (only alphabet and no space): Dragonball
The number you want to shift by: 3
gudjrqedoo
```

- 6) Write a function, namely `easydecrypt(en_strs, shift)`, that takes the encrypted text and shift value, and then decrypt the letter. If you can successfully write a function to encrypt, to decrypt a text is just a matter of reverse operation. Expected output is as follows:

```
letter = input("Enter the message you want to encrypt: ")
shift = int(input("The number you want to shift by: "))

en_strs = easyencrypt(letter, shift)
print("Encrypted text is: ",en_strs)

de_strs = easydecrypt(en_strs, shift)
print('Decrypted Text is: ',de_strs)
```

```
Enter the message you want to encrypt: Dragonball is famous in Japan.!!!
The number you want to shift by: 3
Encrypted text is:  Gudjrqedoo lv idprxv lq Mdsdq.!!!
Decrypted Text is:  Dragonball is famous in Japan.!!!
```

## ASSIGNMENTS

1. Write a function to take a string of a combination of 0s or 1s which is the representation of binary number, then convert the binary number to decimal number. The function then returns the integer, which will be printed outside the function. Note that your code must check and ensure that the input will not start with 0.
2. Based on exercises 5 and 6, modify `easyencrypt()` and `easydecrypt()` to include the third parameter, which is a shift direction. If a direction is 'l' (left), `easyencrypt()` must rotate characters left-wise. For example, if a shift is equal to 3 and direction is 'l' (for left), a character 'd' must be changed to 'a'. If the direction is 'r' (for right), `easyencrypt()` will shift a character right-wise (just like in exercise 5).
3. Based on exercises 5 and 6, all letters are converted to lowercase letters. Modify both functions to keep each letter's type as is. For example, shift = 3 and direction is 'r', a capital letter must remain a capital letter.

A man → D pdq