Department of Computer Science

Final Exam Spring 2020

Name: Syed M. Hassan Shah **ID: 6853**

Semester: 8th

Subject: Data Sciences

BS (CS,SE) Instructor: M.Ayub Khan

Note:

At the top of the answer sheet there must be the ID, Name and semester of the concerned Student.

Students must have to provide the output of their respective programs. Students have same answers or programs will be considered fail. Programs in Python and codes should be explained clearly.

As this assignment is online so incase of any ambiguity my Whatsapp no. is 034499121116.

Q1. a. Why Functions are used discuss in detail?

Answer:

A function is a block of reusable code that is used to perform a specific action. The advantages of using functions are:

- Reducing duplication of code
- Decomposing complex problems into simpler pieces
- Improving clarity of the code
- Reuse of code
- Information hiding

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task. To carry out that specific task, the function might or might not need multiple inputs. When

the task is carried out, the function can or cannot return one or more values. Functions in Python are first-class citizens. It means that functions have equal status with other objects in Python. Functions can be assigned to variables, stored in collections, or passed as arguments. This brings additional flexibility to the language.

b. How arguments are used in function, write a simple program in Python?

Answer:

An argument is the values that are sent to the function when it is called.

argv

The special syntax **argv* in function definitions in python is used to pass a variable number of arguments to a function. It is used to pass a non-keyworded, variable-length argument list.

```
In [4]: def uni_name(*argv):
            for arg in argv:
                print (arg)

        uni_name('i','am','student','of','iqra universty')
```

```
i
am
student
of
iqra universty
```

****kwarg***

The special syntax ***kwargs* in function definitions in python is used to pass a keyworded, variable-length argument list. We use the name *kwargs* with the double star. The reason is because the double star allows us to pass through keyword arguments (and any number of them).

```
In [5]: def student(**kwargs):
            for key, value in kwargs.items():
                print ("%s is %s" %(key, value))

        student(name = 'hassan', ID = '6853')
```

```
name is hassan
ID is 6853
```

Q2. a. Why .upper(),.lower(),capitalize() and .swapcase() function are used ?

Answer:

lower()

In Python, lower() is a built-in method used for string handling. The lower() methods returns the lowercased string from the given string. It converts all uppercase characters to lowercase. If no uppercase characters exist, it returns the original string.

Syntax :

string.lower()

Parameters:

lower() does not take any parameters

Returns :

It converts the given string in into lowercase and returns the string.

upper()

In Python, upper() is a built-in method used for string handling. The upper() methods returns the uppercased string from the given string. It converts all lowercase characters to uppercase.If no lowercase characters exist, it returns the original string.

Syntax :

string.upper()

Parameters:

upper() does not take any parameters

Returns :

It converts the given string in into uppercase and returns the string.

capitalize()

In Python, the **capitalize()** method converts the first character of a string to capital **(uppercase)** letter. If the string has its first character as capital, then it returns the original string.

Syntax:

string_name.capitalize()

string_name: It is the name of string of whose first character we want to capitalize.

Parameter: The capitalize() function does not takes any parameter.

Return value: The capitalize() function returns a string with the first character in capital.

swapcase()

The string swapcase() method converts all uppercase characters to lowercase and vice versa of the given string, and returns it.

Syntax:

string_name.swapcase()

Here string_name is the string whose cases are to be swapped.

Parameter: The swapcase() method does not takes any parameter.

Return value:

The swapcase() method returns a string with all the cases changed.

b. Write a program in which the discussed functions are used.

Note : Q2 part a functions.

Upper()

```
In [9]: # Python code for implementation of upper()

        # checking for uppercase characters
        string = 'iqra national university'
        print(string.upper())
```

```
IQRA NATIONAL UNIVERSITY
```

Lower()

```
In [8]: # Python code for implementation of lower()

        # Checking for lowercase characters
        string = 'IQRA NATIONAL UNIVERSITY'
        print(string.lower())
```

```
iqra national university
```

Capitalize()

```
In [5]: # Python program to demonstrate the
        # use of capitalize() function

        # capitalize() first letter of
        # string.
        name = "iqra"
        name2= "national"
        name3= "university"

        print(name.capitalize()+ name2.capitalize()+ name3.capitalize())
```

```
IqraNationalUniversity
```

Swapcase()

```
In [7]: # Python program to demonstrate the use of
        # swapcase() method

        string = "mY nAmE Is haSSan"

        # prints after swappong all cases
        print(string.swapcase())
```

```
My NaMe iS HAssAN
```

Q3. a. What are the rules for defining the function?

Answer:

You can define functions to provide the required functionality. Here are simple rules to define a function in Python.

- Function blocks begin with the keyword **def** followed by the function name and parentheses (( )).
- Any input parameters or arguments should be placed within these parentheses. You can also define parameters inside these parentheses.

- The first statement of a function can be an optional statement - the documentation string of the function or *docstring*.
- The code block within every function starts with a colon (:) and is indented.
- The statement return [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return None.

  Syntax

  def functionname(parameters):
    "function_docstring"
    function_suite
    return [expression]

  By default, parameters have a positional behavior and you need to inform them in the same order that they were defined.

1. Step 1: Declare the function with the keyword def followed by the function name.
2. Step 2: Write the arguments inside the opening and closing parentheses of the function, and end the declaration with a colon.
3. Step 3: Add the program statements to be executed.

All the functions that are written by any us comes under the category of user defined functions. Below are the steps for writing user defined functions in Python.

- In Python, 'def' keyword is used to declare user defined functions.
- An indented block of statements follows the function name and arguments which contains the body of the function.

b. Write a suitable program of our defined function in Python?

In [9]:
```
# Python program to
# demonstrate functions

# Declaring a function
def fun():
    print("data science")

# Driver's code
# Calling function
fun()
```

```
data science
```

Q4. a. What are the rules for defining the function and Parameter passing to the function?

Answer:

- Arguments are passed by value; that is, when a function is called, the parameter receives a copy of the argument's value, not its address. This rule applies to all scalar values, structures, and unions passed as arguments.
- Modifying a parameter does not modify the corresponding argument passed by the function call. However, because arguments can be addresses or pointers, a function can use addresses to modify the values of variables defined in the calling function.

  A function can take multiple arguments, these arguments can be objects, variables(of same or different data types) and functions. Python functions are first class objects. In the example below, a function is assigned to a variable. This assignment doesn't call the function. It takes the function object referenced by shout and creates a second name pointing to it, yell.

  Important methods of Parameter Passing

1. **Pass By Value :** This method uses *in-mode* semantics. Changes made to formal parameter do not get transmitted back to the caller. Any modifications to the formal parameter variable inside the called function or method affect only the separate storage location and will not be reflected in the actual parameter in the calling environment. This method is also called as ***call by value***.

   Pass by reference(aliasing) : This technique uses *in/out-mode* semantics. Changes made to formal parameter do get transmitted back to the caller through parameter passing. Any changes to the formal parameter are

reflected in the actual parameter in the calling environment as formal parameter receives a reference (or pointer) to the actual data. This method is also called as <**em>call by reference**. This method is efficient in both time and space.

b. Write a suitable program of our defined function by parameter passing in Python?

Answer:

```
In [1]: # Python program to illustrate functions
        # can be treated as objects
        def shout(text):
            return text.upper()

        print(shout('Hello'))

        yell = shout

        print(yell('Hello'))
```

```
HELLO
HELLO
```

```
In [ ]:
```

Q5. a. What are return values to a Function discuss in detail?

Answer:

A return statement is used to end the execution of the function call and "returns" the result (value of the expression following the return keyword) to the caller. The statements after the return statements are not executed. If the return statement is without any expression, then the special value none is returned.

Note: Return statement cannot be used outside the function.

Syntax:

```
def fun():
   statements
   .
   .
   return [expression]
```

Returning multiple values:
In Python, we can return multiple values from a function.

b. Write a suitable program of a Function with returning value?
Answer:
Returning value:

```
In [5]: # Python program to
        # demonstrate return statement

        def add(a, b):

            # returning sum of a and b
            return a + b

        res = add(2, 3)
        print("Result of add function is {}".format(res))
```

```
Result of add function is 5
```

Returning multiple values:

```
In [3]: # A Python program to return multiple
        # values from a method using class
        class Test:
            def __init__(self):
                self.str = "data scince assignment"
                self.x = 20

        # This function returns an object of Test
        def fun():
            return Test()

        # Driver code to test above method
        t = fun()
        print(t.str)
        print(t.x)
```

```
data scince assignment
20
```

thankyou