Python_Notes_Weak_06

Functions: A function is a block of code which only runs when it is called. We can pass data, known as parameters, into a function. A function can return data as a result.

Creating functions: In Python a function is defined using the def keyword:

syntax

function define.

```
def Function_Name( p1,p2,p3,...):
        Statements
```

Function call: To call a function, use the function name followed by parenthesis.

Syntax:

```
Function_Name(a1,a2,a3,...)
```

a = arguments(value)

p = parameters (variables)

Arguments: Information can be passed into functions as arguments.Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

Parameters or Arguments?
The terms *parameter* and *argument* can be used for the same thing: information that are passed into a function.

From a function's perspective:
A parameter is the variable listed inside the parentheses in the function definition.
An argument is the value that is sent to the function when it is called.

Number of Arguments
By default, a function must be called with the correct number of arguments. Meaning that if your function expects 2 arguments, you have to call the function with 2 arguments, not more, and not less.

Keyword Arguments
We can also send arguments with the *key* = *value* syntax.
This way the order of the arguments does not matter.

Return Values
To let a function return a value, use the return statement:

Example:

Create a function for average of three numbers.

https://github.com/ibraheem-02/Python_Programs/blob/main/Functions.ipynb

```
def avg (n1,n2,n3):
  sum = n1 + n2 + n3
  average = sum /3
  return average

result = avg(10,20,30)
print(result)
```

https://github.com/ibraheem-02/Python_Programs/blob/main/Assignment.ipynb

Define five functions sum,sub/difference,multiplication,division and average each function have two parameters take two numbers from user and perform the given operation. Show the menu for user choice 1.Sum,2.sub,3.multiplication,4.division,5.Average and 0.Exit.

```
# Define the arithmetic functions
def add(a, b):
    return a + b

def subtract(a, b):
    return a - b

def multiply(a, b):
    return a * b

def divide(a, b):
    if b == 0:
        return "Error: Cannot divide by zero!"
    return a / b

def average(a, b):
    return (a + b) / 2
# Main program loop
while True:
    # Display menu
    print("\n===== Calculator Menu =====")
    print("1. Addition")
    print("2. Subtraction")
    print("3. Multiplication")
    print("4. Division")
```

```
    print("5. Average")
    print("0. Exit")

    # Get user choice
    choice = input("\nEnter your choice (0-5): ")

    # Exit if user chooses 0
    if choice == '0':
        print("Goodbye!")
        break  # Exit the loop

    # Check if choice is valid (1-5)
    if choice not in ['1', '2', '3', '4', '5']:
        print("Invalid choice! Try again.")
        continue  # Restart the loop

    # Get two numbers
    num1 = int(input("Enter first number: "))
    num2 = int(input("Enter second number: "))

    # Perform the chosen operation
    if choice == '1':
        print("Sum:", add(num1, num2))
    elif choice == '2':
        print("Difference Result : ", subtract(num1, num2))
    elif choice == '3':
        print("Multiplication Result : ", multiply(num1, num2))
    elif choice == '4':
        print("Division Result: ", divide(num1, num2))
    elif choice == '5':
        print("Average Result : ", average(num1, num2))
```

Dictionary: Dictionaries are used to store data values in key:value pairs. A dictionary is a collection which is ordered*, changeable and do not allow duplicates. Dictionaries are written with curly brackets, and have keys and values:

Changeable: Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created.

Duplicates Not Allowed: Dictionaries cannot have two items with the same key

Dictionary Items - Data Types: The values in dictionary items can be of any data type.

Dictionary Items: Dictionary items are ordered, changeable, and do not allow duplicates.

Dictionary items are presented in key:value pairs, and can be referred to by using the key.

Python Collections (Arrays)

There are four collection data types in the Python programming language:

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered, unchangeable*, and unindexed. No duplicate members.
- **Dictionary** is a collection which is ordered** and changeable. No duplicate members

Set *items* are unchangeable, but you can remove and/or add items whenever you like.

Example

```
faculty = {

            "name"    : "Ma'am Farhat",
            "course"  : "Python",
            "year"    : "3rd",
            "semester": "2nd"

}
print (faculty.keys())
print (faculty.values())
print (faculty.items())

# change in value
faculty["name"] = "Ma'am Farhat Naureen"
print (faculty)

# adding new key
faculty["department"] = "Computer Science"
print (faculty)
```

Range : The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.

Syntax

range*(start, stop, step*)

Example: print squar of first 7 numbers using range. Output in single line

```
for el in range (1,8):
   print (el*el,end == " ")
```

output.1 4 9 16 25 36 49

Parameter Values

Parameter	Description
start	Optional. An integer number specifying at which position to start. Default is 0
stop	Required. An integer number specifying at which position to stop (not included).
step	Optional. An integer number specifying the incrementation. Default is

For Loop: A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages. With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.

Example: Print odd number from a list

```
L = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

for el in L:
    if el % 2 != 0:
        print (el)
```

Else in For Loop: The else keyword in a for loop specifies a block of code to be executed when the loop is finished.

```
L = [1, 3, 5,  7, 9,]
for el in L:
    print (el)
else:
  print("Loop ended here")
```

Output in a single line

```
L = [1, 2, 3, 4, 5]

for el in L:
    # print (el)
# display out put in single line
     print (el, end = " ") #with space in single line
    # print (el, end = "") #with no space in single line
    # print (el, end = ",") #with comma in single line
```

Take a number from user and print a table using range function

```
n = int(input("Enter a number: "))
for el in range (1,11):
  print (n, "*", el, "=", n*el)
```