PYTHON

Lecture – 13

Recap

- Functions
- Paper Work
    - Write a python code that take name, roll, age of three students in a dictionary and print all data.

Contents

- Functions
- Variable scopes

Functions

- A **function** is a *block of reusable code* that performs a specific task.
- It allows to structure code, making it modular, easier to maintain, and reusable.
- In Python, a function is defined using the **def** keyword, followed by the function name and parentheses (). [`def addition():`]
- You can pass data, known as parameters, into a function (*receive/ catch data*).
- A function can return data as a result (*return or back data*).
- A function only runs when it is called.
-

Key Elements of Function

- **def**: This keyword is used to define a function.
- **function_name**: This is the name of the function. It should be meaningful and indicate the purpose of the function.
- **parameters**: These are the input values passed to the function.
- **return**: This keyword is used to return a value from the function.
- **Docstring**: A short description of what the function does (optional, but recommended for clarity).

Example

- A function without parameter:

```
def hello():
    print("Hello python")
hello()
```

- A function with parameter:

```
def hello(name):
        print(f"Hello {name}")
hello("Doha")
```

Example

- Write a program that take two numbers as input and print their sum as output.

Example

- Write a program that take two numbers as input and print their sum as output.

```
num1=int(input("Enter first number: "))
num2=int(input("Enter 2nd number: "))
sum=num1+num2
print(f"Sum of {num1} and {num2} is: {sum}")
```

Now, write this code using function.

Example

- Addition using function:

```
def addition(n1, n2): #function declaration
    sum=n1+n2
    return sum

num1=int(input("Enter first number: "))
num2=int(input("Enter 2nd number: "))
summation=addition(num1,num2) #function calling
print(f"Sum of {num1} and {num2} is: {summation}")
```

Exercises on Function

1. write a function that find the maximum number from a given number list.
2. Write a function is_even(number) that returns True if the given number is even and False if it's odd.
3. Write a function factorial(n) that calculates and returns the factorial of a given number n
3. Write a function sum_list(numbers) that takes a list of numbers and returns the sum of all elements in the list.
4. Write a function is_prime(n) that checks if the given number is prime. Return True if it is prime, and False otherwise.

Exercises on Function

12.6 write a function that find the maximum number from a given number list.

```
def find_max(numbers):
    max_num = -1000
    for num in numbers:
        if num > max_num:
            max_num = num   # Update max_num number
    return max_num

# Numbers List
number_list = [34, 12, 56, 78, 23, 45]
max_number = find_max(number_list)
print(f"The maximum number is: {max_number}")
```

Lamda

- A lambda function is a small anonymous function.
- A lambda function can take any number of arguments, but can only have one expression.
- Example:

```
add = lambda x, y: x + y
print(add(5, 3)) # Output: 8

x = lambda a, b, c : a + b + c
print(x(5, 6, 2))

x = lambda a : a + 10
print(x(5))
```

Python Variable Scope

The **scope of a variable** in Python is defined as **the specific area or region** where the variable is **accessible** to the user.

Python variables are classified in three categories −

- Local Variables
- Global Variables
- Nonlocal Variables

Local Variable

- A variable created inside a function belongs to the *local scope* of that function, and can only be used inside that function.

```
def myfunc():
            x = 300 #x is local in myfunc()
            print(x)
    myfunc()
```

Scope of Variables in Python

Local Variable

A local variable is **defined within a specific function or block of code**. It can only be **accessed by** the function or block where it was defined, and it has a limited scope.

```
def myfunction():
  a = 10                    #local variable
  b = 20                    #local variable
  print("variable a:", a)
  print("variable b:", b)
  return a+b


print (myfunction())
```

Scope of Variables in Python

Local Variable

A local variable is **defined within a specific function or block of code**. It can only be **accessed by** the function or block where it was defined, and it has a limited scope.

```
n = int(input())
list = []
for i in range(n):
    x = int(input())                #local variable
    list.append(x)
print(list)
```

Nonlocal Variable

The variable x is not available outside the function, but it is available for any function inside the function. What about y?

```
def myfunc():
  x = 300
  def myinnerfunc():
    y=10
    print(x)
  myinnerfunc()
myfunc()
```

Nonlocal Variable

If you use the nonlocal keyword, the variable will belong to the outer function:

```
def myfunc():
  x = 300
  def myinnerfunc():
    nonlocal y=10
    print(x)
  myinnerfunc()
myfunc()
```

Scope of Variables in Python

Nonlocal Variables

The Python variables that are not defined in either local or global scope are called nonlocal variables.
They are used in nested functions.

```
def yourfunction():
  a = 5
  b = 6
  # nested function
  def myfunction():
     # nonlocal function
     nonlocal a
     nonlocal b
     a = 10
     b = 20
     print("variable a:", a)
     print("variable b:", b)
     return a+b
  print (myfunction())
yourfunction()
```

Output: 30

variable should not belong to the inner function

Global Variable

- A variable created in the main body of the Python code is a global variable and belongs to the global scope.
- Global variables are available from within any scope, global and local.

```
x = 300
def myfunc():
  print(x)

myfunc()
print(x)
```

Global Variable (using global keyword)

- If you need to create a global variable, but are stuck in the local scope, you can use the global keyword.
- The global keyword makes the variable global.

```
def myfunc():
  global x
  x = 300

myfunc()
print(x)
```

Exercises

13.1 Write a function sum_numbers() that takes two numbers as arguments and returns their sum.
13.2 Write a function factorial(n) that calculates the factorial of a given number n.
13.3 Write a function is_prime() that takes a number as an argument and returns True if the number is prime, and False otherwise.
13.4 Write a function find_max() that takes a list of numbers and returns the maximum value.
13.5 Write a function find_even() that takes a list of numbers as input and returns a list of only the even numbers.
13.6 Write a function to calculate sum of digits in a number.