Implementing Algorithms by using Python

Prof. Damitha Karunaratna

University of Colombo school of computing

Accessing Python

Python provide two tools to work with the interpreter.

1) interactive interpreter (also known as a **shell)**

It goes through Read –Eval-Print Loop (REPL).

2) Integrated Development and Learning Environment (IDLE)

How to find the folder where Python is installed

```
import os

import sys

os.path.dirname(sys.executable)
```

Or in DOS O/S type the following command

```
where python
```

You may run Python IDLE directly at the command prompt by typing

```
python -m idlelib
```

Add a key-board short cut.

Find the folder where the file idle.bat (**Integrated Development and Learning Environment)** is installed.

Add a new shortcut

What you should know about a programming language

1. How an application can be broken down to components.
2. Structure of a program
3. How a program can communicate with the external world.
4. Data types and classification of data types.
5. Variables and how types are assigned to variables.
6. Different type of operations such as assignment operators, mathematical operators, Logical operators, bit-wise operators
7. In which order statement are executed and this order of execution can be changed.
8. How narration of code can be embedded with executable statements

Various high-level components in a Python application.

1. How an application can be broken down to components.
    1. Modules
        1. Defining a Module.
        2. Using definitions in a module
    2. Functions
    3. Statement blocks

Structure of a Python program

- Collection of functions only (as in C or Java??)
- A mixture of
  - Collection of functions and
  - a sequence of executable statements?

Main()????

- Many programming languages have a special function that is automatically executed when an operating system starts to run a program. This function is usually called main() and must have a specific return type and arguments according to the language standard.
- The Python interpreter executes scripts starting at the top of the file, and there is no specific function that Python automatically executes.

How programs can communicate with the external world

Using input output features.

Sending and receiving user input from the standard input/output devices.

- Standard input device – Keyboard
- Standard output device – Monitor

- Sending and receiving user input from other devices such as data files, database.

Data types and classification of data types.

1. Data types :
    1. Integer, Float, Boolean, Strings
    2. Tuples, Lists ,Dictionaries, Sets
2. Classification of data types.
    1. Sequencies vs non-sequence
    2. Mutable vs non-mutable
    3. Hashes

What you should know about a programming language

1. How a program can communicate with the external world.
2. Data types and classification of data types.
3. Variables and how types are assigned to variables.
4. Different type of operations such as assignment operators, mathematical operators, Logical operators, bot-wise operators
5. In which order statement are executed and this order of execution can be changed.
   1. Selection
   2. Looping
6. How narration of code can be embedded with executable statements

Project

Construct a menu driven Python program to manage the marks obtained by students at an examination.

Quiz

What are Jython, IronPython, PyPy: Cpython ?

Python implementations

The specification of the Python language is implemented by using different programming languages.

Example :

Jython: Java implementation of python.

IronPython: Implemented in C#.

PyPy: Implementation of python in python(Rpython).

Cpython : Implemented in C.

```
import platform
platform.python_implementation()
```

Id function

Id() function return the identity of an object.

The identity of an object is unique and constant for the object during the lifetime of that object.

In Cpython implementation it is also the starting address of the memory location of the object.

Accessing a value at a memory location(Only for Teachers)

```
import ctypes

ctypes.cast(id(x),ctypes.py_object).value
```

Data Type?

Why data types are needed in programming.

Python is a **Dynamically typed** Language ?

Dynamic typing means that **the type of the variable is determined only during runtime**.

What are the data types of the following variables?

Data type = A collection of values + a set of operation allowed on such data item.

v1 = 2

v2 = 2.4

v3 = “abc”

v4 = True

v5 = (1,2,3)

V6 = [1,2,3]

Problems

What is done by the Python interpreter when executing the following Python statement.

```
 x = 23
y = 23
```

What is the problems in the following statements.

```
x = [1,2,3]

Print(x[1])

x[1] = 5


y = “abc”

print(y[1])

y[1] = “5”
```

Packing values to a variable

```
v = 1,2,3
```

What is the value of v?

Un-packing values of a collection

```
x,y,z =[1,2,3]
```

What are the values of variables x,y,z?

Problems

```
x=1

y=2


x,y =y,x
```

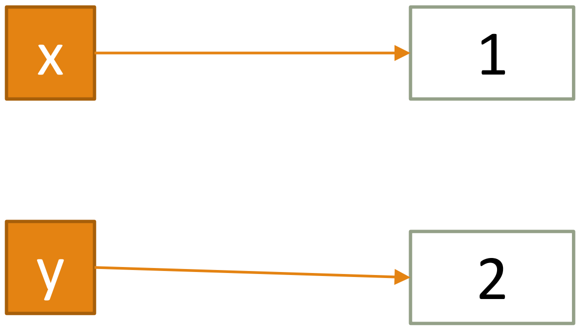
What are the values of x and y after executing these line? What is the logic?

Problems – step 0

```
x=1

y=2


x,y =y,x
```



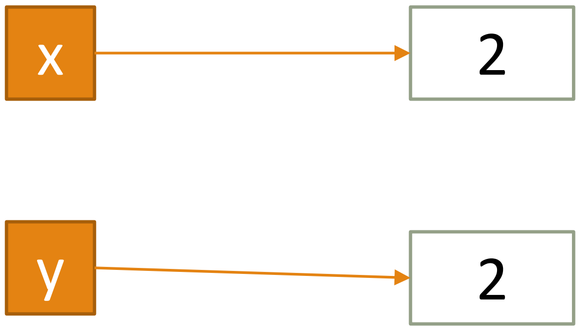
What are the values of x and y after executing these line? What is the logic?

Problems – Step 1

x=1

y=2

x,y =y,x

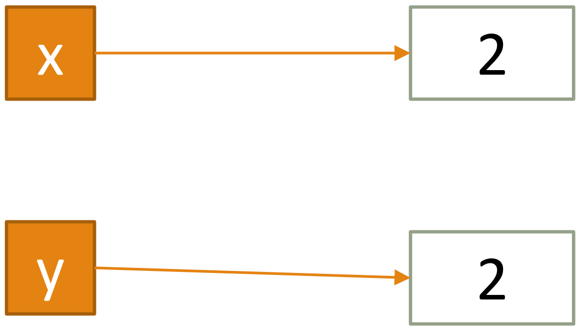


Problems – Step 2

x=1

y=2

x,y =y,x



Problems

x=1

y=2

x,y = y,x

Logic

Step1:

x,y = (2,1)

Problem

What is the output of the following Python program when executed?

```
for i in range(1,10):
    print(i)
```

Problem

How do you change this code if the same sequence of instructions has to executed at multiple points of a program?

```
# some code sequence

#Generate integers from 1 to 10

# some code sequence

#Generate integers from 11 to 20

# some code sequence

#Generate integers from 21 to 30
```

Problem

```
for i in range(1,11):
    print(i)
for i in range(11,21):
    print(i)
for i in range(21,31):
    print(i)


//Same code is repeated at multiple points of the program
// with different parameters
```

What will happen to your code if python does not provide range function?

Functions

```
//Function definition
def functionName(parameters):
  functionBody


//parameters are optional
//A function has 1)Definition  2)Invocation
```

Functions

```
#Function definition
def f1():
    print("I am in a function")
#function invocation
f1()
```

Functions

```
#Function definition
def printNums(f,l):
    for i in range(f,l):
        print(i)
printNums(1,11)
printNums(11,21)
printNums(21,30)
```

What will happen if the function is called as printNums(5)

Functions with return values

Create a function to sum up all integers in a sequence from j to k.

Sum = j+(j+1)+(j+2)+……..+k

Implementation 1

```
def mysum(f,l):
   x = sum(range(f,l+1))
   return x
sum1 = mysum(1,10)
print("Sum of numbers from 1 to 10 = ", mysum(1,10))
);
```

Implementation 2

```
def mysum(f,l):
    x = sum(range(f,l+1))
    return x
print("Sum of numbers from 1 to 10 = ",mysum(1,10));
```

Passing parameters by position

```
def fx(a,b):
    print("Value of a=",a,"Value of b=",b)


fx(2,3)
```

Matching by using positions

Matching by using keywords

Passing parameters by keywords

```
def fx(a,b):
    print("Value of a=",a,"Value of b=",b)


fx(b=2,a=3)
```

Function overloading

What is function overloading ?

Defining multiple functions with the same name but with different parameters.

Many object-oriented languages allow function overloading. How about Python.

Python does not support method overloading. You may overload a method but can use the last overloaded method.

Default parameters

```
def myadd(x=9,y=5):
    print("x=",x,"y=",y,"sum =",x+y)


myadd(2,2);
myadd(2);
myadd()
```

Recursive functions

Recursive function – A function that calls itself.

Fact(n) = n *Fact(n-1)

```
def fun(x):
    if(x == 1):
        return 1
    else:
        return x*fun(x-1)
print(fun(5))
```

fun(5) <- 5*fun(4) <- 5*4*fun(3) <-5*4*3*fun(2)<-5*4*3*2*1

Functions with different number of parameters- For teachers

```
def f1(*x):
   return(sum(x))


print(f1(1,2))
print(f1(1,2,3,4))
```

High order functions - For teachers

```
def fx(a,b):

    return a + b


def fy(a,b):

    return a - b


def f1(f,a,b):

    return(f(a,b))


print("Calling function with fx ",f1(fx,5,2))

print("Calling function with fy ",f1(fy,5,2))
```

Positional Vs Keyword parameters - For teachers

```
def fx(*a,**b):
    print(a)
    print(b)


fx(1,2,3,a=5,b=6,c=7)
```

Scope of variables:

Creating variables in Python:

Unlike other programming languages, Python has no command for declaring variables.

A new variable is created when it is assigned a value.

Example :

```
x = 123    # Create a variable
print(x)     # Use the value of the variable x
x = 40  # Modify the value of the variable
```

Scope of variables:

Scope of a variable is the block of code of the entire program where that variable is visible and can be used, modified its value. There are three types of variable scopes.

- Global Scope
- Local Scope
- Non-local Scope

Scope of variables: Global scope

```
a = "111"  #globl scope
//Variable is declared outside of any function

def f1():
    print("value of a inside function f1=",a)

f1()
print("Value of a outside functions =",a)
```

Scope of variables: local scope

All variables created inside a function (including variables in the function header) has local scopes.

Example :

```
def f1():
    x = "123"
    print("value of x inside function f1=",x)

f1()
print("Value of a x outside functions =",x)
```

Scope of variables: non-local scope

```
x = 200
def f1():
    def f2():
        #Value of x is non-local inside function f2 but it is not global
        print("value of x in function f2 =",x);

    f2()

f1()
print("Value of a x outside functions =",x)
```

Listing out all objects visible

The function dir() can be used to list all objects visible.

Examples

```
dir()

dir(int)
```

Python Magic or Dunder methos and variables

Magic methods in Python are the special methods that start and end with the double underscores.

They are also called “dunder” methods.

Magic methods are not meant to be invoked directly by you.

References

- https://www.youtube.com/watch?v=NcaMLIUpjbY
- https://www.youtube.com/watch?v=NcaMLIUpjbY
- https://www.youtube.com/watch?v=0BhSWyDEDC4&t=121s
- https://www.youtube.com/watch?v=arxWaw-E8QQ