Python

Introduction to Output/Input, Assignment, Variables, Types, Loops, Conditions

General guidelines

- Python is easy to use programming language
- Python is case sensitive, so PYTHON, Python and python are different
- In Python, reserve words, functions and variable/object names are generally in small letters
- Every statement/construct (except headers) should be in a line
- Header are first lines of function, loop or condition, etc must be terminated with colon :
- Block of statements or dependent statement after headers should be indented, e.g.,

    header:

        block statements (each on a new line)

Output

print(expr)

or

print(expr1, expr2, … exprN)

Evaluates the expression expr and put result (its values) on the output device at the cursor's location

expr may be any type of expression in Python.

, end=string expr can be added before)

Input

var = input()

or

var = input(string_expr)

or

var1, var2, …, vark = input(…).*split()*

Get data (value) from input device at cursor's location and stores that in the var as a string

var is the name of the variable

Assignment operation

var = expr

or

var1, var2, ..., varN = expr1, expr2, ..., exprN

Evaluate each expr and stores its value in respective in var one by one

Simple Scalar Data Types (built-in)

bool   **Boolean values**
True or False

int   **Integer values**
sequence of digits with possibly – sign at beginning

float   **Real number values**

str   **Stings, a sequences of any characters**
enclosed in quotes ″ or ′

complex   to store complex numbers

Identifiers

- **Identifier (a variable name, function name, etc) must start from an alphabet or underscore character**
- It has a sequence of alphabet, digits and underscore _
- **Capital and small alphabets are treated as different in variable names**
- It must be unique within scope and must not be the Python reserve word
- **A, a, i, x, alpha, num, average, _k, student_1, While are valid identifiers (a variable name, function name, etc)**
- **while** is not valid as it a reserve word, **1hy** is not valid as starting from digit, and **Sr No** is also not valid having space in it. Complete list of Python reserve word may be available in book, or you may find them at

  https://www.programiz.com/python-programming/keyword-list

A Program example

```
print("enter three values one by one")

a = input()
b = input()
c = input()

a = float(a)
b = float(b)
c = float(c)

avrg = ( a + b + c ) / 3

print("their average is", end=" ")
print(avrg)
```

Selection (Conditions)

```
if condition:
    Statement 1
    Statement 2
    Statement 3
    …
    Statement k
```

```
if condition:
        Statement y1;
        Statement y2;
        …
        Statement yk;
else:
        Statement n1;
        Statement n2;
        …
        Statement nk;
```

- Lines starting with word if is its header and hence are terminated by colon : , else is also terminated with colon :

- if only: if condition is true statements 1..k are executed otherwise these all have to be skipped.

- if and else both available: if condition is true statements y1..yk are executed and statements n1..nk are skipped

  but if condition is false y1..yk are skipped and statements n1..nk are executed

Iterations or repetitions (Loops)

```
initialize loop variables;
while condition:
        Statement 1
        Statement 2
        Statement 3
        ...
        Statement k

        advance loop variables;
```

Executes zero or more times

```
for var in range:
        Statement 1
        Statement 2
        Statement 3
        ...
        Statement k
```

- Lines in **greenish** color are not part of loop syntax, but they are very mandatory

- Lines starting with words while or for are headers of loop and hence are terminated by colon **:**

- In while, when condition is true statements 1..k + advance loop variables are executed, and condition is retested after them, ...

- In for loop, statements 1..k are executed for each value in range and var can be used in statements 1..k to refer values in range 1by1

Built in functions

	Function	Description
•	abs()	Returns the absolute value of a number
•	ascii()	Returns a readable version of an object. *Replaces none-ascii characters with escape character*
•	bin()	Returns the binary version of a number
•	bool()	Returns the boolean value of the specified object
•	chr()	Returns a character from the specified Unicode code
•	complex()	Returns a complex number
•	divmod()	Returns the quotient and the remainder when argument1 is divided by argument2
•	float()	Returns a floating point number
•	help()	Executes the built-in help system
•	hex()	Converts a number into a hexadecimal value
•	id()	Returns the id of an object
•	input()	Allowing user input

Built in functions

Function	Description
int()	Returns an integer number
isinstance()	Returns True if a specified object is an instance of a specified object
len()	Returns the length of an object
oct()	Converts a number into an octal
ord()	Returns code for the specified character
pow()	Returns the value of x to the power of y
print()	Prints to the standard output device
range()	Returns a sequence of numbers, starting from 0 and increments by 1 (by default)
round()	Rounds a numbers
str()	Returns a string object
type()	Returns the type of an object

Values are Objects in Python