# OPEN ELECTIVE
# OCS1903-PROGRAMMING USING PYTHON
# WEEK 2

# CONTENTS

- Python Basics
  - Comments
  - Print
  - User Input
  - Variables
  - Datatypes
  - Operators

# Python Basics-Comments

1. Comments are used for documenting the code .
2. Comment lines will not be executed by the python interpreter.

**Single Line Comment:**

#This is a single line comment

**Multi-line Comment:**

“”” This is multiple ”””

**Multiline statements:**

```
result=(2*5)+
       9
```

```
result=(2+3)*(7+8)\
  *(9+10)
```

```
result=(2*4*
    4)
```

(Acceptable for all kinds of brackets)

# Python Basics-Print statement

To ***display output*** on the screen of the user

```
>>> print("hello")
hello
>>> print(1)
1
>>> print('a')
a
>>> print(3.24)
3.24
>>>
```

```
>>> a=10
>>> b=20
>>> print("The sum of a and b:",a+b)
The sum of a and b: 30
>>>
```

```
>>> print(a,b)
10 20
>>>
```

```
>>> print(1,2,"we",3.25)
1 2 we 3.25
>>>
```

# Python Basics-User Input statement

To ***get user input*** through keyboard

All inputs got from the user are treated as **Strings**

```
>>> a=input("Enter your name:")
Enter your name:sorna
>>> a
'sorna'
>>>
```

```
>>> type(a)
<class 'str'>
>>>
```

```
>>> b= input("Enter your age")
Enter your age32
>>> b
'32'
```

```
>>> type(b)
<class 'str'>
>>>
```

```
>>> a=input()
3.14
>>> a
'3.14'
>>> type(a)
<class 'str'>
>>>
```

# Python –Fundamental Components

- Operators
- Identifiers
- Variables
- Datatypes

## Identifiers

- Are names given to anything that you want to identify in a program
- Helps to refer to that item from any place in the program
- Can start with an underscore (_) or a upper or lower case alphabet
- Can have digits

Case-Sensitive
Cannot match keywords

Eg : Bill_id
_billid1

## Variables

- An identifier for the data and it holds data in your program
- Is a location (or set of locations) in memory where a value can be stored
- A quantity that can change during program execution
- **No declaration of variables**

```
customer_id = 101                          # Integer
customer_name = "John"                     # String
bill_amount = 675.45                       # Floating-point
x = 5.3 + 0.9j                             # complex number
print(customer_id, customer_name, bill_amount)   #prints 101 John 675.45
print(x.real)                              #prints 5.3
print(x.imag + 3)                          #prints 3.9
```

# Variables and its dimensions

Identify the variable

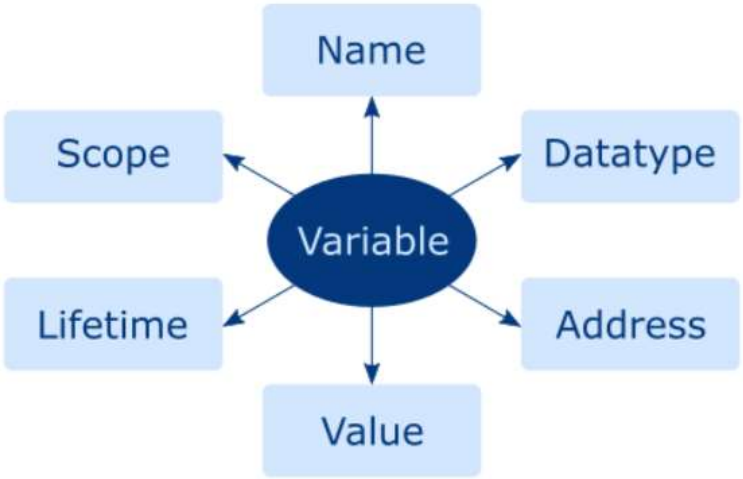
Specify where it can be accessed

Determine type , memory required and operations that can be performed on it

Specify how long it will be stored in memory

Specify where it is stored

Data held by the variable



# Datatypes

| Category | Data Type | Example |
|---|---|---|
| Integer Type | int | 675 |
| | complex | 2 + 5i |
| Floating Type | float | 642.43 |
| Textual | String | Infosys |
| Logical | boolean | True, False |

# Built-in functions

For handling different types of user inputs

int()-returns a integer number
float()-returns a decimal/floating point number
bool()-returns a Boolean value
complex()-returns a complex number
str()-returns a string

```
#Program to convert user inputs to different datatypes
name=input("Enter ur name")
age=int(input("Enter ur age"))
height=float(input("Enter ur height"))
print(name,age,height)
print(type(age),type(height),type(name))
```

# Simple Python codes

```
#Python program to convert no of days into no of seconds
no_of_days=10
no_of_seconds=10*24*60*60
print("The no of seconds is:",no_of_seconds)
```

```
#program to find no of flights in airport
in_flights=int(input("Enter the intial no of flights:"))
takeoff=int(input("TAKEOFF:"))
landed=int(input("LANDED:"))
res_fli=in_flights+landed-takeoff
print("The no of flights are:",res_fli)
```

# Operators

- Arithmetic Operators
- Relational Operators
- Assignment Operators
- Bitwise Operators
- Logical Operators
- Membership Operators
- Identity Operators

# Arithmetic Operators

- An arithmetic operator is a mathematical operator that takes two operands and performs a calculation on them. They are used for simple arithmetic.

# Arithmetic Operators

- Arithmetic Operators
  - Used for performing arithmetic operations

| Operators | Description | Example |
|---|---|---|
| + | Additive operator (also used for String concatenation) | 2 + 3 = 5 |
| - | Subtraction operator | 5 – 3 = 2 |
| * | Multiplication operator | 5 * 3 = 15 |
| / | Division operator | 6 / 2 = 3.0 |
| % | Modulus operator | 7 % 2 = 1 |
| // | Truncation division (also known as floor division) | 10 // 3 = 3<br>10.0 // 3 = 3.0 |
| ** | Exponentiation | 10 ** 3 = 1000 |

```
#Demo Program to test Arithmetic Operators
a=100
b=10
print ("The Sum = ",a+b)
print ("The Difference = ",a-b)
print ("The Product = ",a*b)
print ("The Quotient = ",a/b)
print ("The Remainder = ",a%30)
print ("The Exponent = ",a**2)
print ("The Floor Division =",a//30)
#Program End
```

Output:
The Sum = 110
The Difference = 90
The Product = 1000
The Quotient = 10.0
The Remainder = 10
The Exponent = 10000
The Floor Division = 3

# Relational or Comparative operators

- A Relational operator is also called as Comparative operator which checks the relationship between two operands. If the relation is true, it returns True; otherwise it returns False

# Relational Operators

| Operators | Description |
|---|---|
| == | Equal to |
| < | Less than |
| > | Greater than |
| <= | Less than or equal to |
| >= | Greater than or equal to |
| != | Not equal to |

```
#Demo Program to test Relational Operators
a=int (input("Enter a Value for A:"))
b=int (input("Enter a Value for B:"))
print ("A = ",a," and B = ",b)
print ("The a==b = ",a==b)
print ("The a > b = ",a>b)
print ("The a < b = ",a<b)
print ("The a >= b = ",a>=b)
print ("The a <= b = ",a<=0)
print ("The a != b = ",a!=b)
#Program End
```

Output:
Enter a Value for A:35
Enter a Value for B:56
A = 35 and B = 56
The a==b = False
The a > b = False
The a < b = True
The a >= b = False
The a <= b = False
The a != b = True

# Logical operators

- In python, Logical operators are used to perform logical operations on the given relational expressions. There are three logical operators they are and, or and not

# Logical Operators

**Logical operators** are used to combine one or more relational expressions.

| Operators | Description |
|---|---|
| AND | Result will be true, if both the expressions are true. If any one or both the expressions are false, the result will be false |
| OR | Result will be true, even if one of the expression is true. If both the expressions are false, the result will be false |
| NOT | If the expression is true, result will be false and vice versa |

If A=(m>200), B=(m>300)

| A | B | A AND B |
|---|---|---|
| True | True | True |
| True | False | False |
| False | True | False |
| False | False | False |

| A | B | A OR B |
|---|---|---|
| True | True | True |
| True | False | True |
| False | True | True |
| False | False | False |

| A | NOT A |
|---|---|
| True | False |
| False | True |

```
#Demo Program to test Logical Operators
a=int (input("Enter a Value for A:"))
b=int (input("Enter a Value for B:"))
print ("A = ",a, " and b = ",b)
print ("The a > b or a == b = ",a>b or a==b)
print ("The a > b and a == b = ",a>b and a==b)
print ("The not a > b = ",not a>b)
#Program End
```

Enter a Value for A:50

Enter a Value for B:40

A = 50 and b = 40

The a > b or a == b = True

The a > b and a == b = False

The not a > b = False

# ASSIGNMENT OPERATORS

- In Python, = is a simple assignment operator to assign values to variable. Let a = 5 and b = 10 assigns the value 5 to a and 10 to b these two assignment statement can also be given as a,b=5,10 that assigns the value 5 and 10 on the right to the variables a and b respectively.
- There are various compound operators in Python like +=, -=, *=, /=, %=, **= and //= are also available.

# Assignment Operators

- Assignment Operators

| Operators | Description | Example | Equivalent |
|---|---|---|---|
| = | Assignment from right side operand to left side | c = 50; c = a; | |
| += | Add & assigns result to left operand | c += a | c = c + a |
| -= | Subtract & assigns result to left operand | c -= a | c = c – a |
| *= | Multiply & assigns result to left operand | c *= a | c = c * a |
| /= | Divide & assigns result to left operand | c /= a | c = c / a |
| %= | Calculates remainder & assigns result to left operand | c %= a | c = c % a |
| //= | Performs floor division & assigns result to left operand | c //= a | c = c // a |
| **= | Performs exponential calculation & assigns result to left operand | c **= a | c = c ** a |

```
#Demo Program to test Assignment Operators
x=int (input("Type a Value for X : "))
print ("X = ",x)
print ("The x is =",x)
x+=20
print ("The x += 20 is =",x)
x-=5
print ("The x -= 5 is = ",x)
x*=5
```

```
print ("The x *= 5 is = ",x)
x/=2
print ("The x /= 2 is = ",x)
x%=3
print ("The x %= 3 is = ",x)
x**=2
print ("The x **= 2 is = ",x)
x//=3
print ("The x //= 3 is = ",x)
#Program End
```

Type a Value for X : 10
X = 10
The x is = 10
The x += 20 is = 30
The x -= 5 is = 25
The x *= 5 is = 125
The x /= 2 is = 62.5
The x %= 3 is = 2.5
The x **= 2 is = 6.25
The x //= 3 is = 2.0

# Bitwise Operators

- **Bitwise Operators**
  - performs bit by bit operation on bits

| Operators | Description |
|---|---|
| & | Binary AND |
| \| | Binary OR |
| ^ | Binary XOR |
| ~ | Binary Ones Complement |
| << | Binary Left Shift |
| >> | Binary Right Shift |

# Shifts

## BITWISE OPERATORS

### << Shift Left

| SYNTAX | BINARY FORM | VALUE |
|---|---|---|
| x = 7; | 00000111 | 7 |
| x=x<<1; | 00001110 | 14 |
| x=x<<3; | 01110000 | 112 |
| x=x<<2; | 11000000 | 192 |

**We can also use a formula to find the answer,**

**$x<<y=x*2^y$**

## BITWISE OPERATORS

### >> Shift Right

| SYNTAX | BINARY FORM | VALUE |
|---|---|---|
| x = 192; | 11000000 | 192 |
| x=x>>1; | 01100000 | 96 |
| x=x>>2; | 00011000 | 24 |
| x=x>>3; | 00000011 | 3 |

**We can also use a formula to find the answer,**

**$x>>y=x/2^y$**

# Membership and Identity Operators

- **Membership Operators**
  - Checks for whether the given element is found within a sequence of Strings, Lists, Dictionaries or Tuples

| Operators | Description |
|---|---|
| in | Returns true if given element is found in given sequence else false |
| not in | Returns true if given element is not found in given sequence else false |

- **Identity Operators**
  - Are used to compare memory locations or addresses of 2 objects

| Operators | Description |
|---|---|
| is | Returns true if variables or objects on both sides of operator are referring to same object else false |
| is not | Returns false if variables or objects on both side of operator are referring to same object else true |

# Coding Standards in python

- Set of guidelines
  - To Enhance the readability and Clarity of the program
  - Make it easy to debug and maintain the program
- All the letters in a variable name should be in lowercase
- When there are more than two words in variable name, underscore can be used between internal words
- Use meaningful names for variables
- Limit all lines to a maximum of 79 characters
- A function and class should be separated by 2 blank lines
- Methods within classes should be separated by single blank line

# Formatting output

- There are several ways to present the output of a program, data can be printed in a human-readable form, or written to a file for future use. Sometimes user often wants more control the formatting of output than simply printing space-separated values. There are several ways to format output.
- To use formatted string literals, begin a string with f or F before the opening quotation mark or triple quotation mark.
- The str.format()method of strings help a user to get a fancier Output

# Simple Python codes

Jack and his three friends have decided to go for a trip by sharing the expenses of the fuel equally . Write a Python program to calculate the amount (in Rs) each of them need to put in for the complete (both to and fro) journey. The program should also display True, if the amount to be paid by each person is divisible by 5, otherwise it should display False. (**Hint**: Use the relational operators in print statement.

Assume that mileage of the vehicle, amount per litre of fuel and distance for one way are given.

| Sample Input | | | Expected Output |
|---|---|---|---|
| Mileage of the vehicle (km/litre of fuel) | Amount per litre of fuel (Rs) | Distance for one way (kms) | |
| 12 | 65 | 96 | 260.0<br>True |

```
mileage=int(input())
fuel_cost=int(input())
distance=int(input())
total_distance=2*distance
fuel_required=total_distance/mileage
total_amount=fuel_cost*fuel_required
each_amt=total_amount/4;
print(each_amt);
print(each_amt%5==0)
```


THANK
YOU