APSSDC

Andhra Pradesh State Skill Development Corporation

(https://apssdc.in)

Day03 Online Training on Python Programming

Day03 Objectives

- Python Basics
  - print()
  - input()
  - identifier and Properties of identifier
  - Keywords in Python
  - Data Types
  - Type Conversions
- Operators in Python
  - Arthematic Operators
  - Logical Operators
  - Relational Operators
  - Assignment Operators
  - Bitwise Operators
  - Membership Operators
  - Identity Operators

print() --> Display output to the user

In [1]:
```
1 print("Hello World")
```

```
Hello World
```

In [2]:
```
1 print("Hello World", "Hello World2", "Hello world3")
```

```
Hello World Hello World2 Hello world3
```

In [4]:
```
1 print("Hello World", "Hello World2", "Hello world3", sep = ' :) ')
```

```
Hello World :) Hello World2 :) Hello world3
```

In [5]:
```
print("Hello")
print("World")
print("From")
print("APSSDC")
```

```
Hello
World
From
APSSDC
```

In [6]:
```
print("Hello", end = '-')
print("World",  end = '-')
print("From",  end = '-')
print("APSSDC",  end = '-')
```

```
Hello-World-From-APSSDC-
```

In [7]:
```
something = input("Enter something")
```

```
Enter something10
```

In [9]:
```
print(something)
```

```
10
```

In [10]:
```
print(type(something))
```

```
<class 'str'>
```

- Numbers - int, float, complex
- char - Strings
- Boolean - True, False
- list
- tuple
- dictionary
- sets

Type Casting/ Conversion

In [11]:
```
a = int(input("Enter a number: "))
```

```
Enter a number: 100
```

In [12]:
```
print(a)
```

```
100
```

In [13]:
```
1 type(a)
```

Out[13]:
```
int
```

In [14]:
```
1 a = int(input("Enter a number: "))
```

```
Enter a number: 10o

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-14-6c927185374a> in <module>
----> 1 a = int(input("Enter a number: "))

ValueError: invalid literal for int() with base 10: '10o'
```

Number Systems

- Decimal - 10
- Hexadecimal - 16
- Octal - 8
- Binary - 2

In [1]:
```
1 binary = bin(10)
2
3 print(binary, type(binary))
```

```
0b1010 <class 'str'>
```

In [2]:
```
1 hexa = hex(10)
2
3 print(hexa, type(hexa))
```

```
0xa <class 'str'>
```

In [3]:
```
1 octal = oct(10)
2
3 print(octal, type(octal))
```

```
0o12 <class 'str'>
```

In [5]:
```
bin2dec = int('1010', 2)
print(bin2dec, type(bin2dec))

oct2dec = int('1010', 8)
print(oct2dec, type(oct2dec))

hex2dec = int('1010', 16)
print(hex2dec, type(hex2dec))
```

```
10 <class 'int'>
520 <class 'int'>
4112 <class 'int'>
```

In [7]:
```
ord('A')
```

Out[7]: 65

In [8]:
```
ord('Z')
```

Out[8]: 90

In [9]:
```
chr(65)
```

Out[9]: 'A'

In [10]:
```
float(1), float('10')
```

Out[10]: (1.0, 10.0)

In [11]:
```
str(10), str(10.55), str(10+8j)
```

Out[11]: ('10', '10.55', '(10+8j)')

Identifiers

- It is a valid name in python which is given to Variables, Functions, Classes, Methods, Modules, Packages

Properties of Identifiers in Python

- identifiers cannot start with digit
- Identifiers can be Alphanumeric [A-Z, a-z, 0-9]
- It cannot contain any Special characters expect _
- it can start with _

In [13]:
```
_ = 50

print(_, type(_))
```

```
50 <class 'int'>
```

In [14]:
```
5a_ = 10
```

```
  File "<ipython-input-14-6d62420d0576>", line 1
    5a_ = 10
     ^
SyntaxError: invalid syntax
```

In [15]:
```
a5_ = 50

print(a5_)
```

```
50
```

In [16]:
```
a-v = 50

print(a-v)
```

```
  File "<ipython-input-16-469b4270f74b>", line 1
    a-v = 50
     ^
SyntaxError: cannot assign to operator
```

PEP8 Guidelines (https://realpython.com/python-pep8/#:~:text=The%20primary%20focus%20of%20PEP,and%20style%2C

Type	Naming Convention	Examples
Function	Use a lowercase word or words. Separate words by underscores to improve readability.	`function` , `my_function`
Variable	Use a lowercase single letter, word, or words. Separate words with underscores to improve readability.	`x` , `var` , `my_variable`
Class	Start each word with a capital letter. Do not separate words with underscores. This style is called camel case.	`Model` , `MyClass`
Method	Use a lowercase word or words. Separate words with underscores to improve readability.	`class_method` , `method`
Constant	Use an uppercase single letter, word, or words. Separate words with underscores to improve readability.	`CONSTANT` , `MY_CONSTANT` , `MY_LONG_CONSTANT`
Module	Use a short, lowercase word or words. Separate words with underscores to improve readability.	`module.py` , `my_module.py`
Package	Use a short, lowercase word or words. Do not separate words with underscores.	`package` , `mypackage`

Keywords in Python

They are reserve words in python which we can't use them as the identifiers

In [19]:
```
import keyword

key = keyword.kwlist

key, len(key)
```

Out[19]:
```
(['False',
  'None',
  'True',
  'and',
  'as',
  'assert',
  'async',
  'await',
  'break',
  'class',
  'continue',
  'def',
  'del',
  'elif',
  'else',
  'except',
  'finally',
  'for',
  'from',
  'global',
  'if',
  'import',
  'in',
  'is',
  'lambda',
  'nonlocal',
  'not',
  'or',
  'pass',
  'raise',
  'return',
  'try',
  'while',
  'with',
  'yield'],
 35)
```

Operators in Python

1. Arthematic Operators

- Addition: +

- Subtraction: -
- Multiplication: *
- Division: /
- Mudulus division: %
- Floor Division: //
- Power: **

In [20]:
```
a = 5
b = 9

print(a + b)
print(a - b)
print(a * b)
print(a / b)
print(a // b)
print(a % b)
print(a ** b)
```

```
14
-4
45
0.5555555555555556
0
5
1953125
```

Logical Operators

- and
- or
- not

In [21]:
```
a = 5
b = 10
c = 0
```

In [22]:
```
_
```

Out[22]: 50

inp1	inp2	inp1 and inp2	inp1 or inp2
0	0	0	0
1	0	0	1
0	1	0	1
1	1	1	1

In [24]:
```
1 a and b
```

Out[24]: 10

In [25]:
```
1 a and c
```

Out[25]: 0

In [26]:
```
1 c and b
```

Out[26]: 0

In [27]:
```
1 a or b
```

Out[27]: 5

In [28]:
```
1 c or a
```

Out[28]: 5

In [29]:
```
1 c or c
```

Out[29]: 0

In [32]:
```
1 a = 'APSSDC'
2 b = 50
3 zero = 0
4 zer = '0'
```

In [33]:
```
1 print(a and b, a and zero, zero and b, zer and a)
```

```
50 0 0 APSSDC
```

In [35]:
```
1 print(a or zer, zer or zero, zero or b, zer or a)
```

```
APSSDC 0 50 0
```

In [36]:
```
1 zero or zero
```

Out[36]: 0

In [37]:
```
1 not a
```

Out[37]: False

In [38]:
```
1 not zero
```

Out[38]: True

Relational Operators

- >
- <
- <=
- >=
- ==
- !=

In [39]:

```
a = 5
b = 10
c = 5

print(a > b) # False
print(a == c) # True
print(a < b) # True
print(a >= b) # False
print(a != c) # False
print(a <= b) # True
```

```
False
True
True
False
False
True
```

Day03 Outcomes

- Python Basics
  - print()
  - input()
  - Data Types
  - Type Conversion
- Identifiers
  - Properties of Identifiers
  - Identifier Guidelines
- Operators in Python
  - Arthematic
  - Logical
  - Relational