Python First Program

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

```
Hello World
```

In [2]:
```
import sys

# Print only the Python version as a string
print("Python version:", sys.version)
```

```
Python version: 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:26:23)
[MSC v.1916 64 bit (AMD64)]
```

Python Indentation

- Indentation refers to the spaces at the beginning of a code line.
- Python uses indentation to indicate a block of code.

In [3]:
```
#with Indentation
if 10 > 2:
    print("Ten is greater than two!")
```

```
Ten is greater than two!
```

In [4]:
```
# Without Indentation
if 5 > 2:
print("Five is greater than two!")
```

```
  Cell In[4], line 3
    print("Five is greater than two!")
    ^
IndentationError: expected an indented block after 'if' statement on line 2
```

Python Comments

- Comments can be used to explain Python code.
- Comments can be used to make the code more readable.
- Comments starts with a #, and Python will ignore them

In [5]:
```
#This is a comment
print("Hello, World!")
```

```
Hello, World!
```

In [6]:
```
print("Hello, World!") #This is a comment
```

```
Hello, World!
```

Multiline Comments

- Python does not really have a syntax for multiline comments.

- To add a multiline comment you could insert a # for each line:

In [7]:
```
#This is a comment
#written in
#more than just one line
print("Hello, World!")
```

```
Hello, World!
```

Python Identifiers

- A Python identifier is a name used to identify a variable, function, class, module or other object.
- An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).
- Python is a case sensitive programming language. Thus, 'Manpower' and 'manpower' are two different identifiers in Python.

Python keywords

- python keywords are Reserved Words
- you cannot use them as constant or variable or any other identifier names.
- All the Python keywords contain lowercase letters

In [8]:
```
import keyword

# Get the list of Python keywords
python_keywords = keyword.kwlist

# Display the list of keywords
print(python_keywords)

len(python_keywords)
```

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

Out[8]:
```
35
```

Python Variables

- Python variables are the reserved memory locations used to store values with in a Python Program.
- This means that when you create a variable you reserve some space in the memory.
- Variables are containers for storing data values.

In [9]:
```
# creating a Variable
```

```
a = 10
print(a)
# where 'a' is variable when is store a Value '10' by assign in it
```

```
10
```

Rules to Create a Variables

- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- A variable name cannot start with a number or any special character like $, (, * % etc.
- Python reserved keywords cannot be used naming the variable.
- Python variable names are case-sensitive which means Name and NAME are two different variables in Python.
- Variable name can't start with Digits

In [10]:
```
#vaild Variables

var = 13
var_3 = 'Python'
_var = 'Welcome to Data Science Course'
```

In [11]:
```
#invaild Variables

1_var = 565
@var = 'Data'
for = 'Python'
```

```
  Cell In[11], line 3
    1_var = 565
     ^
SyntaxError: invalid decimal literal
```

In [12]:
```
#You can get the data type of a variable with the type() function.
x = 57
y = "Data science"
print(type(x))
print(type(y))
```

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

Multiple Variables

In [13]:
```
x, y, z = "Apple", "Banana", "Cherry"

print(x)
print(y)
print(z)
```

```
Apple
Banana
Cherry
```

In [14]:
```
#One Value to Multiple Variables
x=y=z = 100
print(x)
print(y)
print(z)
```

```
100
100
100
```

Unpack a Collection¶

If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called unpacking.

In [15]:
```
fruits = ["apple", "banana", "cherry"]
x, y, z = fruits
print(x)
print(y)
print(z)
```

```
apple
banana
cherry
```

Python Data Types

- It defines what type of data we are going to store in a variable.
- The data stored in memory can be of many types.
- Python has various built-in data types

-- Data Type --> Examples

- Numeric --> int, float, complex
- String --> str
- Sequence --> list, tuple, range
- Mapping --> dict
- Boolean --> bool
- Set --> set,frozenset

Python Numbers

There are three numeric types in Python:

- int
- float
- complex

In [16]:
```
x = 1    # int
y = 2.8  # float
z = 1j   # complex
```

In [17]:
```
print(type(x))
print(type(y))
print(type(z))
```

```
<class 'int'>
<class 'float'>
<class 'complex'>
```

Type Conversion

- You can convert from one type to another with the int(), float(), and complex() methods
- Note: You cannot convert complex numbers into another number type.

In [18]:
```
x = 1    # int
y = 2.8  # float
z = 1j   # complex

#convert from int to float:
a = float(x)

#convert from float to int:
b = int(y)

#convert from int to complex:
c = complex(y)

print(a)
print(b)
print(c)

print(type(a))
print(type(b))
print(type(c))
```

```
1.0
2
(2.8+0j)
<class 'float'>
<class 'int'>
<class 'complex'>
```

Python Booleans

Booleans represent one of two values: True or False.

In [19]:
```
print(15 > 9)
print(13 == 9)
print(1 < 9)
```

```
True
False
True
```

Evaluate Values and Variables

The bool() function allows you to evaluate any value, and give you True or False in return,

In [20]:
```
print(bool("Hello"))
print(bool(98765))
```

```
True
True
```

Python Operators

Operators are used to perform operations on variables and values.

Python divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

Python Arithmetic Operators

- (addition)
- (subtraction)
- (multiplication)
- / (division)
- % (modulo - returns the remainder of the division)
- ** (exponentiation)

In [22]:
```
a = 10
b = 3
print(a + b)  # 13
print(a - b)  # 7
print(a * b)  # 30
print(a / b)  # 3.3333333333333335
print(a % b)  # 1
print(a ** b) # 1000
```

```
13
7
30
3.3333333333333335
1
1000
```

Comparison Operators: - == (equal to) - != (not equal to) - < (less than) - > (greater than) - <= (less than or equal to) - >= (greater than or equal to)

In [23]:
```
x = 5
y = 8
print(x == y)  # False
print(x != y)  # True
print(x < y)   # True
print(x > y)   # False
print(x <= y)  # True
print(x >= y)  # False
```

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

Logical Operators: and (logical AND) or (logical OR) not (logical NOT)

In [24]:
```
p = True
q = False
print(p and q)  # False
print(p or q)   # True
print(not p)    # False
```

```
False
True
False
```

Assignment Operators: = (assignment) += (add and assign) -= (subtract and assign) *= (multiply and assign) /= (divide and assign) %= (calculate modulo and assign) **= (exponentiate and assign)

In [25]:
```
x = 10
x += 5  # equivalent to x = x + 5
print(x)  # 15
```

```
15
```

Identity Operators: is (True if the operands are identical) is not (True if the operands are not identical)

In [26]:
```
a = [1, 2, 3]
b = a
print(a is b)  # True
```

```
True
```

Membership Operators: in (True if a value is found in the sequence) not in (True if a value is not found in the sequence)

In [27]:
```
myList = [1, 2, 3, 4]
print(3 in myList)    # True
print(5 not in myList) # True
```

```
True
True
```

Bitwise Operators: & (bitwise AND) | (bitwise OR) ^ (bitwise XOR) ~ (bitwise NOT) << (left shift) >> (right shift)

In [28]:
```
a = 5  # 0b0101 in binary
b = 3  # 0b0011 in binary
print(a & b)  # 1 (0b0001 in binary)
```

```
1
```

Precedence in python

Here is a general overview of operator precedence in Python, from highest to lowest precedence: Parentheses: () Exponentiation: ** Unary plus and minus: +x, -x Multiplication, division, and modulo: *, /, % Addition and subtraction: +, - Bitwise shift operators: <<, >> Bitwise AND, XOR, and OR: &, ^, | Comparison operators: ==, !=, <, >, <=, >=, is, is not Logical NOT: not Logical AND: and Logical OR: or Conditional expression (ternary operator): x if condition else y Assignment operators: =, +=, -=, *=, /=, %=, **=, &=, |=, ^=, <<=, >>=

In [29]:
```
result = 2 + 3 * 4  # Multiplication has higher precedence
# Result is 14 (2 + (3 * 4))

result = (2 + 3) * 4  # Parentheses override precedence
# Result is 20 ((2 + 3) * 4)
```

In [ ]:
```

```

Follow Us on Social Media

Linkedin: https://www.linkedin.com/company/eternaltek/about/?viewAsMember=true

Medium: https://medium.com/@eternaltek.info

WhatsApp Channel: https://whatsapp.com/channel/0029Va5onCbDjiOTi6D1vU36

In [31]:
```
!pip install nbconvert
```

```
Requirement already satisfied: nbconvert in c:\users\teks108\anaconda3\lib\site-packa
ges (6.5.4)
Requirement already satisfied: lxml in c:\users\teks108\anaconda3\lib\site-packages
(from nbconvert) (4.9.3)
Requirement already satisfied: beautifulsoup4 in c:\users\teks108\anaconda3\lib\site-
packages (from nbconvert) (4.12.2)
Requirement already satisfied: bleach in c:\users\teks108\anaconda3\lib\site-packages
(from nbconvert) (4.1.0)
Requirement already satisfied: defusedxml in c:\users\teks108\anaconda3\lib\site-pack
ages (from nbconvert) (0.7.1)
Requirement already satisfied: entrypoints>=0.2.2 in c:\users\teks108\anaconda3\lib\s
ite-packages (from nbconvert) (0.4)
Requirement already satisfied: jinja2>=3.0 in c:\users\teks108\anaconda3\lib\site-pac
kages (from nbconvert) (3.1.2)
Requirement already satisfied: jupyter-core>=4.7 in c:\users\teks108\anaconda3\lib\si
te-packages (from nbconvert) (5.3.0)
Requirement already satisfied: jupyterlab-pygments in c:\users\teks108\anaconda3\lib
\site-packages (from nbconvert) (0.1.2)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\teks108\anaconda3\lib\site
-packages (from nbconvert) (2.1.1)
Requirement already satisfied: mistune<2,>=0.8.1 in c:\users\teks108\anaconda3\lib\si
te-packages (from nbconvert) (0.8.4)
Requirement already satisfied: nbclient>=0.5.0 in c:\users\teks108\anaconda3\lib\site
-packages (from nbconvert) (0.5.13)
Requirement already satisfied: nbformat>=5.1 in c:\users\teks108\anaconda3\lib\site-p
ackages (from nbconvert) (5.9.2)
Requirement already satisfied: packaging in c:\users\teks108\anaconda3\lib\site-packa
ges (from nbconvert) (23.1)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\teks108\anaconda3\lib
\site-packages (from nbconvert) (1.5.0)
Requirement already satisfied: pygments>=2.4.1 in c:\users\teks108\anaconda3\lib\site
-packages (from nbconvert) (2.15.1)
Requirement already satisfied: tinycss2 in c:\users\teks108\anaconda3\lib\site-packag
es (from nbconvert) (1.2.1)
Requirement already satisfied: traitlets>=5.0 in c:\users\teks108\anaconda3\lib\site-
packages (from nbconvert) (5.7.1)
Requirement already satisfied: platformdirs>=2.5 in c:\users\teks108\anaconda3\lib\si
te-packages (from jupyter-core>=4.7->nbconvert) (3.10.0)
Requirement already satisfied: pywin32>=300 in c:\users\teks108\anaconda3\lib\site-pa
ckages (from jupyter-core>=4.7->nbconvert) (305.1)
Requirement already satisfied: jupyter-client>=6.1.5 in c:\users\teks108\anaconda3\li
b\site-packages (from nbclient>=0.5.0->nbconvert) (7.4.9)
Requirement already satisfied: nest-asyncio in c:\users\teks108\anaconda3\lib\site-pa
ckages (from nbclient>=0.5.0->nbconvert) (1.5.6)
Requirement already satisfied: fastjsonschema in c:\users\teks108\anaconda3\lib\site-
packages (from nbformat>=5.1->nbconvert) (2.16.2)
Requirement already satisfied: jsonschema>=2.6 in c:\users\teks108\anaconda3\lib\site
-packages (from nbformat>=5.1->nbconvert) (4.17.3)
Requirement already satisfied: soupsieve>1.2 in c:\users\teks108\anaconda3\lib\site-p
ackages (from beautifulsoup4->nbconvert) (2.4)
Requirement already satisfied: six>=1.9.0 in c:\users\teks108\anaconda3\lib\site-pack
ages (from bleach->nbconvert) (1.16.0)
Requirement already satisfied: webencodings in c:\users\teks108\anaconda3\lib\site-pa
ckages (from bleach->nbconvert) (0.5.1)
Requirement already satisfied: attrs>=17.4.0 in c:\users\teks108\anaconda3\lib\site-p
ackages (from jsonschema>=2.6->nbformat>=5.1->nbconvert) (22.1.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in c:\us
ers\teks108\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat>=5.1->nbconve
rt) (0.18.0)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\teks108\anaconda3\l
```

```
ib\site-packages (from jupyter-client>=6.1.5->nbclient>=0.5.0->nbconvert) (2.8.2)
Requirement already satisfied: pyzmq>=23.0 in c:\users\teks108\anaconda3\lib\site-pac
kages (from jupyter-client>=6.1.5->nbclient>=0.5.0->nbconvert) (23.2.0)
Requirement already satisfied: tornado>=6.2 in c:\users\teks108\anaconda3\lib\site-pa
ckages (from jupyter-client>=6.1.5->nbclient>=0.5.0->nbconvert) (6.3.2)
```

In [ ]: