
CodroidHub Summer Training

Title:-

Introduction of python

History of python

Python syntax

Features of python

Advantages & Disadvantages of python

Uses and applications of python

Difference between POPS and OOPS

Variable

Data types

Arithmetic operators

Logical operators

if-else condition

Match in python

Submitted by: YUDHVEER SINGH

ROLL NO: (2322016)

BRANCH: CSE

SUBMITTED TO: Mr. DEVASHISH KUMAR
(FOUNDER, CodroidHub Private Limited)

PYTHON

What is Python?

Python is a programming language that is interpreted, object-oriented, and considered to be high-level too. Python is a set of instructions that we give in the form of a Program to our computer to perform any specific task. It is a Programming language having properties like it is interpreted, object-oriented and it is high-level too. Due to its beginner-friendly syntax, it became a clear choice for beginners to start their programming journey. The major focus behind creating it is **making it easier for developers to read and understand, also reducing the lines of code.**

History of Python

Python was **created in 1980s by Guido van Rossum**. During his research at the **National Research Institute for Mathematics and Computer Science in the Netherlands**, he created Python – a super easy programming language in terms of reading and usage. The first ever version was released in the year 1991 which had only a few built-in data types and basic functionality.

Later, when it gained popularity among scientists for numerical computations and data analysis, in 1994, Python 1.0 was released with extra features like map, lambda, and filter functions. After that adding new functionalities and releasing newer versions of Python came into fashion.

- *Python 1.5 released in 1997*
- *Python 2.0 released in 2000*
- *Python 3.0 in 2008 brought newer functionalities*

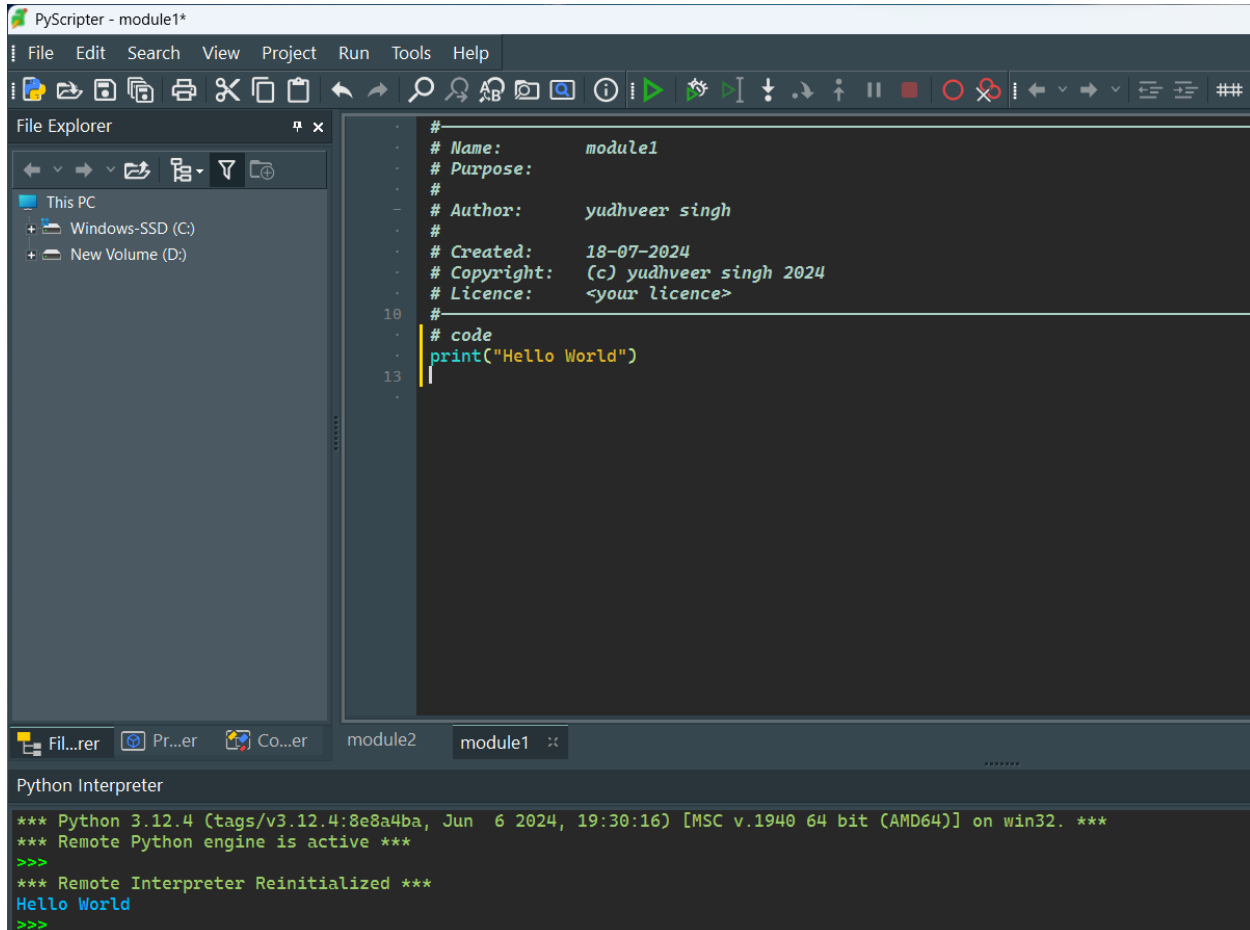
The latest version of Python, Python 3.12.4 – released on June 6, 2024.

Python Syntax

Syntax in a programming language is a standard way of expressing values or statements which every programming language follows.

To print a statement- `print("Hello World")`

Output: `Hello World`



The screenshot displays the PyScripter IDE interface. The top menu bar includes File, Edit, Search, View, Project, Run, Tools, and Help. Below the menu is a toolbar with various icons for file operations and execution. The left pane shows the File Explorer with 'This PC' selected, displaying 'Windows-SSD (C:)' and 'New Volume (D:)'. The main editor area shows a Python script with the following content:

```
#
# Name:      module1
# Purpose:
#
# Author:    yudhveer singh
#
# Created:   18-07-2024
# Copyright: (c) yudhveer singh 2024
# Licence:   <your licence>
10 #
11 # code
12 print("Hello World")
13
```

The bottom pane shows the Python Interpreter output:

```
*** Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [MSC v.1940 64 bit (AMD64)] on win32. ***
*** Remote Python engine is active ***
>>>
*** Remote Interpreter Reinitialized ***
Hello World
>>>
```

Features of Python

Python has plenty of features that make it the most demanding and popular. Let's read about a few of the best features that Python has:

- *Easy to read and understand*
- *Interpreted language*
- *Object-oriented programming language*
- *Free and open-source*
- *Versatile and Extensible*
- *Multi-platform*
- *Hundreds of libraries and frameworks*

- *Flexible, supports GUI*
- *Dynamically typed*
- *Huge and active community*

• **Advantages and Disadvantages of Python**

- Every programming language comes with benefits and limitations as well

Advantages of Python:

- *Easy to learn, read, and understand*
- *Versatile and open-source*
- *Improves productivity*
- *Supports libraries*
- *Huge library*
- *Strong community*
- *Interpreted language*

Disadvantages of Python:

- *Restrictions in design*
- *Memory inefficient*
- *Weak mobile computing*
- *Runtime errors*
- *Slow execution speed*

Uses and Applications of Python

1. Web Development

Developers prefer Python for web Development, due to its easy and feature-rich framework. They can create Dynamic websites with the best user experience using Python frameworks. Some of the frameworks are -Django, for Backend development and Flask, for Frontend development. Most internet companies, today are using Python framework as their core technology, because this is not only easy to

implement but is highly scalable and efficient. Web development is one of the top Applications of Python, which is widely used across the industry to create highly efficient websites.

2. Data Science

Data scientists can **build powerful AI models** using Python snippets. Due to its easily understandable feature, it allows developers to write complex algorithms. Data Science is used to create models and neural networks which can learn like human brains but are much faster than a single brain. It is used to extract patterns from past data and help organizations take their decisions. Also, companies use this field to make their future investments.

3. Web Scrapping and Automation

You can also automate your tasks using Python with libraries like [Beautiful Soup](#), [pandas](#), [matplotlib](#), etc. for **scrapping and web automation**. Businesses use AI bots as customer support to cater to the needs of the customers, it not only saves their money but also proved to be providing a better customer experience. Web scrapping helps the business in analyzing their data and other competitors' data to increase their share in the market. It will help the organizations, make their data organize and scale business by finding patterns from the scrapped data.

4. CAD

You can also use Python to work on **CAD (computer-aided designs) designs**, to create 2D and 3D models digitally. There is dedicated CAD software available in the market, but you can also develop CAD applications using Python also. You can develop a Python-based CAD application according to your customizability and complexity, depending on your project. Using Python for CAD development allows easy deployment and integration across cross-platforms.

5. Artificial Intelligence and Machine Learning

Using libraries like Pandas, and TensorFlow, experts can work on **data analysis and machine learning applications** for statistical analysis, data manipulation, etc. Python is one of the most used Programming languages in this field. It is worth saying that Python is the language of AI and ML. Python has contributed a lot to this field with its huge collection of libraries and large community support. Also, the field of Artificial intelligence and Machine learning is exponentially evolving, hence the use of Python is also going to increase a lot.

6. Game Development

Python can also be used by developers to **build games** using Pygame to develop 2D and 3D games. Some of the popular games built using Python are Pirates of the Caribbean, Battlefield 2, etc. Python has a library named **Pygame**, which is used to build interesting games. Since the gaming industry is gaining a lot market in recent years the use of these kinds of development has increased a lot in recent past. Also, it is very easy to build games using this library, you can also try to build some basic games

Conclusion

Python has a lot of reasons which make it a more popular and highly demanding programming language. High Community support and a large number of libraries and frameworks in Python make it the best choice for developers and beginners to choose it single handily. Python has use cases in web Development, Game Development, Automation, and technologies like AI, ML, and Data Analytics. Python is releasing its never version and adding new functions for the betterment of developers. But it has some limitations as well as discussed in the article like it is slow in execution, so Competitive Programmer prefer it less. But overall it is growing rapidly and has a very bright future ahead for this Programming language.

Difference between POPS and OOPS

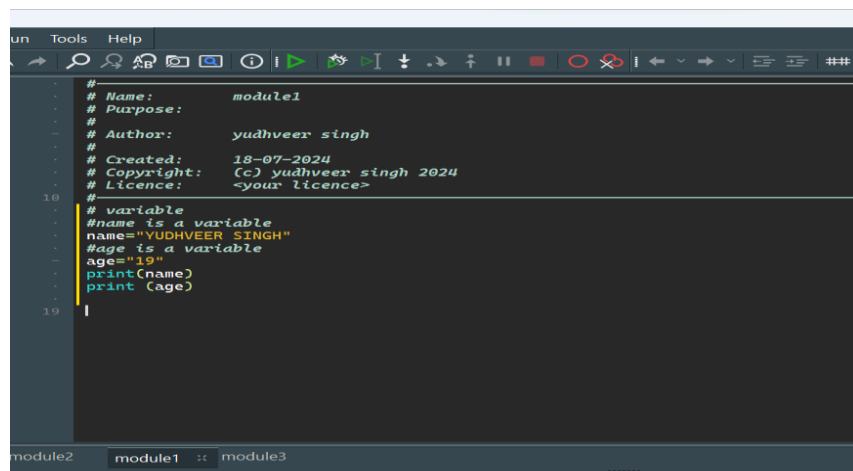
Feature	POPS (Point of Purchase Systems)	OOPS (Order of Payment Systems)
Purpose	Designed for retail environments to facilitate the sale of goods to customers.	Used in various businesses to manage and track payments and orders.
Main Functionality	Includes inventory management, sales tracking, and customer management.	Focuses on payment processing, invoicing, and order management.
User Interface	Often has a graphical user interface suitable for retail employees.	May have a more complex interface for managing orders and payments.
Integration	Typically integrated with barcode scanners, receipt printers, and cash drawers.	Often integrated with accounting software and online payment gateways.
Examples of Use	Retail stores, supermarkets, and convenience stores.	E-commerce platforms, subscription services, and service-based businesses.
Payment Handling	Handles immediate payment transactions at the point of sale.	Manages various payment methods, including deferred payments and invoices.
Inventory Management	Integral part of the system, tracks stock levels in real-time.	May include basic inventory features but not as robust as POPS.

Feature	POPS (Point of Purchase Systems)	OOPS (Order of Payment Systems)
Customer Interaction	Direct interaction with customers during the purchase process.	May have limited direct customer interaction, focuses more on backend operations.
Reporting	Generates sales reports, inventory levels, and customer purchase history.	Provides reports on payments received, outstanding invoices, and order statuses.
Security	Emphasizes secure transactions at the point of sale.	Focuses on secure handling of payment information and order processing.

Variables

A variable is used to store data that can be referenced and manipulated in your code. Variables can hold different types of data, such as numbers, strings, lists, or even complex objects. Here's a basic overview of variables in Python:

Example:-



```

# Name:      module1
# Purpose:
#
# Author:    yudhveer singh
#
# Created:   18-07-2024
# Copyright: (c) yudhveer singh 2024
# Licence:   <your licence>
10
# variable
#name is a variable
name="YUDHVEER SINGH"
#age is a variable
age="19"
print(name)
print (age)
19

```

Outpt:-YUDHVEER SINGH

19

Variable Naming Rules :-

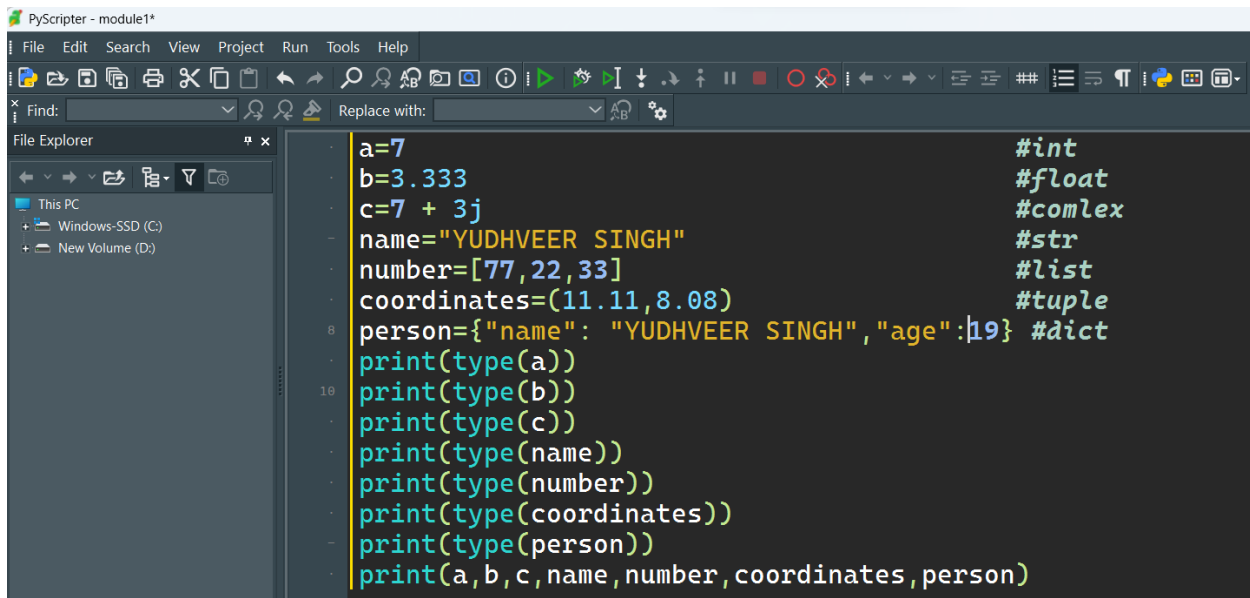
- Variable names must start with a letter (a-z, A-Z) or an underscore (_).
- The rest of the name can contain letters, numbers, or underscores.

- Variable names are case-sensitive (age and Age are different variables).
- Avoid using Python reserved words (keywords) as variable names (e.g., if, else, for, while).

Data types

Data types are classifications that dictate how the interpreter will treat and store different kinds of data. Here are the primary built-in data types in Python.

Example:-



```
PyScripter - module1*
File Edit Search View Project Run Tools Help
Find: Replace with:
File Explorer
This PC
Windows-SSD (C:)
New Volume (D:)

a=7 #int
b=3.333 #float
c=7 + 3j #complex
name="YUDHVEER SINGH" #str
number=[77,22,33] #list
coordinates=(11.11,8.08) #tuple
8 person={"name": "YUDHVEER SINGH", "age":19} #dict
print(type(a))
10 print(type(b))
print(type(c))
print(type(name))
print(type(number))
print(type(coordinates))
print(type(person))
print(a,b,c,name,number,coordinates,person)
```


Output:-

```

Python Interpreter

*** Remote Interpreter Reinitialized ***
<class 'int'>
<class 'float'>
<class 'complex'>
<class 'str'>
<class 'list'>
<class 'tuple'>
<class 'dict'>
7 3.333 (7+3j) YUDHVEER SINGH [77, 22, 33] (11.11, 8.08) {'name': 'YUDHVEER SINGH', 'age': 19}
>>>

```

Arithmetic operator

Arithmetic operators in Python are used to perform basic mathematical operations. Here is a list of the arithmetic operators available in Python along with examples:

Example:-

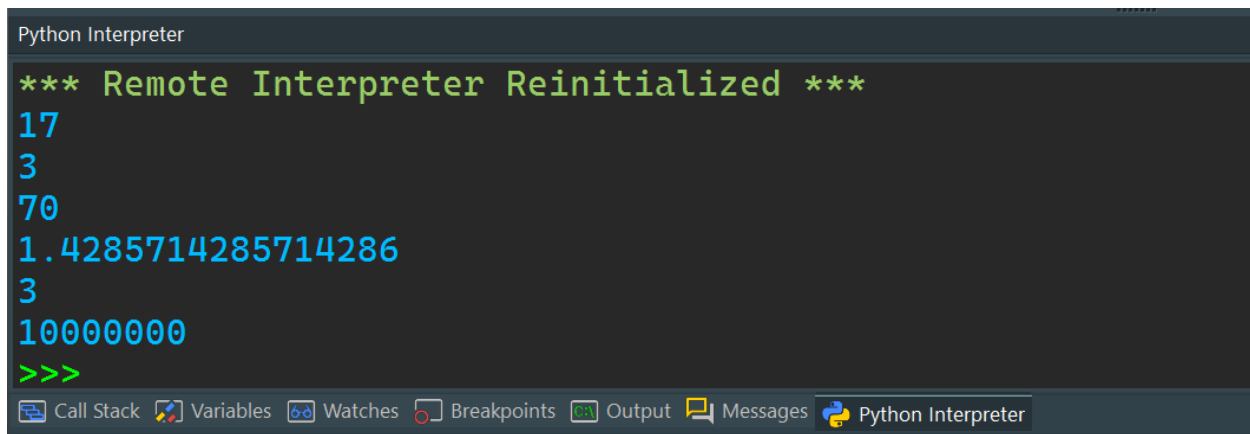
```

PyScripter - module1*
File Edit Search View Project Run Tools Help
Find:
File Explorer
This PC
Windows-SSD (C:)
New Volume (D:)

#author=YUDHVEER SINGH
a=10
b=7
sum=a+b
diff=a-b
mult=a*b
div=a/b
mod=a%b
exponentiation=a**b
10 print(sum)
   print(diff)
   print(mult)
   print(div)
   print(mod)
15 print(exponentiation)

```

Output:-



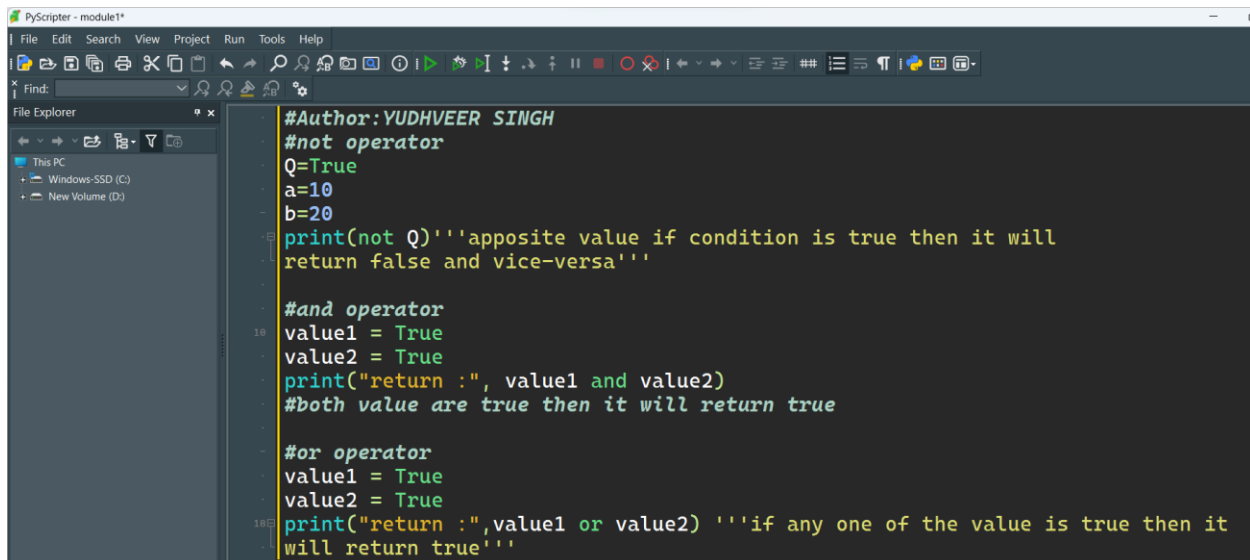
```
Python Interpreter

*** Remote Interpreter Reinitialized ***
17
3
70
1.4285714285714286
3
10000000
>>>
```

Logical operator

logic operator are used to combine multiple condition together and evaluate them as a single Boolean expression. There are three types of logical operators in python: and, or, not.

Example:-



```
PyScripter - module1*

File Explorer
  This PC
  Windows-SSD (C:)
  New Volume (D:)

#Author: YUDHVEER SINGH
#not operator
Q=True
a=10
b=20
print(not Q)'''apposite value if condition is true then it will
return false and vice-versa'''

#and operator
value1 = True
value2 = True
print("return :", value1 and value2)
#both value are true then it will return true

#or operator
value1 = True
value2 = True
print("return :",value1 or value2) '''if any one of the value is true then it
will return true'''
```

Output:-

```
Python Interpreter
*** Remote Interpreter Reinitialized ***
False
return : True
return : True
>>>
```

Call Stack Variables Watches Breakpoints Output Messages Python Interpreter

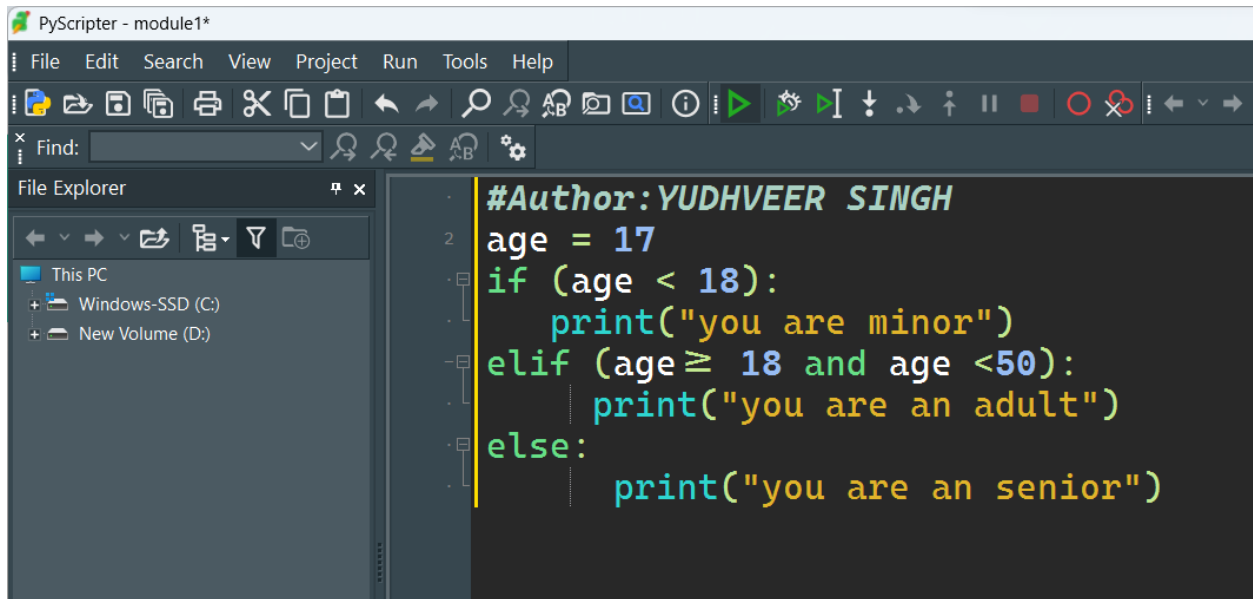
if-else condition

The if, elif, and else statements are used to perform conditional logic. Here's a basic example to illustrate how these statements are used:

Here's a breakdown of each component:

1. ****if Statement****: The if statement evaluates a condition. If the condition is True, the block of code under the if statement is executed.
2. ****elif Statement****: The elif (short for "else if") statement is used to check multiple expressions for True and execute a block of code as soon as one of the conditions is True. You can have multiple elif statements.
3. ****else Statement****: The else statement is optional and catches anything which isn't caught by the preceding conditions. If none of the previous conditions are True, the block of code under the else statement is executed.

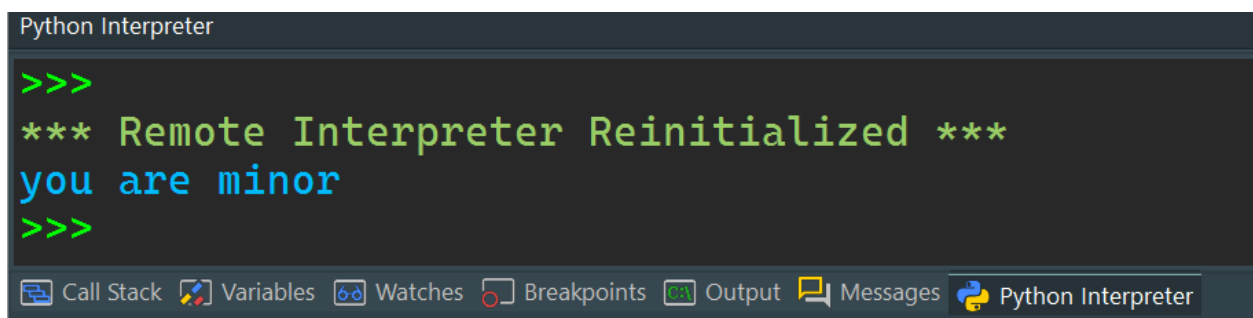
Example:-



The image shows a PyScripter window titled "PyScripter - module1*". The window has a menu bar (File, Edit, Search, View, Project, Run, Tools, Help) and a toolbar with various icons. A File Explorer pane on the left shows the file structure. The main editor area contains the following Python code:

```
#Author:YUDHVEER SINGH
age = 17
if (age < 18):
    print("you are minor")
elif (age ≥ 18 and age <50):
    print("you are an adult")
else:
    print("you are an senior")
```

Output:-



The image shows a Python Interpreter window. The output is as follows:

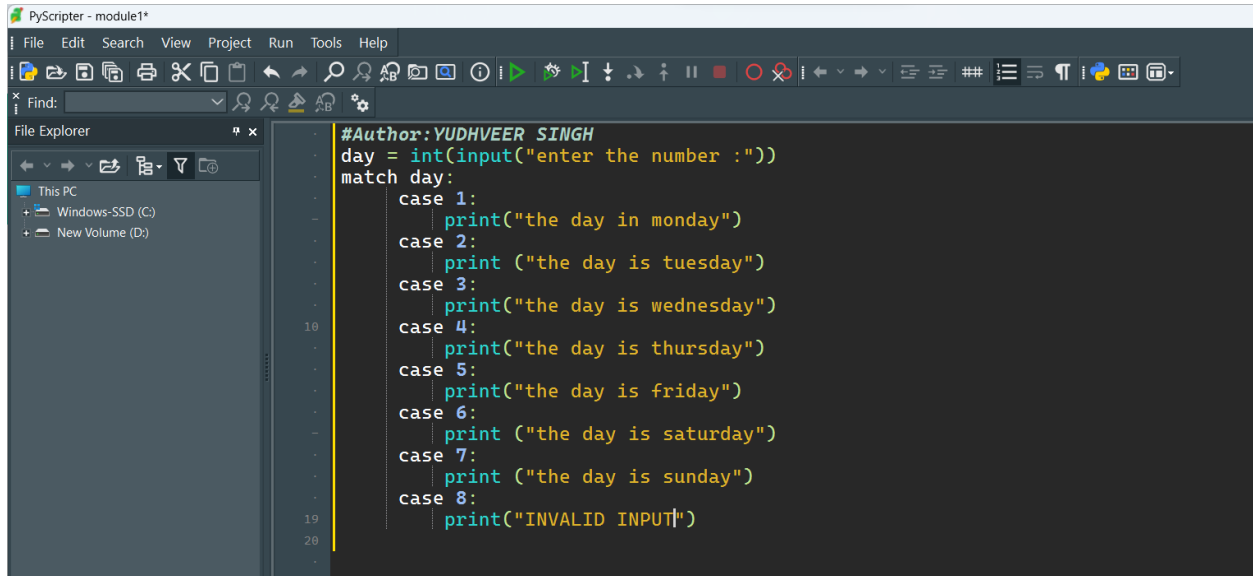
```
>>>
*** Remote Interpreter Reinitialized ***
you are minor
>>>
```

The bottom of the window features a toolbar with icons for Call Stack, Variables, Watches, Breakpoints, Output, Messages, and the Python Interpreter itself.

Match case statement in python

In Python 3.10 and later, the match statement (similar to switch-case statements in other languages) is introduced for pattern matching. It allows for more readable and concise handling of multiple conditions.

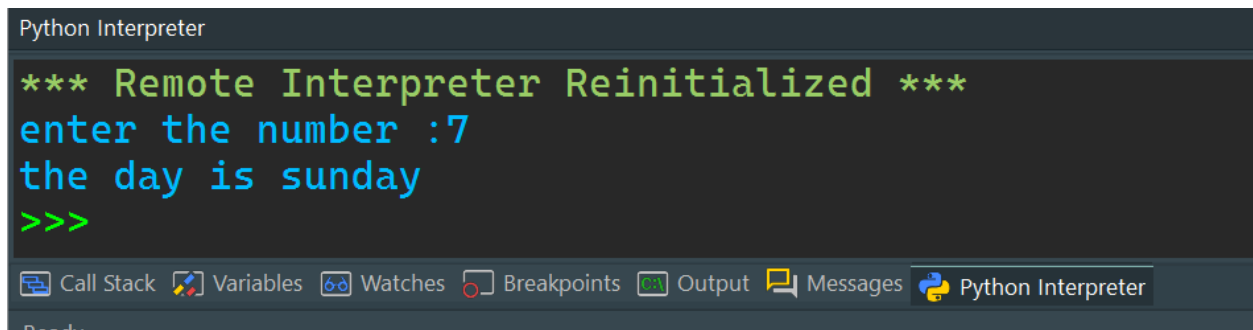
Example:-



The image shows the PyScripter IDE interface. On the left is a File Explorer showing 'This PC', 'Windows-SSD (C:)', and 'New Volume (D:)'. The main editor displays a Python script with a match statement. The script is as follows:

```
#Author:YUDHVEER SINGH
day = int(input("enter the number :"))
match day:
    case 1:
        print("the day in monday")
    case 2:
        print ("the day is tuesday")
    case 3:
        print("the day is wednesday")
    case 4:
        print("the day is thursday")
    case 5:
        print("the day is friday")
    case 6:
        print ("the day is saturday")
    case 7:
        print ("the day is sunday")
    case 8:
        print("INVALID INPUT")
```

Output:-



The image shows the Python Interpreter window. It displays the output of the script. The text is as follows:

```
*** Remote Interpreter Reinitialized ***
enter the number :7
the day is sunday
>>>
```

At the bottom, there is a toolbar with icons for Call Stack, Variables, Watches, Breakpoints, Output, Messages, and Python Interpreter. The Python Interpreter tab is currently selected.