CodroidHub Summer Training

Index

SUBMITTED BY: ASMIT

ROLL NO: (2322811)

BRANCH: AI&ML

SUBMITTED TO: Mr. DEVASHISH SIR

(FOUNDER, CodroidHub Private Limited)

PYTHON

What is Python ?

Python is a high-level, interpreted programming language known for its readability, simplicity, and versatility. It was created by Guido van Rossum and first released in 1991.

Application :

- **Web Development:** Frameworks like Django and Flask are popular for building web applications.
- **Data Science and Machine Learning:** Libraries such as NumPy, pandas, matplotlib, SciPy, and scikit-learn make Python a preferred choice for data analysis, visualization, and machine learning.
- **Automation and Scripting:** Python is widely used for automating repetitive tasks and writing scripts for system administration.
- **Software Development:** Python can be used for backend development, desktop applications, and game development.
- **Scientific Computing:** Python is used extensively in scientific research and simulations due to its simplicity and powerful libraries like SciPy and SymPy.

Features:

- **Easy to Learn and Use**: Python has a simple syntax similar to English, which makes it easy to learn and write code quickly.
- **Interpreted Language**: Python code is executed line by line, which makes debugging easier and allows for interactive testing and experimentation.
- **Dynamically Typed**: You don't need to declare the data type of a variable; Python automatically determines the type at runtime.
- **High-Level Language**: Python abstracts away many of the complexities involved in programming, such as memory management, making it easier to focus on problem-solving.
- **Extensive Standard Library**: Python comes with a rich standard library that provides modules and functions for tasks like file I/O, system calls, web development, and more.
- **Object-Oriented**: Python supports object-oriented programming (OOP) paradigms, allowing for the creation of reusable and modular code.
- **Readable and Maintainable Code**: Python's emphasis on readability and clean code ensures that programs are easier to maintain and update.

Difference between OOPS and POPS

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.
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.

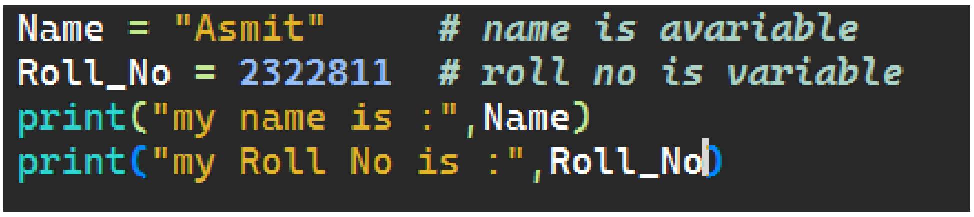
Variables

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.

Variable Naming Rules :-

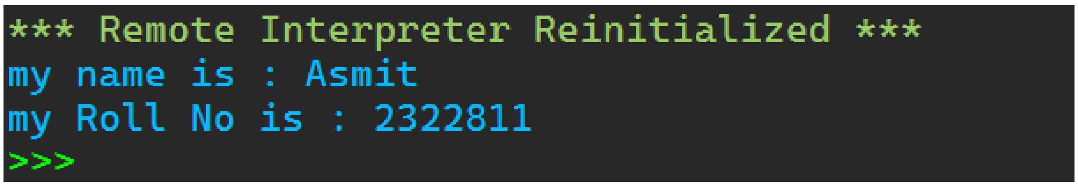
- Variable names must start with a letter (a-z, A-Z) or an underscore (_).
- The rest of the name can contain letters, numbers, or underscores.
- Variable names are case-sensitive (age and Age are different variables).
- Avoid using Python reserved words (keywords) as variable names (e.g., if, else, for, while).

Example:



```
Name = "Asmit"      # name is avariable
Roll_No = 2322811   # roll no is variable
print("my name is :",Name)
print("my Roll No is :",Roll_No)
```

Output:

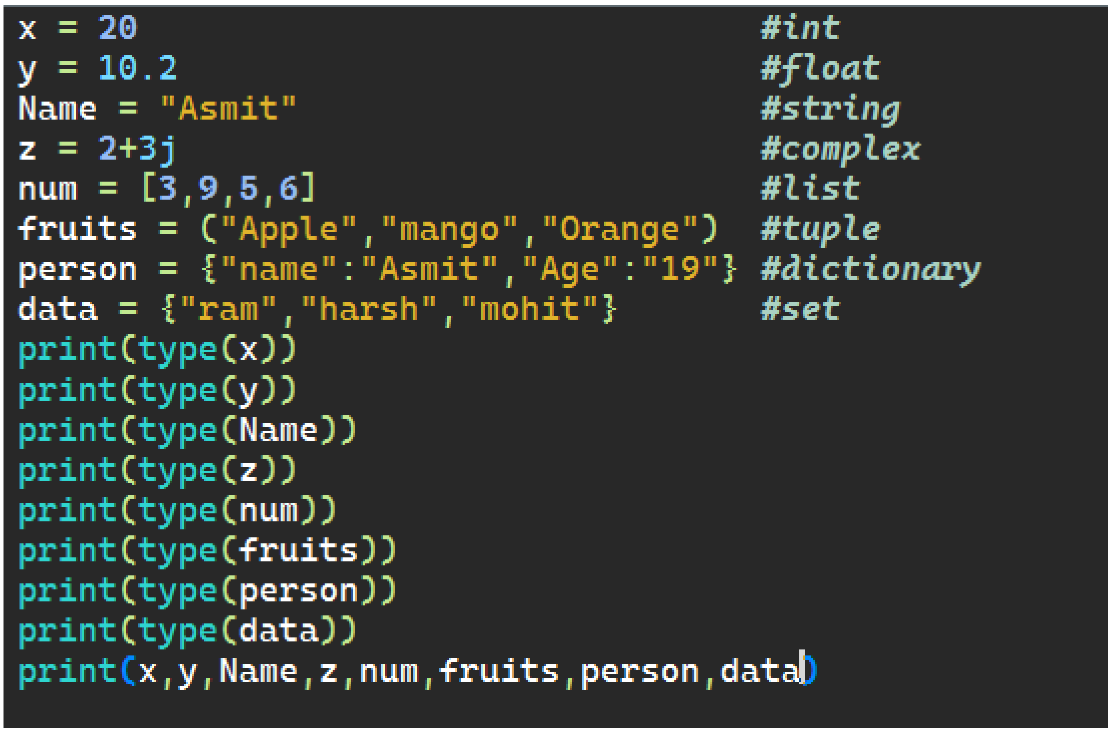


```
*** Remote Interpreter Reinitialized ***
my name is : Asmit
my Roll No is : 2322811
>>>
```

Data Types

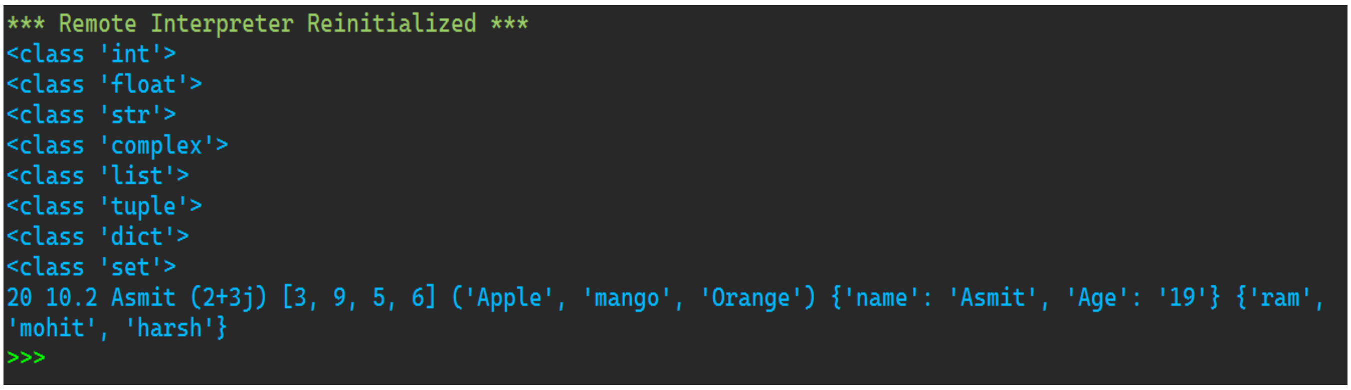
Data Types are classifications that specify the type of value a variable can hold.

Example:



```
x = 20                                   #int
y = 10.2                                 #float
Name = "Asmit"                           #string
z = 2+3j                                 #complex
num = [3,9,5,6]                          #list
fruits = ("Apple","mango","Orange")      #tuple
person = {"name":"Asmit","Age":"19"}     #dictionary
data = {"ram","harsh","mohit"}           #set
print(type(x))
print(type(y))
print(type(Name))
print(type(z))
print(type(num))
print(type(fruits))
print(type(person))
print(type(data))
print(x,y,Name,z,num,fruits,person,data)
```

Output:

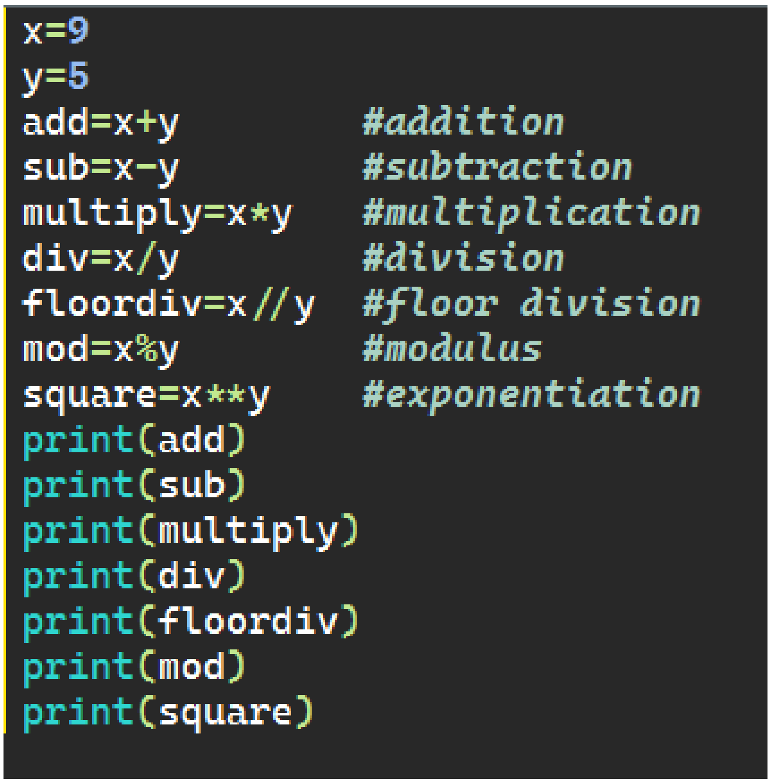


```
*** Remote Interpreter Reinitialized ***
<class 'int'>
<class 'float'>
<class 'str'>
<class 'complex'>
<class 'list'>
<class 'tuple'>
<class 'dict'>
<class 'set'>
20 10.2 Asmit (2+3j) [3, 9, 5, 6] ('Apple', 'mango', 'Orange') {'name': 'Asmit', 'Age': '19'} {'ram',
'mohit', 'harsh'}
>>>
```

Arithmetic Operator

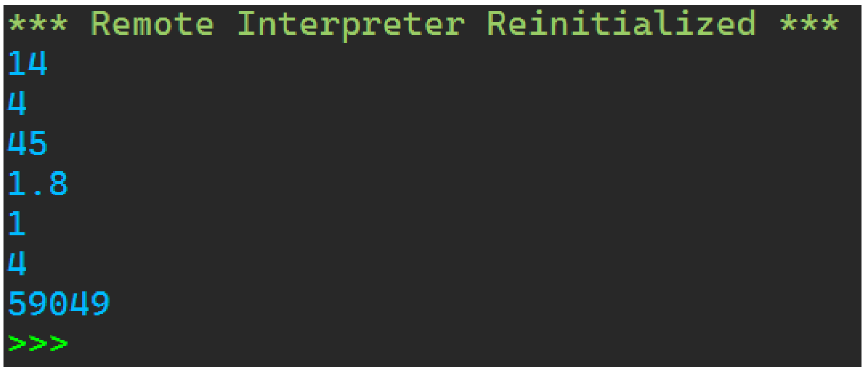
Arithmetic operators are used to perform mathematical operations. Here are the basic arithmetic operators:

Example:



```
x=9
y=5
add=x+y           #addition
sub=x-y           #subtraction
multiply=x*y      #multiplication
div=x/y           #division
floordiv=x//y     #floor division
mod=x%y           #modulus
square=x**y       #exponentiation
print(add)
print(sub)
print(multiply)
print(div)
print(floordiv)
print(mod)
print(square)
```

Output;

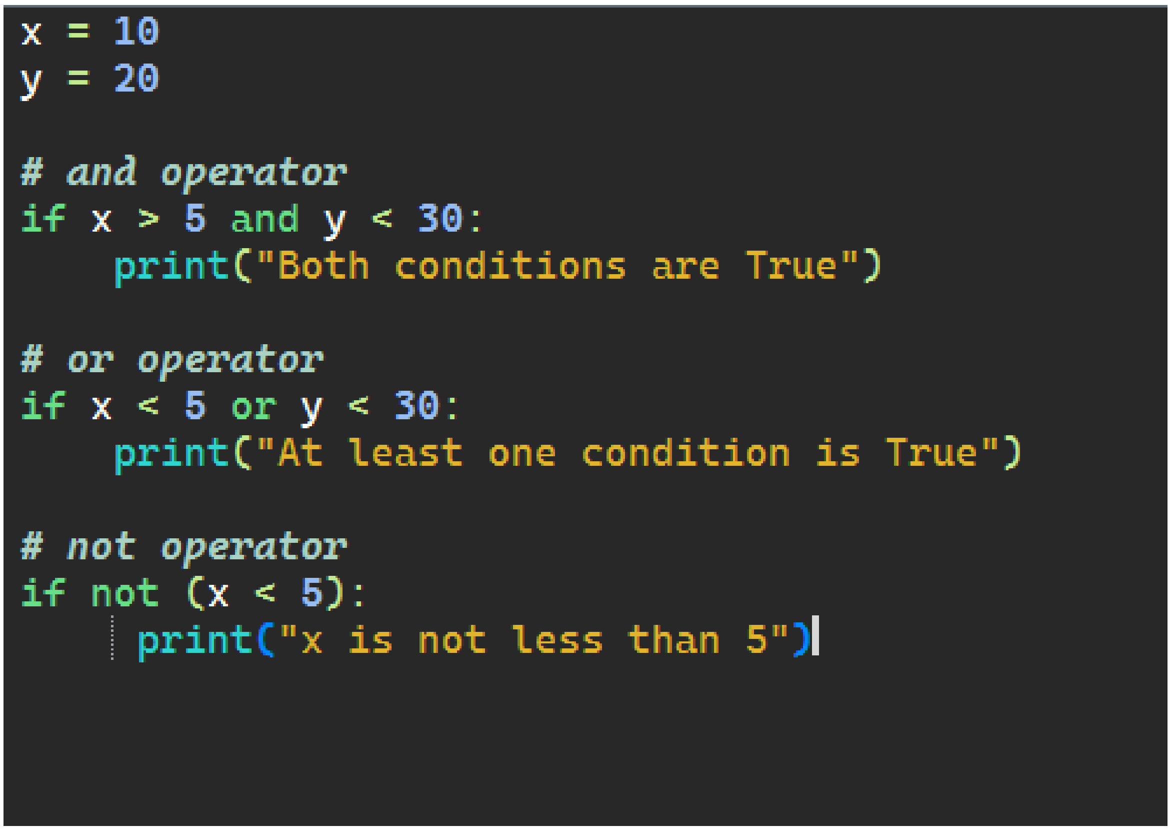


```
*** Remote Interpreter Reinitialized ***
14
4
45
1.8
1
4
59049
>>>
```

Logical operator

Python logical operators are used to combine conditional statements , allowing you to perform operations based on multiple conditions.

Example:



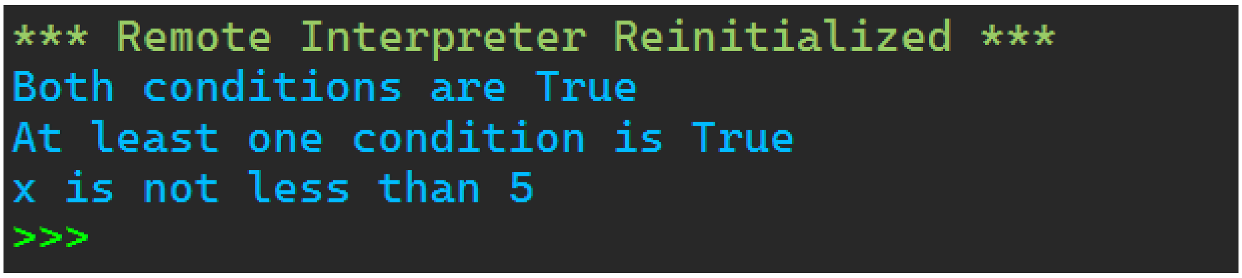
```python
x = 10
y = 20

# and operator
if x > 5 and y < 30:
    print("Both conditions are True")

# or operator
if x < 5 or y < 30:
    print("At least one condition is True")

# not operator
if not (x < 5):
    print("x is not less than 5")
```

Output:



```
*** Remote Interpreter Reinitialized ***
Both conditions are True
At least one condition is True
x is not less than 5
>>>
```

Conditional Statements

Conditional Statements are statements that provide a choice for the control flow based on a condition. It means that the control flow of the program will be decided based on the outcome of the condition.

If statement:

The if statement is used to test a specific condition. If the condition is True, the block of code following the if statement is executed.

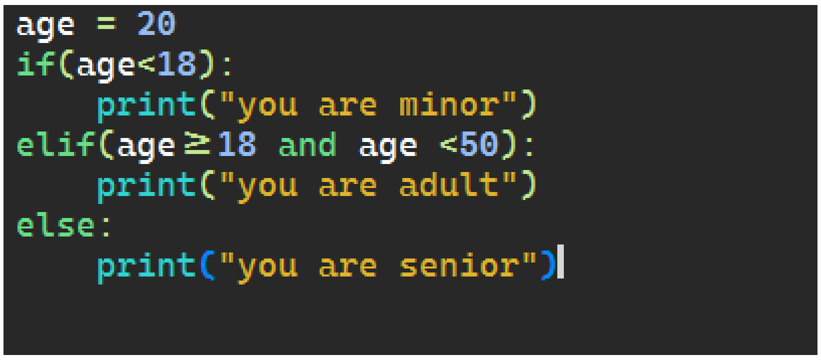
Elif statement:

The elif statement allows for multiple conditions to be checked in sequence. If the if condition is False, the elif condition is evaluated. If the elif condition is True, the associated block of code is executed.

`Else` statement:

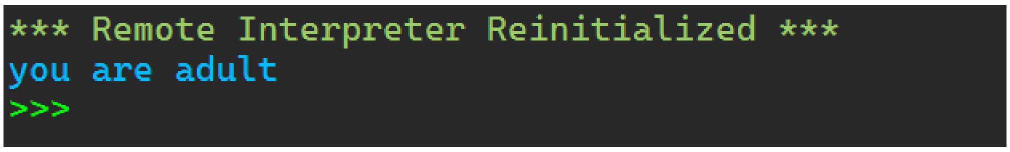
The `else` statement follows an `if` statement and executes a block of code if the `if` condition evaluates to False.

Example:



```
age = 20
if(age<18):
    print("you are minor")
elif(age≥18 and age <50):
    print("you are adult")
else:
    print("you are senior")
```

Output:

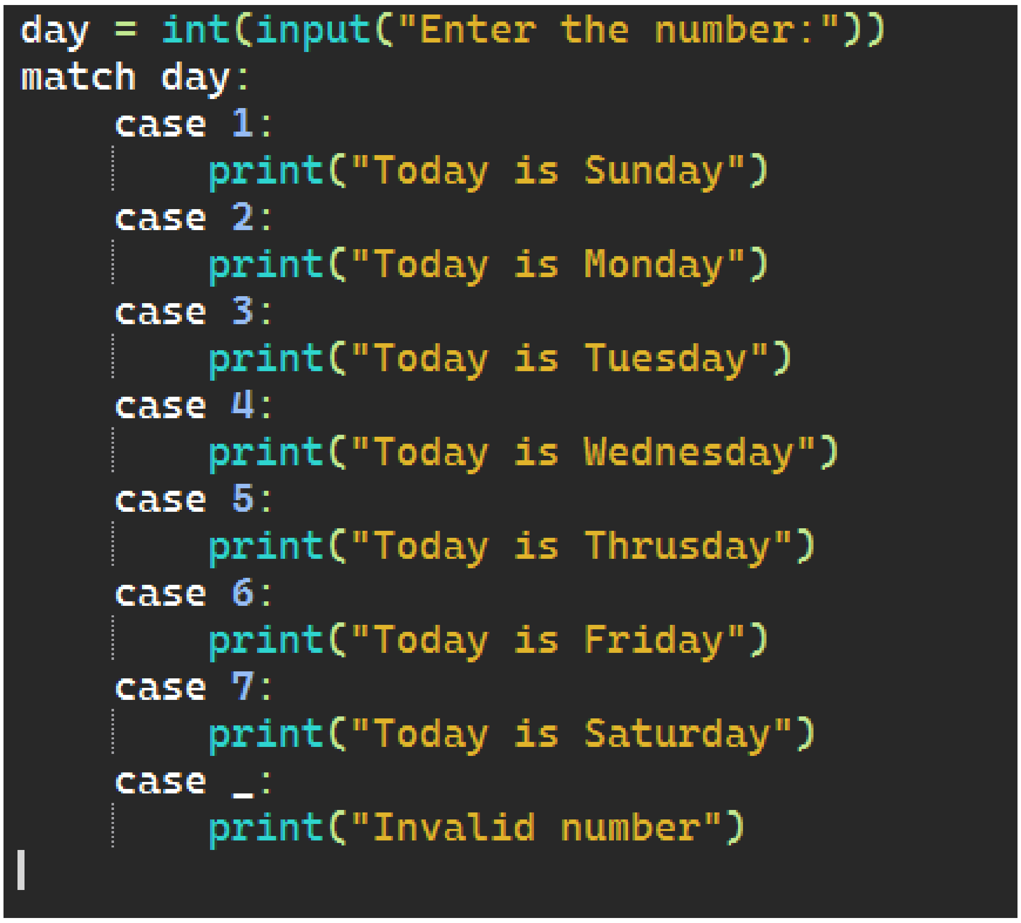


```
*** Remote Interpreter Reinitialized ***
you are adult
>>>
```

Match Statement in Python

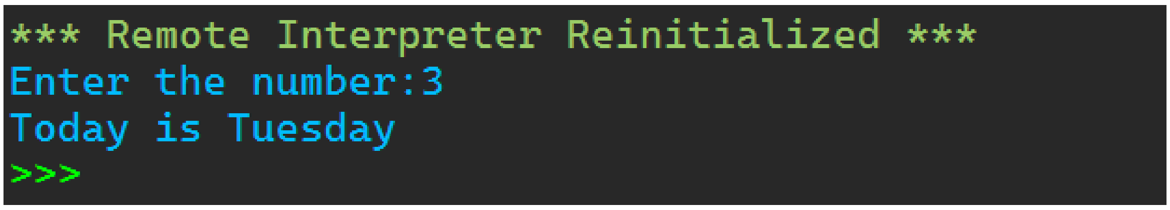
The match case statement in Python is initialized with the match keyword followed by the parameter to be matched. Then various cases are defined using the case keyword and the pattern to match the parameter. The “_” is the wildcard character that runs when all the cases fail to match the parameter value.

Example:



```python
day = int(input("Enter the number:"))
match day:
    case 1:
        print("Today is Sunday")
    case 2:
        print("Today is Monday")
    case 3:
        print("Today is Tuesday")
    case 4:
        print("Today is Wednesday")
    case 5:
        print("Today is Thrusday")
    case 6:
        print("Today is Friday")
    case 7:
        print("Today is Saturday")
    case _:
        print("Invalid number")
```

Output:



```
*** Remote Interpreter Reinitialized ***
Enter the number:3
Today is Tuesday
>>>
```