

```
# Single line comment

print("I'm Jahid", end=" ")
print("I am a Python Programmer.")
print("I love coding")

'''

long_variable_name = 10
print(long_variable_name)
Multi-line comment
'''


number = 7
print("The number is", number)

'''

Python's built-in Data Types:
-----
01. Numeric Types
-----
These types are used to represent numbers.
A. int: Integers (e.g., 5, -100).
B. float: Floating-point numbers (e.g., 3.14, -0.5).
C. complex: Complex numbers (e.g., 1 + 2j).

02. Sequence Types
-----
These types represent ordered collections of items.
A. str: Strings (a sequence of characters). It's immutable.
B. list: Lists (an ordered, mutable collection).
C. tuple: Tuples (an ordered, immutable collection).
D. range: A sequence of numbers often used for looping. It's immutable.

03. Mapping Type
-----
This type is a collection of key-value pairs.
A. dict: Dictionaries (an unordered, mutable collection of key-value pairs).

04. Set Types
-----
These types are unordered collections of unique items.
A. set: Sets (a mutable collection).
B. frozenset: Frozen sets (an immutable collection).

05. Boolean Type
-----
This type represents logical values.
A. bool: Booleans (either True or False).

06. Binary Types
-----
These types handle binary data.
A. bytes: An immutable sequence of bytes.
B. bytearray: A mutable sequence of bytes.
C. memoryview: Provides an efficient way to access the memory of another binary object without copying it.

07. None Type
This type represents the absence of a value.
A. NoneType: The single object of this type is None.

08. Custom Types: User-defined classes (if applicable)
'''


print("The Data Type of", number, "is", type(number))

user_input = input("Please enter your favorite programming language: ")
print("You entered:", user_input)
```

পাইথনের অপারেটর (Python Operators)

অপারেটর হলো এমন কিছু বিশেষ প্রতীক বা চিহ্ন যা কোনো অপারেশন বা কাজ সম্পাদনের জন্য ব্যবহৃত হয়। এই অপারেটরগুলো ডেটা বা ভ্যারিয়েবলের ওপর কাজ করে।

পাইথনে বিভিন্ন ধরনের অপারেটর রয়েছে। নিচে এদের বিস্তারিত আলোচনা করা হলো:

১. অ্যারিথমেটিক অপারেটর (Arithmetic Operators)

এই অপারেটরগুলো সাধারণ গাণিতিক হিসাব-নিকাশের জন্য ব্যবহৃত হয়।

অপারেটর	নাম	উদাহরণ	ফলাফল
+	যোগ (Addition)	$5 + 3$	8
-	বিয়োগ (Subtraction)	$10 - 4$	6
*	গুণ (Multiplication)	$6 * 2$	12
/	ভাগ (Division)	$15 / 3$	5.0
%	মডিউলাস (Modulus)	$10 \% 3$	1 (ভাগশেষ)
**	এক্সপোনেনশিয়াল (Exponentiation)	$2 ** 3$	8 (২-এর কিউব)
//	ফ্লোর ডিভিশন (Floor Division)	$10 // 3$	3 (পূর্ণসংখ্যার ভাগফল)

২. অ্যাসাইনমেন্ট অপারেটর (Assignment Operators)

এই অপারেটরগুলো ভ্যারিয়েবলের মধ্যে কোনো মান বা ডেটা রাখতে ব্যবহৃত হয়।

অপারেটর	উদাহরণ	একই কাজ	ফলাফল
=	<code>x = 5</code>		<code>x</code> -এর মান 5 হবে।
+=	<code>x += 3</code>	<code>x = x + 3</code>	<code>x</code> -এর মান 8 হবে।
-=	<code>x -= 2</code>	<code>x = x - 2</code>	<code>x</code> -এর মান 3 হবে।
*=	<code>x *= 4</code>	<code>x = x * 4</code>	<code>x</code> -এর মান 20 হবে।
/=	<code>x /= 2</code>	<code>x = x / 2</code>	<code>x</code> -এর মান 2.5 হবে।
%=	<code>x %= 3</code>	<code>x = x % 3</code>	<code>x</code> -এর মান 1 হবে।
**=	<code>x **= 2</code>	<code>x = x ** 2</code>	<code>x</code> -এর মান 25 হবে।
//=	<code>x //= 3</code>	<code>x = x // 3</code>	<code>x</code> -এর মান 1 হবে।

৩. কম্প্যারিসন অপারেটর (Comparison Operators)

এই অপারেটরগুলো দুটি মান বা ডেটার মধ্যে তুলনা করতে ব্যবহৃত হয় এবং ফলাফল হিসেবে ট্রু বা ফলস (True or False) দেয়।

অপারেটর	নাম	উদাহরণ	ফলাফল
<code>==</code>	সমান (Equal)	<code>5 == 5</code>	True
<code>!=</code>	অসমান (Not equal)	<code>5 != 3</code>	True
<code>></code>	বড় (Greater than)	<code>10 > 7</code>	True
<code><</code>	ছোট (Less than)	<code>10 < 7</code>	False
<code>>=</code>	বড় অথবা সমান (Greater than or equal to)	<code>8 >= 8</code>	True
<code><=</code>	ছোট অথবা সমান (Less than or equal to)	<code>8 <= 5</code>	False

৪. লজিক্যাল অপারেটর (Logical Operators)

এই অপারেটরগুলো একাধিক শর্ত বা কন্ডিশনকে একত্রিত করে ফলাফল দেয়।

অপারেটর	বর্ণনা	উদাহরণ	ফলাফল
and	যদি দুটি কন্ডিশনই সত্য হয়, তাহলে True হবে।	$(5 > 3) \text{ and } (10 > 8)$	True
or	যদি দুটি কন্ডিশনের মেকোনো একটি সত্য হয়, তাহলে True হবে।	$(5 > 3) \text{ or } (10 < 8)$	True
not	সত্যকে মিথ্যা এবং মিথ্যাকে সত্য করে দেয়।	<code>not(10 > 5)</code>	False

৫. বিটওয়াইজ অপারেটর (Bitwise Operators)

এই অপারেটরগুলো সংখ্যার বিট (0 এবং 1) নিয়ে কাজ করে। সাধারণত, অ্যাডভান্সড প্রোগ্রামিংয়ে এদের ব্যবহার দেখা যায়।

অপারেটর	নাম
&	Bitwise AND
	Bitwise OR
^	Bitwise XOR
~	Bitwise NOT
<<	Bitwise left shift
>>	Bitwise right shift

৬. মেম্বারশিপ অপারেটর (Membership Operators)

এই অপারেটরগুলো কোনো ডেটা বা ভ্যালু কোনো সিকোয়েন্সের (যেমন: স্ট্রিং, লিস্ট, টাপল) মধ্যে আছে কি না তা পরীক্ষা করে।

অপারেটর	বর্ণনা	উদাহরণ	ফলাফল
in	যদি ভ্যালু সিকোয়েন্সে থাকে, তাহলে True হবে।	'a' in 'apple'	True
not in	যদি ভ্যালু সিকোয়েন্সে না থাকে, তাহলে True হবে।	'z' not in 'apple'	True

৭. আইডেন্টিটি অপারেটর (Identity Operators)

এই অপারেটরগুলো দুটি ভ্যারিয়েবলের মেমরি লোকেশন একই কি না তা পরীক্ষা করে।

অপারেটর	বর্ণনা	উদাহরণ	ফলাফল
is	যদি দুটি ভ্যারিয়েবলের মেমরি লোকেশন একই হয়, তাহলে True হবে।	x = [1, 2], y = [1, 2], x is y	False (কারণ দুটি ভিন্ন অবজেক্ট)
is not	যদি দুটি ভ্যারিয়েবলের মেমরি লোকেশন একই না হয়, তাহলে True হবে।	x is not y	True

পাইথনে লুপ

ভূমিকা

পাইথনে লুপ হলো এমন একটি প্রক্রিয়া যা নিম্নে আমরা কোনো কোড বুক বা নির্দেশনা একাধিকবার চালাতে পারি। এটি প্রোগ্রামিংয়ে সময় এবং কোড বাঢ়তে সাহায্য করে। পাইথনে মূলত দুই ধরনের লুপ ব্যবহৃত হয়: `for` লুপ এবং `while` লুপ।

for লুপ

for লুপ ব্যবহার করা হয় যখন আমরা জানি কতবার কোনো কোড চালাতে হবে। এটি সাধারণত কোনো সিকেয়েন্স (যেমন: লিস্ট, স্ট্রিং, বা রেঞ্জ) এর উপর কাজ করে।

উদাহরণ ১: ১ থেকে ৫ পর্যন্ত সংখ্যা প্রিন্ট করা

```
for i in range(1, 6):
    print(i)
```

আউটপুট:

```
1  
2  
3  
4  
5
```

ব্যাখ্যা: এখানে `range(1, 6)` ফাংশন ১ থেকে ৫ পর্যন্ত সংখ্যা তৈরি করে। `i` ভেরিয়েবল প্রতিবার একটি সংখ্যা নিয়ে প্রিন্ট করে।

উদাহরণ ২: লিস্টের উপাদান প্রিন্ট করা

```
fruits = ["আপেল", "কলা", "আম"]
for fruit in fruits:
    print(fruit)
```

আউটপুট:

```
আপেল
কলা
আম
```

ব্যাখ্যা: এখানে `fruits` লিস্টের প্রতিটি উপাদান `fruit` ভেরিয়েবলে নিয়ে প্রিন্ট করা হচ্ছে।

for লুপের স্ট্রাকচার

```
for variable in sequence:
    # কোড বুক যা প্রতিবার চালবে
```

while লুপ

while লুপ তখন ব্যবহৃত হয় যখন আমরা জানি না কতবার লুপ চালাবে, কিন্তু একটি শর্ত সত্য থাকা পর্যন্ত লুপ চালাতে চাই।

উদাহরণ ১: ১ থেকে ৫ পর্যন্ত সংখ্যা প্রিন্ট করা

```
i = 1
while i <= 5:
    print(i)
    i += 1
```

আউটপুট:

```
1  
2  
3  
4  
5
```

ব্যাখ্যা: এখানে `i` এর মান ১ থেকে শুরু হয়। যতক্ষণ `i` এর মান ৫ এর সমান বা কম থাকে, ততক্ষণ লুপ চলে এবং `i` এর মান ১ করে বাড়ে।

উদাহরণ ২: ব্যবহারকারীর ইনপুট পর্যন্ত লুপ চালানো

```
password = ""
while password != "secret":
    password = input("পাসওয়ার্ড দিন: ")
    print("সার্কিফ পাসওয়ার্ড!"")
```

আউটপুট:

```
পাসওয়ার্ড দিন: test
পাসওয়ার্ড দিন: secret
সার্কিফ পাসওয়ার্ড!
```

ব্যাখ্যা: এখানে লুপ ততক্ষণ চলে যতক্ষণ না ব্যবহারকারী "secret" পাসওয়ার্ডটি ইনপুট দেয়।

while লুপের স্ট্রাকচার

```
while condition:
    # কোড বুক যা শর্ত সত্য থাকা পর্যন্ত চালবে
    # শর্ত পরিবর্তন করার জন্য ভেরিয়েবল অপেটর করতে হবে
```

break এবং continue

লুপের মধ্যে `break` এবং `continue` ব্যবহার করে লুপের প্রবাহ নিয়ন্ত্রণ করা যায়।

উদাহরণ: break ব্যবহার

```
for i in range(1, 10):
    if i == 5:
        break
    print(i)
```

আউটপুট:

```
1  
2  
3  
4
```

ব্যাখ্যা: `i` এর মান ৫ হলে `break` নূপটি বন্ধ করে দেয়।

উদাহরণ: continue ব্যবহার

```
for i in range(1, 6):
    if i == 3:
        continue
    print(i)
```

আউটপুট:

```
1  
2  
4  
5
```

ব্যাখ্যা: `i` এর মান ৩ হলে `continue` সেই ইটারেশনটি এডিয়ে পরের ইটারেশনে চলে যায়।

উপসংহার

পাইথনে `for` এবং `while` দুটি ব্যবহার করে আমরা বিভিন্ন ধরনের কাজ সহজে করতে পারি। `for` লুপ সিমিট সংখ্যাক ইটারেশনের জন্য ভালো, আর `while` লুপ শর্তিভিত্তি কাজের জন্য উপযোগী। `break` এবং `continue` ব্যবহার করে আমরা লুপের প্রবাহ নিয়ন্ত্রণ করতে পারি।

```
...
-----
Conditional Statement
-----
Largest among Three Numbers:
Write a Python program that takes three numbers as input and prints out the largest among them.
...
try:
    input_one = int(input("Enter The First Number: "))
    input_two = int(input("Enter The Second Number: "))
    input_three = int(input("Enter The Third Number: "))

    if input_one >= input_two and input_one >= input_three:
        largest = input_one
    elif input_two >= input_one and input_two >= input_three:
        largest = input_two
    else:
        largest = input_three

    print(f"The Largest Number is: {largest}")

except ValueError:
    print("Please enter valid integer numbers.")

-----
...
Conditional Statement
-----
Quadrant Identifier:
Write a Python program that takes the coordinates (x, y) of a point as input and prints out which quadrant it belongs to (1st, 2nd, 3rd, or 4th).
...
x = float(input("Enter x coordinate: "))
y = float(input("Enter y coordinate: "))

if x > 0 and y > 0:
    print("The point is in the 1st Quadrant (Q1).")
elif x < 0 and y > 0:
    print("The point is in the 2nd Quadrant (Q2).")
elif x < 0 and y < 0:
    print("The point is in the 3rd Quadrant (Q3).")
elif x > 0 and y < 0:
    print("The point is in the 4th Quadrant (Q4).")
elif x == 0 and y == 0:
    print("The point is at the Origin.")
elif x == 0:
    print("The point lies on the Y-axis.")
elif y == 0:
    print("The point lies on the X-axis.")

-----
...
Conditional Statement
-----
Age Classifier:
Write a Python program that takes a person's age as input and prints out whether they are an infant (0-1), toddler (2-3), child (4-12), teenager (13-19), adult (20+).
...
try:
    age = int(input("Enter the person's age: "))
    if age < 0:
        print("Age cannot be negative.")
    elif age <= 1:
        print("The person is an infant.")
    elif age <= 3:
        print("The person is a toddler.")
    elif age <= 12:
        print("The person is a child.")
    elif age <= 19:
        print("The person is a teenager.")
    else:
        print("The person is an adult.")

except ValueError:
    print("Please enter a valid integer for age.")

#4
...
Conditional Statement
-----
BMI Calculator:
Write a Python program that takes a person's height (in meters) and weight (in kilograms) as input and calculates their Body Mass Index (BMI). Print out their BMI and a message indicating whether they are underweight (<18.5), normal (18.5-24.9), overweight (25-29.9), or obese(>=30).
...
weight = float(input("Enter weight in kg: "))
height = float(input("Enter height in meters: "))

try:
    bmi = weight / (height ** 2)
    if bmi < 18.5:
        print(f"BMI: {bmi:.2f} - Underweight")
    elif 18.5 <= bmi < 25:
        print(f"BMI: {bmi:.2f} - Normal weight")
    elif 25 <= bmi < 30:
        print(f"BMI: {bmi:.2f} - Overweight")
    else:
        print(f"BMI: {bmi:.2f} - Obese")

except ValueError:
    print("Please enter valid numbers for weight and height.")

#5
...
Conditional Statement
-----
Temperature Converter:
Write a Python program that takes a temperature input in Celsius and converts it to Fahrenheit if the temperature is above 0°C, or to Kelvin if the temperature is below 0°C.
...
try:
    celsius = float(input("Enter temperature in Celsius: "))
    if celsius > 0:
        fahrenheit = (celsius * 9/5) + 32
        print(f"Temperature in Fahrenheit: {fahrenheit:.2f}°F")
    elif celsius < 0:
        kelvin = celsius + 273.15
        print(f"Temperature in Kelvin: {kelvin:.2f}K")
    else:
        print("Temperature is 0°C, which is neither above nor below zero.")

except ValueError:
    print("Please enter a valid number for temperature in Celsius.")

#6
...
Conditional Statement
-----
Simple Calculator:
Input two numbers and an operator (+, -, *, /). Use if-elif to perform the operation and print the result. Handle division by zero using conditions.
...
try:
    numbOne = int(input("Enter first number: "))
    numbTwo = int(input("Enter second number: "))
    operator = input("Enter an operator (+, -, *, /): ")
    if operator == '+':
        result = numbOne + numbTwo
        print(f"The result of {numbOne} + {numbTwo} is: {result}")
    elif operator == '-':
        result = numbOne - numbTwo
        print(f"The result of {numbOne} - {numbTwo} is: {result}")
    elif operator == '*':
        result = numbOne * numbTwo
        print(f"The result of {numbOne} * {numbTwo} is: {result}")
    elif operator == '/':
        if numbTwo == 0:
            print("Error: Division by zero is not allowed.")
        else:
            result = numbOne / numbTwo
            print(f"The result of {numbOne} / {numbTwo} is: {result}")
    else:
        print("Invalid operator. Please use +, -, *, or /.")

except ValueError:
    print("Invalid input. Please enter valid numbers.")

#7
...
Loops
-----
FizzBuzz:
Write a Python program that prints the numbers from 1 to 100. But for multiples of three, print "Fizz" instead of the number, and for the multiples of five, print "Buzz". For numbers that are multiples of both three and five, print "FizzBuzz" using a for loop.
...
for i in range(1,101):
    if i % 3 == 0 and i % 5 == 0:
        print("FizzBuzz")
    elif i % 3 == 0:
        print("Fizz")
    elif i % 5 == 0:
        print("Buzz")
    else:
        print(i)

...
#8
...
Loop
-----
Armstrong Number Checker: Write a Python program that takes an integer input from the user and checks if it is an Armstrong number (a number that is equal to the sum of its own digits raised to the power of the number of digits) using a for loop.
...
try:
    number = int(input("Enter an integer: "))
    if number < 0:
        print("Please enter a non-negative integer.")
    else:
        number_str = str(number)
        number_digits = len(number_str)
        total = 0

        for digit in number_str:
            total += int(digit) ** number_digits
            if total == number:
                print(f"{number} is an Armstrong number.")
                break
        else:
            print(f"{number} is not an Armstrong number.")

except ValueError:
    print("Invalid input. Please enter a valid integer.")

#9
...
Loop
-----
Guess the Number Game (while loop):
Consider a random number between 1 and 10. Let user guess until correct. Give hints: "Too high" or "Too low".
...
import random
correct_number = random.randint(1, 10)
while True:
    try:
        guess = int(input('Guess a number between 1 and 10: '))
        if guess < 1 or guess > 10:
            print('Please enter a number between 1 and 10.')
        elif guess > correct_number:
            print('Too high!')
        elif guess < correct_number:
            print('Too low!')
        else:
            print('Congratulations! You guessed the number!')
            break
    except ValueError:
        print('Invalid input. Please enter a number between 1 and 10.')

#10
...
Loop
-----
Password Retry System (while loop):
Predefined password. Let user try until correct or after 3 failed attempts show "Account locked".
...
password = "abcd1234"
attempts = 0

while attempts < 3:
    user_input = input("Enter your password: ")

    if user_input == password:
        print("Access granted")
        break
    else:
        print("Incorrect password")
        attempts += 1

if attempts == 3:
    print("Account locked")

#11
...
Loop
-----
Print the pattern (for/while):
1
22
333
4444
55555
...
for i in range(1, 6):
    print(str(i) * i)

...
#12
...
Loop
-----
Fibonacci Sequence (for loop):
Input N. Print the first N terms of the Fibonacci sequence.
...
n = int(input("Enter the number of terms: "))

if n <= 0:
    print("Please enter a positive integer")
else:
    a, b = 0, 1
    for i in range(n):
        print(a, end=' ')
        temp = a + b
        a = b
        b = temp
    #13
    ...

Loop
-----
Password Retry System (while loop):
Predefined password. Let user try until correct or after 3 failed attempts show "Account locked".
...
password = "abcd1234"
attempts = 0

while attempts < 3:
    user_input = input("Enter your password: ")

    if user_input == password:
        print("Access granted")
        break
    else:
        print("Incorrect password")
        attempts += 1

if attempts == 3:
    print("Account locked")

#14
...
Loop
-----
Guess the Number Game (while loop):
Consider a random number between 1 and 10. Let user guess until correct. Give hints: "Too high" or "Too low".
...
import random
correct_number = random.randint(1, 10)
while True:
    try:
        guess = int(input('Guess a number between 1 and 10: '))
        if guess < 1 or guess > 10:
            print('Please enter a number between 1 and 10.')
        elif guess > correct_number:
            print('Too high!')
        elif guess < correct_number:
            print('Too low!')
        else:
            print('Congratulations! You guessed the number!')
            break
    except ValueError:
        print('Invalid input. Please enter a number between 1 and 10.')
```