**1.HELLO WORLD**

# Storing the string in a variable

my\_string = "Hello, World!"

# Printing the string

print(my\_string)

**2.ARITHEMATIC OPERATORS:**

# Read two integers from input

a = int(input())

b = int(input())

# Print the sum

print(a + b)

# Print the difference (a - b)

print(a - b)

# Print the product

print(a \* b)

**3.PYTHON DIVISION:**

if \_\_name\_\_ == '\_\_main\_\_':

    a = int(input())

    b = int(input())

  # Integer division

    print(a // b)

# Float division

    print(a / b)

**4.LOOPS:**

n = int(input())  # Read input from user

for i in range(n):  # Loop from 0 to n-1

    print(i \*\* 2   # Print square of each number

**5.NESTERED LIST:**

if \_\_name\_\_ == '\_\_main\_\_':

    for \_ in range(int(input())):

        name = input()

        score = float(input())

**6.FINDING PERCENTAGE:**

if \_\_name\_\_ == '\_\_main\_\_':

    n = int(input())

    student\_marks = {}

    for \_ in range(n):

        name, \*line = input().split()

        scores = list(map(float, line))

        student\_marks[name] = scores

    query\_name = input()

**7.LISTS:**

if \_\_name\_\_ == '\_\_main\_\_':

    N = int(input())

    my\_list = []

    for \_ in range(N):

        cmd = input().strip().split()

        command = cmd[0]

        if command == "insert":

            my\_list.insert(int(cmd[1]), int(cmd[2]))

        elif command == "print":

            print(my\_list)

        elif command == "remove":

            my\_list.remove(int(cmd[1]))

        elif command == "append":

            my\_list.append(int(cmd[1]))

        elif command == "sort":

            my\_list.sort()

        elif command == "pop":

            my\_list.pop()

        elif command == "reverse":

            my\_list.reverse()

**8.TUPLES:**

if \_name\_ == '\_main\_':

    n = int(input())

    integer\_list = map(int, input().split())

    t = tuple(integer\_list)

    print(hash(t))

**9.POWER – MOD POWER:**

if \_\_name\_\_ == '\_\_main\_\_':

a = int(input())

b = int(input())

m = int(input())

# First line: a raised to the power of b

print(pow(a, b))

# Second line: a raised to the power of b, modulo m

print(pow(a, b, m))

**10.DEFAULT ARGUMENTS:**

class EvenStream:

    def \_\_init\_\_(self):

        self.current = 0

    def get\_next(self):

        value = self.current

        self.current += 2

        return value

class OddStream:

    def \_\_init\_\_(self):

        self.current = 1

    def get\_next(self):

        value = self.current

        self.current += 2

        return value

# Corrected function

def print\_from\_stream(n, stream=None):

    if stream is None:

        stream = EvenStream()  # Create a new instance each time

    for \_ in range(n):

        print(stream.get\_next())