**DOCTORHARISINGHGOUR**

**VISHWAVIDYALAYA**

**DEPARTMENTOF COMPUTERAPPLICATIONS**



**CSA-CC-3207 PYTHON LAB**

**SESSION:2024-2025**

**SUBMITTED BY**

**SAKSHI PANDEY**

**Y23271024**

**MCA III SEMESTER**

#Program 1 First program in python

print("Hello, World!")



#program 2 for indentation

if 5 > 2:

 print("Five is greater than two!")

if 2 < 5:

        print("Two is lesser than five!")



#Program 3 for get the type of variable

x = 5

y = "John"

print(type(x))

print(type(y))



# Python program 4 showing

# a use of input()

val = input("Enter your value: ")

print(val)



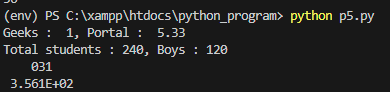
# Python program 5 showing how to use string modulo operator(%)

print("Geeks : %2d, Portal : %5.2f" % (1, 05.333))

print("Total students : %3d, Boys : %2d" % (240, 120))   # print integer value

print("%7.3o" % (25))   # print octal value

print("%10.3E" % (356.08977))   # print exponential value



#Program 6 Working with strings

# Creating a String

# with single Quotes

String1 = 'Welcome to the Geeks World'

print("String with the use of Single Quotes: ")

print(String1)

# Creating a String

# with double Quotes

String1 = "I'm a Geek"

print("\nString with the use of Double Quotes: ")

print(String1)

# Creating String with triple

# Quotes allows multiple lines

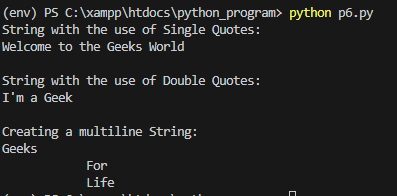
String1 = '''Geeks

            For

            Life'''

print("\nCreating a multiline String: ")

print(String1)



#Program 7 working with Numbers

a = 5.5

b = 3.2

# Addition

c = a + b

print("Addition:", c)

# Subtraction

c = a-b

print("Subtraction:", c)

# Division

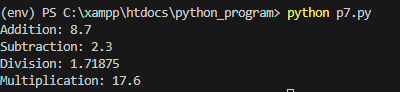
c = a/b

print("Division:", c)

# Multiplication

c = a\*b

print("Multiplication:", c)



#Program 8 Working with Booleans

# Returns False as x is not equal to y

x = 5

y = 10

print(bool(x==y))

# Returns False as x is None

x = None

print(bool(x))

# Returns False as x is an empty sequence

x = ()

print(bool(x))

# Returns False as x is an empty mapping

x = {}

print(bool(x))

# Returns False as x is 0

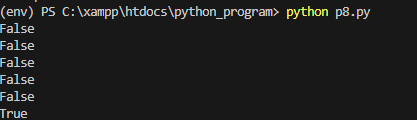
x = 0.0

print(bool(x))

# Returns True as x is a non empty string

x = 'GeeksforGeeks'

print(bool(x))



# Program 9 working with lists

# Initialize an empty list

a = []

# Adding 10 to end of list

a.append(10)

print("After append(10):", a)

# Inserting 5 at index 0

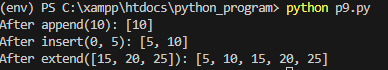
a.insert(0, 5)

print("After insert(0, 5):", a)

# Adding multiple elements  [15, 20, 25] at the end

a.extend([15, 20, 25])

print("After extend([15, 20, 25]):", a)



#Program 10 Working with tuples

# Creating a Tuple

# with Mixed Datatype

Tuple1 = (5, 'Welcome', 7, 'Geeks')

print("\nTuple with Mixed Datatypes: ")

print(Tuple1)

# Creating a Tuple

# with nested tuples

Tuple1 = (0, 1, 2, 3)

Tuple2 = ('python', 'geek')

Tuple3 = (Tuple1, Tuple2)

print("\nTuple with nested tuples: ")

print(Tuple3)

# Creating a Tuple

# with repetition

Tuple1 = ('Geeks',) \* 3

print("\nTuple with repetition: ")

print(Tuple1)

# Creating a Tuple

# with the use of loop

Tuple1 = ('Geeks')

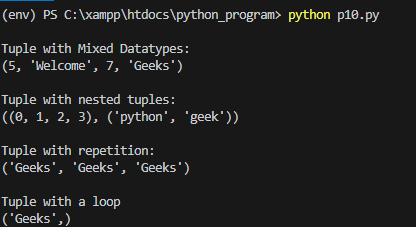
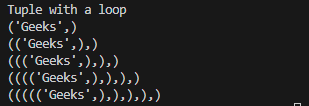
n = 5

print("\nTuple with a loop")

for i in range(int(n)):

    Tuple1 = (Tuple1,)

    print(Tuple1)

#Program 11 Working with sets

# Creating a Set with a List of Numbers

# (Having duplicate values)

set1 = set([1, 2, 4, 4, 3, 3, 3, 6, 5])

print("\nSet with the use of Numbers: ")

print(set1)

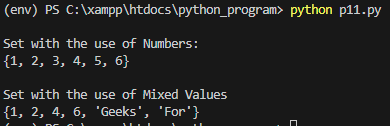
# Creating a Set with a mixed type of values

# (Having numbers and strings)

set1 = set([1, 2, 'Geeks', 4, 'For', 6, 'Geeks'])

print("\nSet with the use of Mixed Values")

print(set1)



#Program 12 Working with Dictionary

Dict = {}

print("Empty Dictionary: ")

print(Dict)

Dict = dict({1: 'Geeks', 2: 'For', 3: 'Geeks'})

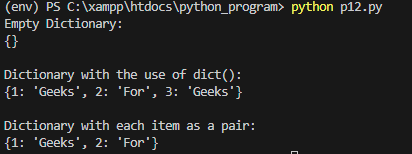
print("\nDictionary with the use of dict(): ")

print(Dict)

Dict = dict([(1, 'Geeks'), (2, 'For')])

print("\nDictionary with each item as a pair: ")

print(Dict)



#Program 13 Working with arrays

import array as arr

a = arr.array('i', [1, 2, 3])

print("Array before insertion : ", end=" ")

for i in range(0, 3):

    print(a[i], end=" ")

print()

a.insert(1, 4)

print("Array after insertion : ", end=" ")

for i in (a):

    print(i, end=" ")

print()

b = arr.array('d', [2.5, 3.2, 3.3])

print("Array before insertion : ", end=" ")

for i in range(0, 3):

    print(b[i], end=" ")

print()

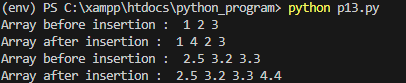
b.append(4.4)

print("Array after insertion : ", end=" ")

for i in (b):

    print(i, end=" ")

print()



#Program 14

# Python program to demonstrate

# type Casting

# int variable

a = 5

# typecast to str

n = str(a)

print(n)

print(type(n))



#Program 15 Use of membership operators

# initialized some sequences

list1 = [1, 2, 3, 4, 5]

str1 = "Hello World"

# using membership 'in' operator

# checking an integer in a list

print(2 in list1)

# checking a character in a string

print('O' in str1)

# Python program to illustrate the use

# of 'is' identity operator

num1 = 5

num2 = 5

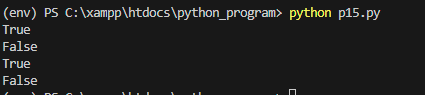
lst1 = [1, 2, 3]

lst2 = [1, 2, 3]

# using 'is' identity operator on different data types

print(num1 is num2)

print(lst1 is lst2)



#program 16

# if..else chain statement

letter = "A"

if letter == "B":

    print("letter is B")

else:

    if letter == "C":

        print("letter is C")

    else:

        if letter == "A":

            print("letter is A")

        else:

            print("letter isn't A, B and C")



#program 17

# simple match case statement

def runMatch():

    num = int(input("Enter a number between 1 and 3: "))

    # match case

    match num:

        # pattern 1

        case 1:

            print("One")

        # pattern 2

        case 2:

            print("Two")

        # pattern 3

        case 3:

            print("Three")

        # default pattern

        case \_:

            print("Number not between 1 and 3")

runMatch()



#Program 18

# Python program to demonstrate

# break statement

s = 'geeksforgeeks'

# Using for loop

for letter in s:

    print(letter)

    # break the loop as soon it sees 'e'

    # or 's'

    if letter == 'e' or letter == 's':

        break

print("Out of for loop")

print()

i = 0

# Using while loop

while True:

    print(s[i])

    # break the loop as soon it sees 'e'

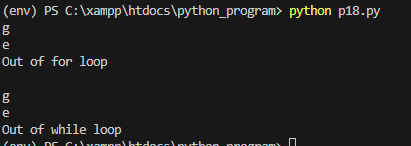
    # or 's'

    if s[i] == 'e' or s[i] == 's':

        break

    i += 1

print("Out of while loop")



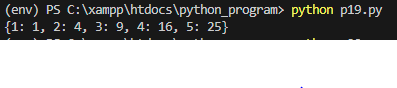
#program 19

# Python code to demonstrate dictionary

# creation using list comprehension

myDict = {x: x\*\*2 for x in [1,2,3,4,5]}

print (myDict)



#Program 20

a = [1, 2, 3, 4, 5]

# Create an empty list 'res' to store results

res = []

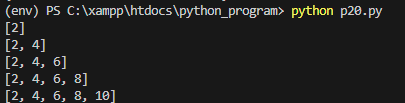
# Iterate over each element in list 'a'

for val in a:

    # Multiply each element by 2 and append it to 'res'

    res.append(val \* 2)

    print(res)



#program 21

# This function has a variable with

# name same as s.

def f():

    s = "Me too."

    print(s)

# Global scope

s = "I love Geeksforgeeks"

f()

print(s)



#program 22

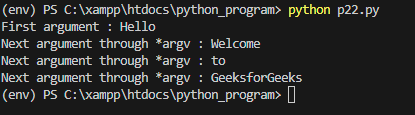
def myFun(arg1, \*argv):

    print("First argument :", arg1)

    for arg in argv:

        print("Next argument through \*argv :", arg)

myFun('Hello', 'Welcome', 'to', 'GeeksforGeeks')



#program 23

class Circle:

    def \_\_init\_\_(self, radius):

        self.radius = radius

    def calculate\_area(self):

        area = 3.14 \* self.radius \*\* 2

        return area

# Creating an instance of Circle

circle\_instance = Circle(5)

# Calling the calculate\_area method

print( circle\_instance.calculate\_area())



#program 24

# A simple Python function to demonstrate

# Polymorphism

def add(x, y, z = 0):

    return x + y+z

# Driver code

print(add(2, 3))

print(add(2, 3, 4))



#program 25

# A Python program to demonstrate inheritance

# Base or Super class. Note object in bracket.

# (Generally, object is made ancestor of all classes)

# In Python 3.x "class Person" is

# equivalent to "class Person(object)"

class Person(object):

    # Constructor

    def \_\_init\_\_(self, name):

        self.name = name

    # To get name

    def getName(self):

        return self.name

    # To check if this person is an employee

    def isEmployee(self):

        return False

# Inherited or Subclass (Note Person in bracket)

class Employee(Person):

    # Here we return true

    def isEmployee(self):

        return True

# Driver code

emp = Person("Geek1")  # An Object of Person

print(emp.getName(), emp.isEmployee())

emp = Employee("Geek2")  # An Object of Employee

print(emp.getName(), emp.isEmployee())

