Cycle-1

1. Program to Print all non-Prime Numbers in an Interval.

Code:

lower\_value = int(input("Please, Enter the Lowest Range Value: "))

upper\_value = int(input("Please, Enter the Upper Range Value: "))

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

print("The Non-Prime Numbers in the range are: ")

for number in range(lower\_value, upper\_value + 1):

if number <= 1:

# Numbers less than or equal to 1 are not prime.

print(number)

else:

is\_prime = True

for i in range(2, int(number\*\*0.5) + 1):

if number % i == 0:

# If the number is divisible by any integer other than 1 and itself, it's not prime.

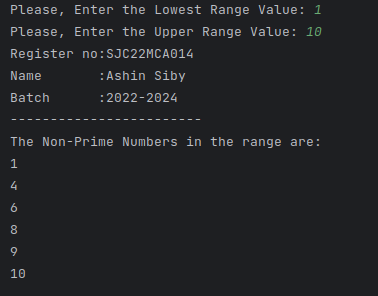
is\_prime = False

break

if not is\_prime:

print(number)

Output



1. Program to print the first N Fibonacci numbers.

Code:

n=int(input("Enter number of terms in series : "))

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

n1=0

n2=1

count=0

if n<0:

print("Please enter a positive number: ")

elif n==1:

print("Fibonacci sequence upto ",n,":")

print(n1)

else:

print("Fibonacci sequence: ")

while count < n:

print(n1)

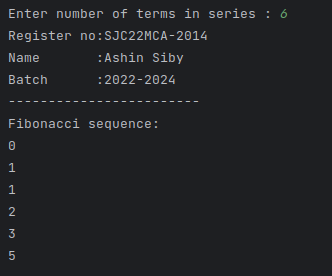
nth=n1+n2

n1=n2

n2=nth

count=count+1

Output



1. Given sides of a triangle, write a program to check whether a given triangle is an isosceles, equilateral or scalene.

Code:

def check\_triangle\_type(a, b, c):

if a == b == c:

return "Equilateral"

elif a == b or b == c or a == c:

return "Isosceles"

else:

return "Scalene"

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

a = float(input("Enter the length of side 'a': "))

b = float(input("Enter the length of side 'b': "))

c = float(input("Enter the length of side 'c': "))

if a <= 0 or b <= 0 or c <= 0:

print("Invalid input. Side lengths must be positive.")

else:

triangle\_type = check\_triangle\_type(a, b, c)

print("The given triangle is:", triangle\_type)

Output

1. Program to check whether given pair of number is coprime

Code:

import math

def are\_coprime(a, b):

gcd = math.gcd(a, b)

return gcd == 1

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

num1 = int(input("Enter the first number: "))

num2 = int(input("Enter the second number: "))

if are\_coprime(num1, num2):

print(f"{num1} and {num2} are coprime.")

else:

print(f"{num1} and {num2} are not coprime.")

Output

1. Program to find the roots of a quadratic equation(rounded to 2 decimal places)

Code:

import math

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

a = float(input("Enter value of a: "))

b = float(input("Enter value of b: "))

c = float(input("Enter value of c: "))

discri = b\*\*2 - 4\*a\*c

if discri > 0:

root1 = (-b + math.sqrt(discri)) / (2\*a)

root2 = (-b - math.sqrt(discri)) / (2\*a)

print(f"Root 1: {round(root1, 2)}")

print(f"Root 2: {round(root2, 2)}")

elif discri == 0:

root = -b / (2\*a)

print(f"Root: {round(root, 2)}")

else:

real\_part = -b / (2\*a)

img\_part = math.sqrt(-discri) / (2\*a)

root1 = complex(real\_part, img\_part)

root2 = complex(real\_part, -img\_part)

print(f"Root 1: {root1.real:.2f} + {root1.imag:.2f}i")

print(f"Root 2: {root2.real:.2f} - {root2.imag:.2f}i")

Output

1. Program to check whether a given number is perfect number or not(sum of factors =number)

Code:

def is\_perfect\_number(num):

if num <= 0:

return False

sum\_of\_divisors = 0

for i in range(1, num):

if num % i == 0:

sum\_of\_divisors += i

return sum\_of\_divisors == num

try:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

num = int(input("Enter a number: "))

if is\_perfect\_number(num):

print(f"{num} is a perfect number.")

else:

print(f"{num} is not a perfect number.")

except ValueError:

print("Invalid input. Please enter a valid number.")

Output

1. Program to display amstrong numbers upto 1000

Code:

def is\_armstrong\_number(num):

num\_str = str(num)

num\_digits = len(num\_str)

armstrong\_sum = sum(int(digit) \*\* num\_digits for digit in num\_str)

return armstrong\_sum == num

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

print("Armstrong numbers up to 1000:")

for num in range(1, 1001):

if is\_armstrong\_number(num):

print(num)

Output

1. Store and display the days of a week as a **List, Tuple, Dictionary, Set.** Also demonstrate different ways to store values in each of them. Display its type also.

Code:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

days\_list = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"]

print("List:", days\_list)

print("Type:", type(days\_list))

days\_tuple = ("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")

print("Tuple:", days\_tuple)

print("Type:", type(days\_tuple))

days\_dict = {0: "Monday", 1: "Tuesday", 2: "Wednesday", 3: "Thursday", 4: "Friday", 5: "Saturday", 6: "Sunday"}

print("Dictionary:", days\_dict)

print("Type:", type(days\_dict))

days\_set = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"}

print("Set:", days\_set)

print("Type:", type(days\_set))

Output

1. Write a program to add elements of given 2 lists

Code:

def add\_lists(list1, list2):

if len(list1) != len(list2):

return None

result = []

for i in range(len(list1)):

result.append(list1[i] + list2[i])

return result

try:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

list1 = input("Enter the first list of numbers separated by spaces: ").split()

list1 = [int(x) for x in list1]

list2 = input("Enter the second list of numbers separated by spaces: ").split()

list2 = [int(x) for x in list2]

result = add\_lists(list1, list2)

if result is None:

print("The lists have different lengths and cannot be added.")

else:

print("Result of addition:", result)

except ValueError:

print("Invalid input. Please enter valid numbers separated by spaces.")

Output

1. Write a program to find the sum of 2 matrices using a nested List.

Code:

for i in range(rows):

row = input(f"Enter elements of row {i + 1} separated by spaces: ").split()

matrix1.append([int(x) for x in row])

print("Enter elements of the second matrix:")

matrix2 = []

for i in range(rows):

row = input(f"Enter elements of row {i + 1} separated by spaces: ").split()

matrix2.append([int(x) for x in row])

result = add\_matrices(matrix1, matrix2)

if result is None:

print("Matrix dimensions are not compatible for addition.")

else:

print("Sum of matrices:")

for row in result:

print(" ".join(map(str, row)))

except ValueError:

print("Invalid input. Please enter valid numbers.")for i in range(rows):

row = input(f"Enter elements of row {i + 1} separated by spaces: ").split()

matrix1.append([int(x) for x in row])

print("Enter elements of the second matrix:")

matrix2 = []

for i in range(rows):

row = input(f"Enter elements of row {i + 1} separated by spaces: ").split()

matrix2.append([int(x) for x in row])

result = add\_matrices(matrix1, matrix2)

if result is None:

print("Matrix dimensions are not compatible for addition.")

else:

print("Sum of matrices:")

for row in result:

print(" ".join(map(str, row)))

except ValueError:

print("Invalid input. Please enter valid numbers.")

Output

1. Write a program to perform bubble sort on a given set of elements.

Code:

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

swapped = False

for j in range(0, n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

swapped = True

if not swapped:

break

try:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

elements = input("Enter elements separated by spaces: ").split()

elements = [int(x) for x in elements]

bubble\_sort(elements)

print("Sorted elements:")

print(elements)

except ValueError:

print("Invalid input. Please enter valid numbers separated by spaces.")

Output

1. Program to find the count of each vowel in a string(use dictionary)

Code:

def count\_vowels(string):

vowel\_counts = {'A': 0, 'E': 0, 'I': 0, 'O': 0, 'U': 0}

string = string.upper()

for char in string:

if char in vowel\_counts:

vowel\_counts[char] += 1

return vowel\_counts

try:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

input\_string = input("Enter a string: ")

vowel\_counts = count\_vowels(input\_string)

print("Vowel counts:")

for vowel, count in vowel\_counts.items():

print(f"{vowel}: {count}")

except ValueError:

print("Invalid input. Please enter a valid string.")

Output

1. Write a Python program that accept a positive number and subtract from this number the sum of its digits and so on. Continues this operation until the number is positive(eg:

256->2+5+6=13

256-13=243

243-9=232……..

Code:

def sum\_of\_digits(n):

digit\_sum = 0

while n > 0:

digit\_sum += n % 10

n //= 10

return digit\_sum

try:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

num = int(input("Enter a positive number: "))

if num <= 0:

print("Please enter a positive number.")

else:

while num > 0:

digit\_sum = sum\_of\_digits(num)

print(f"{num} - {digit\_sum} = {num - digit\_sum}")

num -= digit\_sum

except ValueError:

print("Invalid input. Please enter a valid positive number.")

Output

1. Write a Python program that accepts a 10 digit mobile number, and find the digits which are absent in a given mobile number

Code:

def find\_absent\_digits(mobile\_number):

all\_digits = set("0123456789")

mobile\_digits = set(mobile\_number)

absent\_digits = all\_digits - mobile\_digits

return sorted(list(absent\_digits))

try:

print("Register no:SJC22MCA-2014")

print("Name :Ashin Siby")

print("Batch :2022-2024")

print("------------------------")

mobile\_number = input("Enter a 10-digit mobile number: ")

if len(mobile\_number) == 10 and mobile\_number.isdigit():

absent\_digits = find\_absent\_digits(mobile\_number)

if absent\_digits:

print("Absent digits in the mobile number:", ', '.join(absent\_digits))

else:

print("The mobile number contains all digits from 0 to 9.")

else:

print("Invalid input. Please enter a valid 10-digit mobile number.")

except ValueError:

print("Invalid input. Please enter a valid 10-digit mobile number.")

Output