In [ ]:
```
NAME : Atharva R Karkar
Roll number : 8
Subject : Python {Assignment}
```

1] Write a python code to find given number is prime or not

In [1]:
```
num = int(input("Enter Number: "))
check = False

if num > 1:
    for i in range(2, num):
        if(num % i == 0):
            check = True
            break


    if check:
        print(num, "is not prime number")
    else:
        print(num, "is prime number")
```

```
Enter Number: 5
5 is prime number
```

2] Write a python code to find LCM and GCD of given list

In [4]:
```
# code for LCM
x = int(input("Enter First Number: "))
y = int(input("Enter Second Number: "))
if x > y:
  x, y = y, x
for i in range(1,x+1):
  if x%i == 0 and y%i == 0:
    gcd = i

lcm = (x*y)/gcd

print("LCM of", x, "and", y, "is:", lcm)

# code for GCD
num1 = int(input("Enter First Number: "))
num2 = int(input("Enter Second Number: "))
gcd = 1

for i in range(1, min(num1, num2)):
    if num1 % i == 0 and num2 % i == 0:
        gcd = i
print("GCD of", num1, "and", num2, "is", gcd)
```

```
Enter First Number: 45
Enter Second Number: 9
LCM of 9 and 45 is: 45.0
Enter First Number: 65
Enter Second Number: 13
GCD of 65 and 13 is 1
```

3] Write a python code to find standard deviation of a given list of number

In [5]:
```
lt = [4, 5, 8, 9, 10]
print("The original list : " + str(lt))

mean = sum(lt) / len(lt)
variance = sum([((x - mean) ** 2) for x in lt]) / len(lt)
res = variance ** 0.5

print("Standard deviation of sample is : " + str(res))
```

```
The original list : [4, 5, 8, 9, 10]
Standard deviation of sample is : 2.3151673805580453
```

4] Write a python code to add and delete element from dictionary using function

In [1]:
```
def add_element(dictionary, key, value):
    dictionary[key] = value
    print(f"Element added: {key}: {value}")
def delete_element(dictionary, key):
    if key in dictionary:
        del dictionary[key]
        print(f"Element deleted: {key}")
    else:
        print(f"Key not found: {key}")
my_dict = {'a': 1, 'b': 2, 'c': 3}

add_element(my_dict, 'd', 4)
print("Updated dictionary:", my_dict)

delete_element(my_dict, 'b')
print("Updated dictionary:", my_dict)
```

```
Element added: d: 4
Updated dictionary: {'a': 1, 'b': 2, 'c': 3, 'd': 4}
Element deleted: b
Updated dictionary: {'a': 1, 'c': 3, 'd': 4}
```

5] Write a python code to print 10 student details using class and list

In [2]:

```python
class Student:
    def __init__(self, name, roll_number, grade):
        self.name = name
        self.roll_number = roll_number
        self.grade = grade

students = []

def add_student(name, roll_number, grade):
    student = Student(name, roll_number, grade)
    students.append(student)

def print_student_details():
    for student in students:
        print(f"Name: {student.name}, Roll Number: {student.roll_number}, Grad

add_student("Isha", 101, 'A')
add_student("Swati", 102, 'B')
add_student("Payal", 103, 'C')
add_student("Divyesh", 104, 'A')
add_student("Akshada", 105, 'B')
add_student("Yash", 106, 'C')
add_student("Vishal", 107, 'A')
add_student("Andy", 108, 'B')
add_student("Harshal", 109, 'C')
add_student("Atharva", 110, 'A')


print_student_details()
```

```
Name: Isha, Roll Number: 101, Grade: A
Name: Swati, Roll Number: 102, Grade: B
Name: Payal, Roll Number: 103, Grade: C
Name: Divyesh, Roll Number: 104, Grade: A
Name: Akshada, Roll Number: 105, Grade: B
Name: Yash, Roll Number: 106, Grade: C
Name: Vishal, Roll Number: 107, Grade: A
Name: Andy, Roll Number: 108, Grade: B
Name: Harshal, Roll Number: 109, Grade: C
Name: Atharva, Roll Number: 110, Grade: A
```

6] Write a python code to find student from a given list using class

In [4]:
```
class Student:
    def __init__(self, name, roll_number, grade):
        self.name = name
        self.roll_number = roll_number
        self.grade = grade

students = [
    Student("Isha", 11, 'A'),
    Student("Swati", 12, 'B'),
    Student("Payal", 13, 'C'),
    Student("Renuka", 14, 'A'),
    Student("Akshada", 15, 'B'),
    Student("Yash", 16, 'C'),
    Student("Vishal", 17, 'A'),
    Student("Andy", 18, 'B'),
    Student("Harshal", 19, 'C'),
    Student("Divesh", 20, 'A'),
    Student("Atharva", 21, 'A')
]

def find_student_by_roll_number(roll_number):
    for student in students:
        if student.roll_number == roll_number:
            return student
    return None

roll_to_find = int(input("Enter roll number: "))
found_student = find_student_by_roll_number(roll_to_find)

if found_student:
    print(f"Student found - Name: {found_student.name}, Roll Number: {found_st
else:
    print(f"No student found with Roll Number {roll_to_find}")
```

```
Enter roll number: 21
Student found - Name: Atharva, Roll Number: 21, Grade: A
```

7] Write a python code to inherit employee class to student class

In [6]:
```
class Employee:
    def __init__(self, emp_id, emp_name):
        self.emp_id = emp_id
        self.emp_name = emp_name
    def emp_details(self):
        print("Employee id is: ",self.emp_id)
        print("Employee name is: ",self.emp_name)

class Student(Employee):
    def std_details(self):
        marks = 85
        print("Student marks is: ",marks)

obj = Student(101, "atharva")
obj.emp_details()
obj.std_details()
```

```
Employee id is:  101
Employee name is:  atharva
Student marks is:  85
```

8] Display Fibonacci series up to 10 terms

In [20]:
```
nterms = int(input("How many terms? "))
n1, n2 = 0, 1
count = 0

if nterms <= 0:
    print("Please enter a positive integer")
elif nterms == 1:
    print("Fibonacci sequence upto",nterms,":")
    print(n1)
else:
    print("Fibonacci sequence:")
    while count < nterms:
        print(n1)
        nth = n1 + n2
        n1 = n2
        n2 = nth
        count += 1
```

```
How many terms? 10
Fibonacci sequence:
0
1
1
2
3
5
8
13
21
34
```

9] Find the factorial of a given number

In [25]:
```
num = int(input("Enter number: "))
fact = 1
i = 1
while i <= num:
    fact = i*fact
    i = i + 1
print(fact)
```

```
Enter number: 5
120
```

10] Write a program to iterate a given list and count the occurrence of each element and create a dictionary to show the count of each element.

In [28]:
```
def count_elements(lst):
    element_count = {}

    for element in lst:
        if element in element_count:
            element_count[element] += 1
        else:
            element_count[element] = 1

    return element_count

# Example usage:
my_list = [1, 1, 2, 2, 2, 2, 3, 4, 4, 4, 4]
result = count_elements(my_list)
print(result)
```

```
{1: 2, 2: 4, 3: 1, 4: 4}
```

11] Find the intersection (common) of two sets and remove those elements from the first set

In [29]:
```
first_set = {1, 5, 8, 6, 3, 2}
second_set = {4, 5, 8, 9, 10, 3}

print("First Set ", first_set)
print("Second Set ", second_set)

intersection = first_set.intersection(second_set)
print("Intersection is ", intersection)
for item in intersection:
    first_set.remove(item)

print("First Set after removing common element ", first_set)
```

```
First Set  {1, 2, 3, 5, 6, 8}
Second Set  {3, 4, 5, 8, 9, 10}
Intersection is  {8, 3, 5}
First Set after removing common element  {1, 2, 6}
```

12] Get all values from the dictionary and add them to a list but don’t add duplicates

In [35]:
```
l = [{"name": "Isha", "id" : 1234},
        {"name": "Shraddha", "id": 123},
        {"name": "Isha", "id" : 1234}]
result = list(
    {
        dictionary["name"]: dictionary
        for dictionary in l
    }.values()
)
print(result)
```

```
[{'name': 'Isha', 'id': 1234}, {'name': 'Shraddha', 'id': 1234}]
```

13] Create a Cricle class and intialize it with radius. Make two methods getArea and getCircumference inside this class.

In [37]:
```
class Circle():
    def __init__(self, r):
        self.radius = r

    def getArea(self):
        return self.radius**2*3.14

    def getCircumference(self):
        return 2*self.radius*3.14

NewCircle = Circle(8)
print(NewCircle.getArea())
print(NewCircle.getCircumference())
```

```
200.96
50.24
```

14] Create a Temprature class. Make two methods : 1. convertFahrenheit - It will take celsius and will print it into Fahrenheit. 2. convertCelsius - It will take Fahrenheit and will convert it into Celsius.

In [41]:
```
class Temperature:
    def convert_fahrenheit(self, celsius):
        fahrenheit = (celsius * 9/5) + 32
        print("Temperature in fahrenheit is: ", fahrenheit)

    def convert_celsius(self, fahrenheit):
        celsius = (fahrenheit - 32) * 5/9
        print("Temperature in celsius is: ", celsius)

obj = Temperature()
obj.convert_fahrenheit(25)
obj.convert_celsius(77)
```

```
Temperature in fahrenheit is:  77.0
Temperature in celsius is:  25.0
```

15] Create a Time class and initialize it with hours and minutes. 1. Make a method addTime which should take two time object and add them. E.g.- (2 hour and 50 min)+(1 hr and 20 min) is (4 hr and 10 min) 2. Make a method displayTime which should print the time. 3. Make a method DisplayMinute which should display the total minutes in the Time. E.g.- (1 hr 2 min) should display 62 minute.

In [43]:

```python
class Time:
    def __init__(self, hours, minutes):
        self.hours = hours
        self.minutes = minutes

    def add_time(self, other_time):
        total_hours = self.hours + other_time.hours
        total_minutes = self.minutes + other_time.minutes

        total_hours += total_minutes // 60
        total_minutes %= 60

        return Time(total_hours, total_minutes)

    def display_time(self):
        print(f"{self.hours} hours and {self.minutes} minutes")

    def display_minutes(self):
        total_minutes = self.hours * 60 + self.minutes
        print(f"Total minutes: {total_minutes}")

time1 = Time(2, 50)
time2 = Time(1, 20)

result_time = time1.add_time(time2)
result_time.display_time()

result_time.display_minutes()
```

```
4 hours and 10 minutes
Total minutes: 250
```

16] Write a function “perfect()” that determines if parameter number is a perfect number. Use this function in a program that determines and prints all the perfect numbers between 1 and 1000. [An integer number is said to be “perfect number” if its factors, including 1(but not the number itself), sum to the number. E.g., 6 is a perfect number because 6=1+2+3].

In [48]:
```
def perfect(number):
    factors = [i for i in range(1, number) if number % i == 0]

    return sum(factors) == number

def find_perfect_numbers(start, end):
    perfect_numbers = [num for num in range(start, end + 1) if perfect(num)]
    return perfect_numbers

number_to_check = int(input("Enter number: "))
if perfect(number_to_check):
    print(f"{number_to_check} is a perfect number.")
else:
    print(f"{number_to_check} is not a perfect number.")

result = find_perfect_numbers(1, 1000)
print("Perfect numbers between 1 and 1000:", result)
```

```
Enter number: 28
28 is a perfect number.
Perfect numbers between 1 and 1000: [6, 28, 496]
```

17] Find whether a given string starts with a given character using Lambda

In [2]:
```
starts_with = lambda x: True if x.startswith('P') else False
print(starts_with('Python'))
starts_with = lambda x: True if x.startswith('P') else False
print(starts_with('C++'))
```

```
True
False
```