Write a function add_numbers that takes two numbers as arguments and returns their sum

Write a function multiply_numbers that takes two numbers as arguments and returns their product.

Write a function divide_numbers that takes two numbers as arguments and returns their quotient. Make sure to handle the case where the second number is 0.

In [13]:
```
x=int(input('Enter a x value: '))
y=int(input('Enter a y value: '))
def add(a,b):
    return a+b
sum_of_two_numbers = add(x,y)
print("Sum of {0} and {1} is {2};" .format(x,y, sum_of_two_numbers))

def sub(a,b):
    return a-b
sub_of_two_numbers = sub(x,y)
print("Sub of {0} and {1} is {2};" .format(x,y, sub_of_two_numbers))

def multi(a,b):
    return a*b
multi_of_two_numbers = multi(x,y)
print("Multiple of {0} and {1} is {2};" .format(x,y, multi_of_two_numbers))

if y!=0:
    def div(a,b):
        return a/b
    div_of_two_numbers = div(x,y)
    print("Divide of {0} and {1} is {2};" .format(x,y, div_of_two_numbers))
elif y==0:
    print('Divided y value',y,'is not valid')
```

```
Enter a x value: 10
Enter a y value: 00
Sum of 10 and 0 is 10;
Sub of 10 and 0 is 10;
Multiple of 10 and 0 is 0;
Divided y value 0 is not valid
```

Write a function calculate_average that takes a list of numbers as an argument and returns their average.

In [24]:
```
def Average(lst):
    return sum(lst) / len(lst)

lst = [1, 2, 3, 4, 5, 6, 7, 8]
average = Average(lst)

print("Average of the list =", round(average,2))
```

```
Average of the list = 4.5
```

Write a function is_prime that takes a number as an argument and returns True if the number is prime and False otherwise

In [37]:
```
def is_prime(n):

    if n <= 1:
        return False
    for i in range(2, int(n**0.5)+1):
        if n % i == 0:
            return False
    return True
j=int(input('enter a number: '))
print(is_prime(j))
```

```
enter a number: 23
True
```

OOPS

Create a class Bank with attributes name and accounts. Add methods add_account, remove_account, and get_total_balance that add an account to the list, remove an account from the list, and return the total balance of all accounts, respectively

In [38]:
```
class BankAccount:
    # create the constuctor with parameters: accountNumber, name and balance
    def __init__(self,accountNumber, name, balance):
        self.accountNumber = accountNumber
        self.name = name
        self.balance = balance

    # create Deposit() method
    def Deposit(self , d ):
        self.balance = self.balance + d

    # create Withdrawal method
    def Withdrawal(self , w):
        if(self.balance < w):
            print("impossible operation! Insufficient balance !")
        else:
            self.balance = self.balance - w
    # create bankFees() method
    def bankFees(self):
        self.balance = (95/100)*self.balance

    # create display() method
    def display(self):
        print("Account Number : " , self.accountNumber)
        print("Account Name : " , self.name)
        print("Account Balance : " , self.balance , " $")

# Testing the code :
newAccount = BankAccount(1234567890, "Raam" , 2700)
```

```
# Creating Withdrawal Test
newAccount.Withdrawal(300)
# Create deposit test
newAccount.Deposit(200)
# Display account informations
newAccount.display()
```

```
Account Number :  1234567890
Account Name :  Raam
Account Balance :  2600  $
```