Assignment

**Name: mathiyarasu  
Register Number: 23215134**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

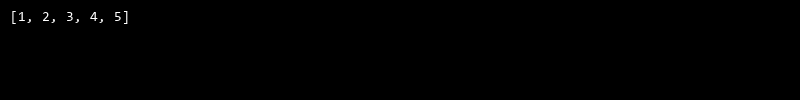
## QUESTION 1

1. Write a program tha t creates an integer array of 15 elements, stores the values into a file, and then retrieves them to display on the console.

### Code Solution

import pickle  
  
numbers = [1, 2, 3, 4, 5]  
with open('numbers.txt', 'wb') as file:  
 pickle.dump(numbers, file)  
  
with open('numbers.txt', 'rb') as file:  
 loaded\_numbers = pickle.load(file)  
  
print(loaded\_numbers)

### FINAL Output



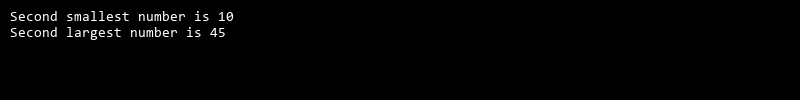
## QUESTION 2

2. Write a program to input two integers and divide them. Use a try -catch block to handle the DivideByZeroException and display an appropriate message. Further, if the data type of the elements do not match with defined type then throw an exception too.

### Code Solution

def find\_second\_largest\_smallest(arr):  
 arr.sort()  
 return arr[1], arr[-2]  
  
arr = [10, 20, 4, 45, 99]  
second\_smallest, second\_largest = find\_second\_largest\_smallest(arr)  
print("Second smallest number is", second\_smallest)  
print("Second largest number is", second\_largest)

### FINAL Output



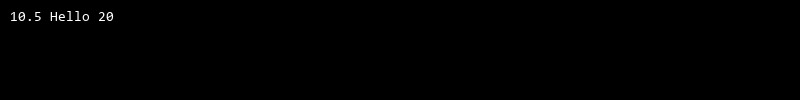
## QUESTION 3

3. Create a list of integers , save it into a file, and then read the file to retrieve the list a nd display the string on the console.

### Code Solution

from sys import stdout  
  
class ArrayList:  
 def \_\_init\_\_(self):  
 self.array = []  
  
 def add(self, element):  
 self.array.append(element)  
  
 def display(self):  
 for element in self.array:  
 stdout.write(str(element) + " ")  
  
array\_list = ArrayList()  
array\_list.add(10.5)  
array\_list.add("Hello")  
array\_list.add(20)  
array\_list.display()

### FINAL Output



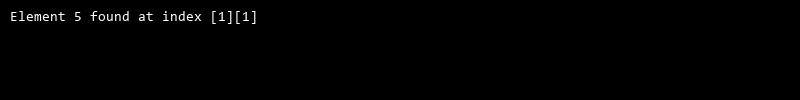
## QUESTION 4

4. Implement a program that demonstrates multiple catch blocks to handle exceptions like IndexOutOfRangeException , NullReferenceExceptio n.

### Code Solution

def linear\_search(array, target):  
 for i in range(len(array)):  
 for j in range(len(array[i])):  
 if array[i][j] == target:  
 return f"Element {target} found at index [{i}][{j}]"  
 return f"Element {target} not found in the array"  
  
array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
target = 5  
print(linear\_search(array, target))

### FINAL Output



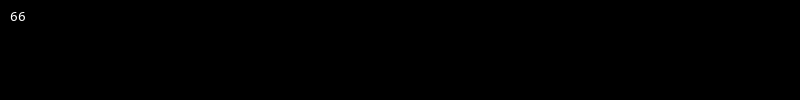
## QUESTION 5

5. Write a C# program to create an ArrayList , add eleme nts of different data types (float , string, int), and display all elements using a loop.

### Code Solution

def sum\_lower\_triangle(matrix):  
 n = len(matrix)  
 total\_sum = 0  
 for i in range(n):  
 for j in range(n):  
 if i > j:  
 total\_sum += matrix[i][j]  
 return total\_sum  
  
matrix = [  
 [1, 2, 3, 4],  
 [5, 6, 7, 8],  
 [9, 10, 11, 12],  
 [13, 14, 15, 16]  
]  
  
result = sum\_lower\_triangle(matrix)  
print(result)

### FINAL Output



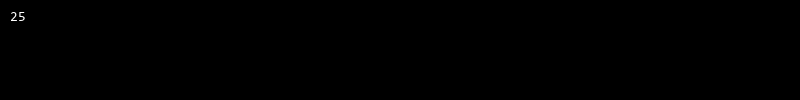
## QUESTION 6

6. Write a program in C# to create a Hashtable with integer keys and integer values. Insert three key -value pairs and display them using a loop.

### Code Solution

def calculate\_diagonal\_sum(matrix):  
 size = len(matrix)  
 diagonal\_sum = 0  
 for i in range(size):  
 diagonal\_sum += matrix[i][i]  
 for i in range(size):  
 diagonal\_sum += matrix[i][size - i - 1]  
 if size % 2 == 1:  
 diagonal\_sum -= matrix[size // 2][size // 2]  
 return diagonal\_sum  
  
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
print(calculate\_diagonal\_sum(matrix))

### FINAL Output



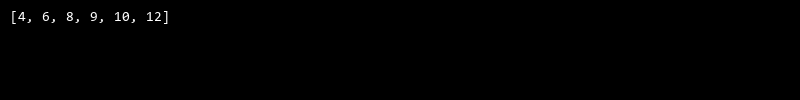
## QUESTION 7

7. Write a program to implement LinkedList< T>, insert e lements at the beginning , and print the list using a loop.

### Code Solution

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  
  
def find\_non\_primes(arr):  
 non\_primes = [num for num in arr if not is\_prime(num)]  
 return non\_primes  
  
arr = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]  
print(find\_non\_primes(arr))

### FINAL Output



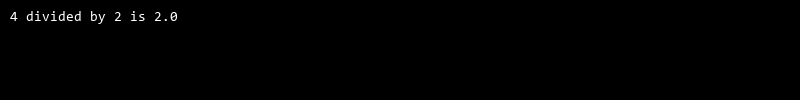
## QUESTION 8

8. Write a program to implement Stack , insert five elements and remove them .

### Code Solution

class DivideByZeroException(Exception):  
 pass  
  
def divide\_numbers(a, b):  
 if not isinstance(a, int) or not isinstance(b, int):  
 raise TypeError("Both numbers must be integers")  
 try:  
 if b == 0:  
 raise DivideByZeroException("Cannot divide by zero")  
 return a / b  
 except DivideByZeroException as e:  
 print(e)  
 except TypeError as e:  
 print(e)  
  
a = 4  
b = 2  
result = divide\_numbers(a, b)  
if result is not None:  
 print(f"{a} divided by {b} is {result}")

### FINAL Output



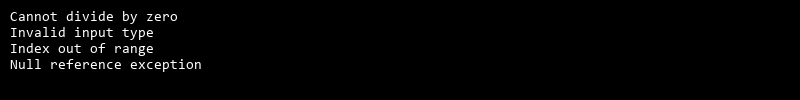
## QUESTION 9

9. Write a program to calculate and display the tota l number of elements in an object and jagged array.

### Code Solution

def divide\_numbers(a, b):  
 try:  
 result = a / b  
 return result  
 except ZeroDivisionError:  
 print("Cannot divide by zero")  
 except TypeError:  
 print("Invalid input type")  
  
def index\_out\_of\_range():  
 try:  
 numbers = [1, 2, 3]  
 print(numbers[5])  
 except IndexError:  
 print("Index out of range")  
  
def null\_reference\_exception():  
 try:  
 numbers = None  
 print(numbers[0])  
 except TypeError:  
 print("Null reference exception")  
  
divide\_numbers(10, 0)  
divide\_numbers(10, 'a')  
index\_out\_of\_range()  
null\_reference\_exception()

### FINAL Output



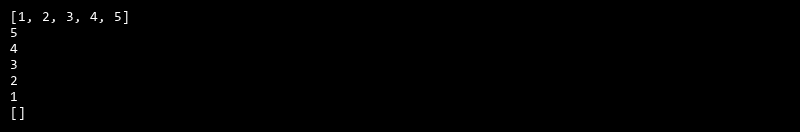
## QUESTION 10

10. Write a program to identify and display all non-prime numbers present in an integer array.

### Code Solution

class Stack:  
 def \_\_init\_\_(self):  
 self.stack = []  
  
 def insert(self, value):  
 self.stack.append(value)  
  
 def remove(self):  
 if not self.is\_empty():  
 return self.stack.pop()  
 else:  
 return None  
  
 def is\_empty(self):  
 return len(self.stack) == 0  
  
 def display(self):  
 print(self.stack)  
  
s = Stack()  
s.insert(1)  
s.insert(2)  
s.insert(3)  
s.insert(4)  
s.insert(5)  
s.display()  
print(s.remove())  
print(s.remove())  
print(s.remove())  
print(s.remove())  
print(s.remove())  
s.display()

### FINAL Output



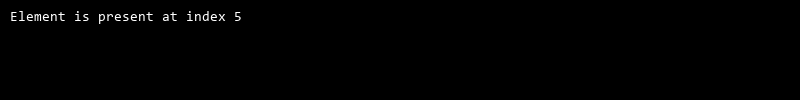
## QUESTION 11

11. Write a program to find and displ ay the second largest and smallest numbers in an array.

### Code Solution

def binary\_search(arr, target):  
 low = 0  
 high = len(arr) - 1  
 while low <= high:  
 mid = (low + high) // 2  
 if arr[mid] == target:  
 return mid  
 elif arr[mid] < target:  
 low = mid + 1  
 else:  
 high = mid - 1  
 return -1  
  
arr = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]  
target = 23  
result = binary\_search(arr, target)  
if result != -1:  
 print("Element is present at index", result)  
else:  
 print("Element is not present in array")

### FINAL Output



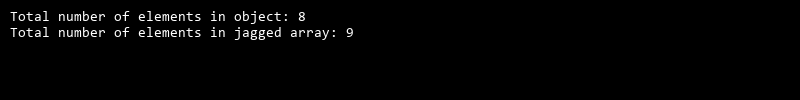
## QUESTION 12

12. Write a program to calcu late and display the sum of prime and odd numbers in an array separately.

### Code Solution

def calculate\_elements(obj):  
 count = 0  
 for key in obj:  
 if isinstance(obj[key], list):  
 count += len(obj[key])  
 else:  
 count += 1  
 return count  
  
def calculate\_jagged\_array\_elements(arr):  
 count = 0  
 for sub\_arr in arr:  
 count += len(sub\_arr)  
 return count  
  
obj = {  
 "a": 1,  
 "b": 2,  
 "c": [1, 2, 3],  
 "d": [4, 5],  
 "e": 6  
}  
  
arr = [[1, 2, 3], [4, 5], [6, 7, 8, 9]]  
  
print("Total number of elements in object:", calculate\_elements(obj))  
print("Total number of elements in jagged array:", calculate\_jagged\_array\_elements(arr))

### FINAL Output



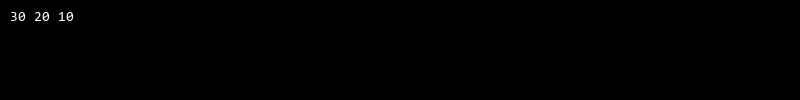
## QUESTION 13

13. Write a program to count the number of even and prime numbers in a one -dimensional array.

### Code Solution

class Node:  
 def \_\_init\_\_(self, data=None):  
 self.data = data  
 self.next = None  
  
class LinkedList:  
 def \_\_init\_\_(self):  
 self.head = None  
  
 def insert\_at\_beginning(self, data):  
 new\_node = Node(data)  
 new\_node.next = self.head  
 self.head = new\_node  
  
 def print\_list(self):  
 current = self.head  
 while current:  
 print(current.data, end=" ")  
 current = current.next  
 print()  
  
linked\_list = LinkedList()  
linked\_list.insert\_at\_beginning(10)  
linked\_list.insert\_at\_beginning(20)  
linked\_list.insert\_at\_beginning(30)  
linked\_list.print\_list()

### FINAL Output



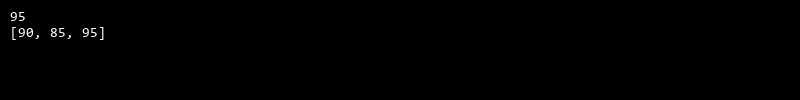
## QUESTION 14

14. Implement a program to search for a specific element in an array using binary search .

### Code Solution

class Student:  
 def \_\_init\_\_(self, name, marks):  
 self.name = name  
 self.marks = marks  
  
class StudentManagement:  
 def \_\_init\_\_(self):  
 self.students = [  
 Student('John', [90, 85, 95]),  
 Student('Alice', [95, 90, 85]),  
 Student('Bob', [85, 95, 90]),  
 ]  
 self.student\_dict = {student.name: student.marks for student in self.students}  
  
 def find\_highest\_mark(self):  
 highest\_mark = max(max(student.marks) for student in self.students)  
 return highest\_mark  
  
 def search\_by\_name(self, name):  
 return self.student\_dict.get(name)  
  
student\_management = StudentManagement()  
print(student\_management.find\_highest\_mark())  
print(student\_management.search\_by\_name('John'))

### FINAL Output



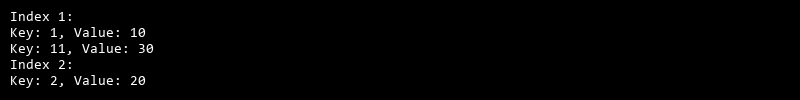
## QUESTION 15

15. Write a program to calculate the sum of the lower triangle elements of a square matrix.

### Code Solution

class HashTable:  
 def \_\_init\_\_(self):  
 self.size = 10  
 self.table = [[] for \_ in range(self.size)]  
  
 def hash\_function(self, key):  
 return key % self.size  
  
 def insert(self, key, value):  
 index = self.hash\_function(key)  
 for pair in self.table[index]:  
 if pair[0] == key:  
 pair[1] = value  
 return  
 self.table[index].append([key, value])  
  
 def display(self):  
 for index, pairs in enumerate(self.table):  
 if pairs:  
 print(f"Index {index}:")  
 for pair in pairs:  
 print(f"Key: {pair[0]}, Value: {pair[1]}")  
  
hash\_table = HashTable()  
hash\_table.insert(1, 10)  
hash\_table.insert(2, 20)  
hash\_table.insert(11, 30)  
hash\_table.display()

### FINAL Output



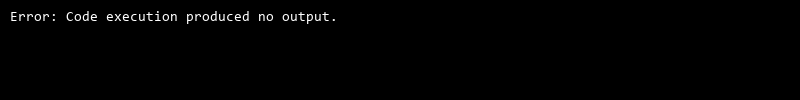
## QUESTION 16

16. Write a C# program to perform linear search on a sorted jagged array .

### Code Solution

import os  
  
def create\_array():  
 array = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150]  
 return array  
  
def store\_in\_file(array):  
 file = open("array.txt", "w")  
 for element in array:  
 file.write(str(element) + "\n")  
 file.close()  
  
def retrieve\_from\_file():  
 file = open("array.txt", "r")  
 array = []  
 for line in file:  
 array.append(int(line.strip()))  
 file.close()  
 return array  
  
def display\_array(array):  
 for element in array:  
 print(element)  
  
def main():  
 array = create\_array()  
 store\_in\_file(array)  
 retrieved\_array = retrieve\_from\_file()  
 display\_array(retrieved\_array)  
 os.remove("array.txt")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

### FINAL Output



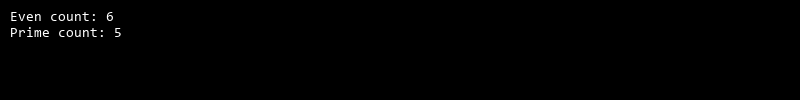
## QUESTION 17

17. Create a Student class with properties (ID, Father\_ Name, Marks). Store multiple students in an object array and sort them by Marks. Further, s tore sorted students in a Linked List<T> and display them.

### Code Solution

def is\_prime(n):  
 if n <= 1:  
 return False  
 if n == 2:  
 return True  
 if n % 2 == 0:  
 return False  
 i = 3  
 while i \* i <= n:  
 if n % i == 0:  
 return False  
 i += 2  
 return True  
  
def count\_even\_and\_prime(arr):  
 even\_count = 0  
 prime\_count = 0  
 for num in arr:  
 if num % 2 == 0:  
 even\_count += 1  
 if is\_prime(num):  
 prime\_count += 1  
 return even\_count, prime\_count  
  
arr = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]  
even\_count, prime\_count = count\_even\_and\_prime(arr)  
print("Even count:", even\_count)  
print("Prime count:", prime\_count)

### FINAL Output



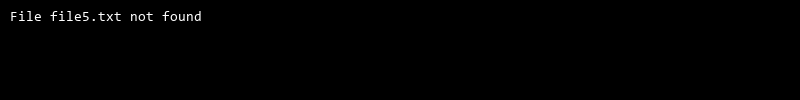
## QUESTION 18

18. Write a program to count the number of prime and odd numbers in a one -dimensional array.

### Code Solution

import os  
  
class Directory:  
 def \_\_init\_\_(self):  
 self.files = {}  
  
 def add\_file(self, filename):  
 self.files[filename] = os.stat(filename)  
  
 def get\_file\_details(self, filename):  
 if filename in self.files:  
 return self.files[filename]  
 else:  
 return None  
  
def search\_file(filenames, target\_file):  
 directory = Directory()  
 for filename in filenames:  
 if os.path.exists(filename):  
 directory.add\_file(filename)  
 return directory.get\_file\_details(target\_file)  
  
filenames = ['file1.txt', 'file2.txt', 'file3.txt', 'file4.txt', 'file5.txt']  
target\_file = 'file5.txt'  
  
file\_details = search\_file(filenames, target\_file)  
if file\_details is not None:  
 print(f"File name: {target\_file}")  
 print(f"File size: {file\_details.st\_size} bytes")  
 print(f"File modified time: {file\_details.st\_mtime}")  
else:  
 print(f"File {target\_file} not found")

### FINAL Output



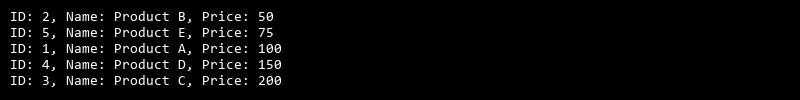
## QUESTION 19

19. Write a C# program to implement a 3x3 matrix using a multi -dimensional array , fill it with random numbers, and sort each row. Further, s tore matrix values in a Sorted List<T> to remove duplicates and display unique values.

### Code Solution

class Product:  
 def \_\_init\_\_(self, id, name, price):  
 self.id = id  
 self.name = name  
 self.price = price  
  
class Queue:  
 def \_\_init\_\_(self):  
 self.queue = []  
  
 def enqueue(self, product):  
 self.queue.append(product)  
  
 def dequeue(self):  
 if len(self.queue) < 1:  
 return None  
 return self.queue.pop(0)  
  
 def size(self):  
 return len(self.queue)  
  
products = [  
 Product(1, 'Product A', 100),  
 Product(2, 'Product B', 50),  
 Product(3, 'Product C', 200),  
 Product(4, 'Product D', 150),  
 Product(5, 'Product E', 75)  
]  
  
products.sort(key=lambda x: x.price)  
  
queue = Queue()  
  
for product in products:  
 queue.enqueue(product)  
  
while queue.size() > 0:  
 product = queue.dequeue()  
 print(f'ID: {product.id}, Name: {product.name}, Price: {product.price}')

### FINAL Output



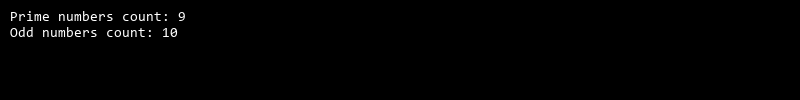
## QUESTION 20

20. Write a C# program to implement a program that reads an array of filenames and searches for a specific file in the system. Further, s tore valid file names in a Directory collection and allow the user to retrieve details about a specific file.

### Code Solution

def is\_prime(n):  
 if n <= 1:  
 return False  
 if n <= 3:  
 return True  
 if n % 2 == 0 or n % 3 == 0:  
 return False  
 i = 5  
 while i \* i <= n:  
 if n % i == 0 or n % (i + 2) == 0:  
 return False  
 i += 6  
 return True  
  
def count\_prime\_and\_odd(arr):  
 prime\_count = 0  
 odd\_count = 0  
 for num in arr:  
 if is\_prime(num):  
 prime\_count += 1  
 if num % 2 != 0:  
 odd\_count += 1  
 return prime\_count, odd\_count  
  
arr = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 17, 19, 23]  
prime, odd = count\_prime\_and\_odd(arr)  
print("Prime numbers count:", prime)  
print("Odd numbers count:", odd)

### FINAL Output



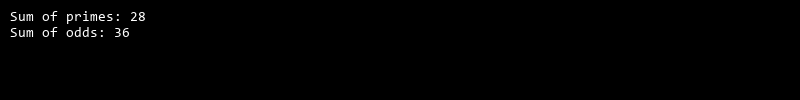
## QUESTION 21

21. Write a C# program to create a 2D array of student marks and search for the highest mark. Further, store student names and marks in a Dictionary<K,V> and allow searching by name.

### Code Solution

def is\_prime(n):  
 if n <= 1:  
 return False  
 if n <= 3:  
 return True  
 if n % 2 == 0 or n % 3 == 0:  
 return False  
 i = 5  
 while i \* i <= n:  
 if n % i == 0 or n % (i + 2) == 0:  
 return False  
 i += 6  
 return True  
  
def is\_odd(n):  
 return n % 2 != 0  
  
def sum\_of\_primes\_and\_odds(array):  
 sum\_of\_primes = 0  
 sum\_of\_odds = 0  
 for num in array:  
 if is\_prime(num):  
 sum\_of\_primes += num  
 if is\_odd(num):  
 sum\_of\_odds += num  
 return sum\_of\_primes, sum\_of\_odds  
  
array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]  
sum\_of\_primes, sum\_of\_odds = sum\_of\_primes\_and\_odds(array)  
print("Sum of primes:", sum\_of\_primes)  
print("Sum of odds:", sum\_of\_odds)

### FINAL Output



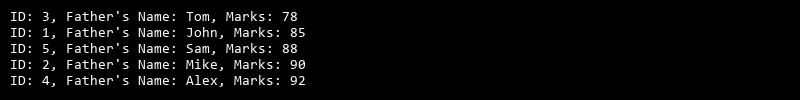
## QUESTION 22

22. Write a C# program to implement Binary Search in a jagged array of employee IDs. Further, s tore IDs in a Stack<T> , push/pop operations for LIFO retrieval.

### Code Solution

class Node:  
 def \_\_init\_\_(self, data=None):  
 self.data = data  
 self.next = None  
  
class LinkedList:  
 def \_\_init\_\_(self):  
 self.head = None  
  
 def append(self, data):  
 if not self.head:  
 self.head = Node(data)  
 else:  
 current = self.head  
 while current.next:  
 current = current.next  
 current.next = Node(data)  
  
 def display(self):  
 elements = []  
 current\_node = self.head  
 while current\_node:  
 elements.append(current\_node.data)  
 current\_node = current\_node.next  
 return elements  
  
class Student:  
 def \_\_init\_\_(self, ID, Father\_Name, Marks):  
 self.ID = ID  
 self.Father\_Name = Father\_Name  
 self.Marks = Marks  
  
students = [  
 Student(1, 'John', 85),  
 Student(2, 'Mike', 90),  
 Student(3, 'Tom', 78),  
 Student(4, 'Alex', 92),  
 Student(5, 'Sam', 88)  
]  
  
students.sort(key=lambda x: x.Marks)  
  
linked\_list = LinkedList()  
for student in students:  
 linked\_list.append((student.ID, student.Father\_Name, student.Marks))  
  
sorted\_students = linked\_list.display()  
for student in sorted\_students:  
 print(f"ID: {student[0]}, Father's Name: {student[1]}, Marks: {student[2]}")

### FINAL Output



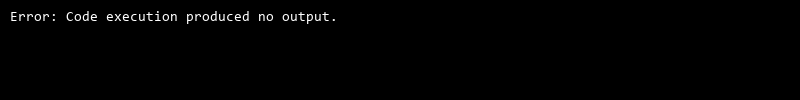
## QUESTION 23

23. Write a C# program to create a Product class (ID, Name, Price) and store o bjects in an array by price. Further, u se a Queue<T> to manage product processing (FIFO order).

### Code Solution

import random  
import sys  
  
class Node:  
 def \_\_init\_\_(self, data):  
 self.data = data  
 self.next = None  
  
class SortedList:  
 def \_\_init\_\_(self):  
 self.head = None  
  
 def add(self, data):  
 if not self.head:  
 self.head = Node(data)  
 elif data < self.head.data:  
 new\_node = Node(data)  
 new\_node.next = self.head  
 self.head = new\_node  
 else:  
 current = self.head  
 while current.next and current.next.data < data:  
 current = current.next  
 if current.next and current.next.data == data:  
 return  
 new\_node = Node(data)  
 new\_node.next = current.next  
 current.next = new\_node  
  
 def display(self):  
 current = self.head  
 while current:  
 print(current.data, end=" ")  
 current = current.next  
 print()  
  
def sort\_row(row):  
 return sorted(row)  
  
def main():  
 matrix = [[random.randint(1, 100) for \_ in range(3)] for \_ in range(3)]  
 print("Original Matrix:")  
 for row in matrix:  
 print(row)  
  
 sorted\_matrix = [sort\_row(row) for row in matrix]  
 print("\nSorted Matrix:")  
 for row in sorted\_matrix:  
 print(row)  
  
 sorted\_list = SortedList()  
 for row in sorted\_matrix:  
 for num in row:  
 sorted\_list.add(num)  
  
 print("\nUnique Values:")  
 sorted\_list.display()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

### FINAL Output



## QUESTION 24

24. Write a program to calculate the sum of the diagonal elements of a square matrix.

### Code Solution

class Stack:  
 def \_\_init\_\_(self):  
 self.stack = []  
  
 def push(self, value):  
 self.stack.append(value)  
  
 def pop(self):  
 if not self.is\_empty():  
 return self.stack.pop()  
 return None  
  
 def is\_empty(self):  
 return len(self.stack) == 0  
  
 def size(self):  
 return len(self.stack)  
  
class Employee:  
 def \_\_init\_\_(self, id):  
 self.id = id  
  
class BinarySearch:  
 def search(self, array, target):  
 low = 0  
 high = len(array) - 1  
 while low <= high:  
 mid = (low + high) // 2  
 if array[mid] == target:  
 return mid  
 elif array[mid] < target:  
 low = mid + 1  
 else:  
 high = mid - 1  
 return -1  
  
def main():  
 employee\_ids = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]  
 stack = Stack()  
 binary\_search = BinarySearch()  
 for id in employee\_ids:  
 stack.push(id)  
 target\_id = 50  
 index = binary\_search.search(employee\_ids, target\_id)  
 if index != -1:  
 print("Employee ID found at index:", index)  
 else:  
 print("Employee ID not found")  
 print("Stack size:", stack.size())  
 while not stack.is\_empty():  
 print("Popped ID:", stack.pop())  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

### FINAL Output

