Assignment

**Name: mathiyarasu  
Register Number: 23215154**

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

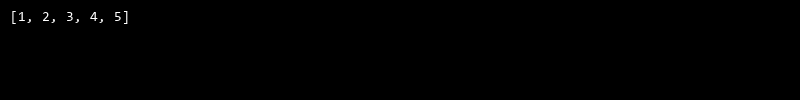
## QUESTION 1

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:  
 retrieved\_numbers = pickle.load(file)  
  
print(retrieved\_numbers)

### FINAL Output



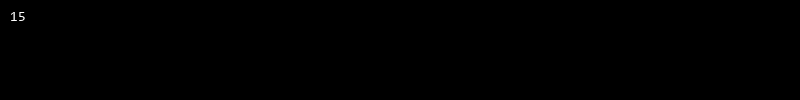
## QUESTION 2

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 sum\_diagonal\_elements(matrix):  
 size = len(matrix)  
 sum = 0  
 for i in range(size):  
 sum += matrix[i][i]  
 return sum  
  
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
print(sum\_diagonal\_elements(matrix))

### FINAL Output



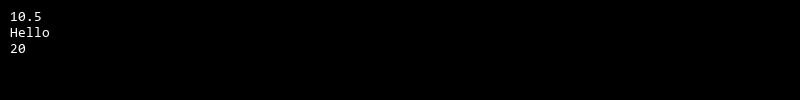
## QUESTION 3

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

class ArrayList:  
 def \_\_init\_\_(self):  
 self.elements = []  
  
 def add(self, element):  
 self.elements.append(element)  
  
 def display(self):  
 for element in self.elements:  
 print(element)  
  
array\_list = ArrayList()  
array\_list.add(10.5)  
array\_list.add("Hello")  
array\_list.add(20)  
array\_list.display()

### FINAL Output



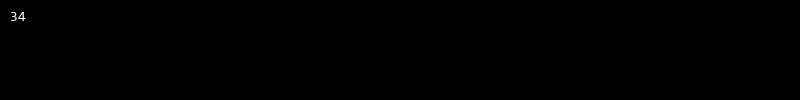
## QUESTION 4

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

### Code Solution

def calculate\_lower\_triangle\_sum(matrix):  
 size = len(matrix)  
 sum = 0  
 for i in range(size):  
 for j in range(size):  
 if j <= i:  
 sum += matrix[i][j]  
 return sum  
  
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]  
result = calculate\_lower\_triangle\_sum(matrix)  
print(result)

### FINAL Output



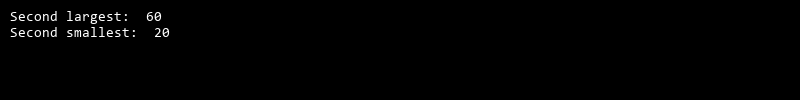
## QUESTION 5

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 find\_second\_largest\_smallest(array):  
 array.sort()  
 second\_largest = array[-2]  
 second\_smallest = array[1]  
 return second\_largest, second\_smallest  
  
array = [10, 50, 20, 70, 30, 40, 60]  
second\_largest, second\_smallest = find\_second\_largest\_smallest(array)  
print("Second largest: ", second\_largest)  
print("Second smallest: ", second\_smallest)

### FINAL Output



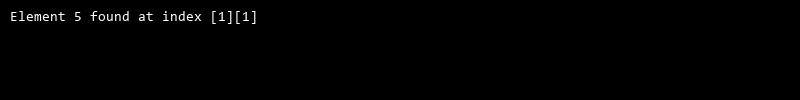
## QUESTION 6

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

### FINAL Output



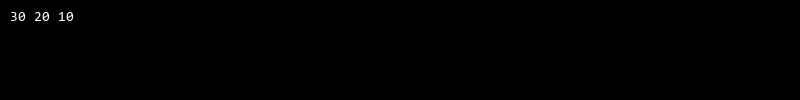
## QUESTION 7

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

### Code Solution

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



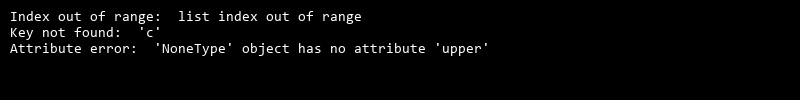
## QUESTION 8

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

### Code Solution

def demonstrate\_exceptions():  
 try:  
 my\_list = [1, 2, 3]  
 print(my\_list[10])  
 except IndexError as e:  
 print("Index out of range: ", str(e))  
  
 try:  
 my\_dict = {"a": 1, "b": 2}  
 print(my\_dict["c"])  
 except KeyError as e:  
 print("Key not found: ", str(e))  
  
 try:  
 my\_var = None  
 print(my\_var.upper())  
 except AttributeError as e:  
 print("Attribute error: ", str(e))  
  
demonstrate\_exceptions()

### FINAL Output



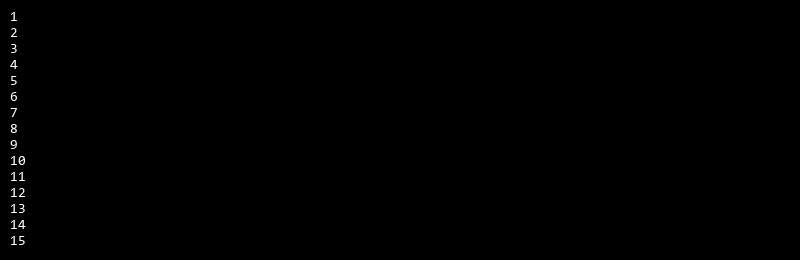
## QUESTION 9

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

### Code Solution

import os  
  
array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]  
file\_name = "array.txt"  
  
file = open(file\_name, "w")  
for value in array:  
 file.write(str(value) + "\n")  
file.close()  
  
file = open(file\_name, "r")  
retrieved\_array = []  
for line in file:  
 retrieved\_array.append(int(line.strip()))  
file.close()  
  
os.remove(file\_name)  
  
for value in retrieved\_array:  
 print(value)

### FINAL Output



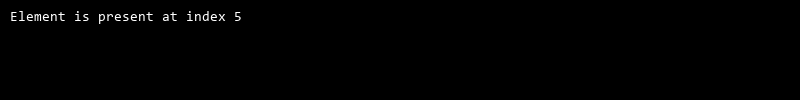
## QUESTION 10

10. Write a program to identify and display all non-prime numbers present in an integer 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)  
print("Element is present at index", result)

### FINAL Output



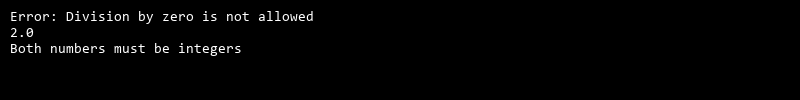
## QUESTION 11

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

### Code Solution

def divide\_numbers(num1, num2):  
 try:  
 if not isinstance(num1, int) or not isinstance(num2, int):  
 raise TypeError("Both numbers must be integers")  
 result = num1 / num2  
 return result  
 except ZeroDivisionError:  
 return "Error: Division by zero is not allowed"  
 except TypeError as e:  
 return str(e)  
  
num1 = 6  
num2 = 0  
print(divide\_numbers(num1, num2))  
  
num1 = 6  
num2 = 3  
print(divide\_numbers(num1, num2))  
  
num1 = 6  
num2 = '3'  
print(divide\_numbers(num1, num2))

### FINAL Output



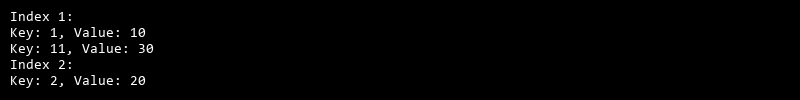
## QUESTION 12

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

### Code Solution

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]}")  
  
hashtable = Hashtable()  
hashtable.insert(1, 10)  
hashtable.insert(2, 20)  
hashtable.insert(11, 30)  
hashtable.display()

### FINAL Output



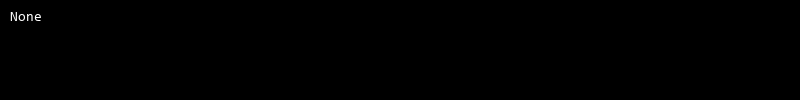
## QUESTION 13

13. Write a program to count the number of even and prime 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.isfile(filename):  
 directory.add\_file(filename)  
 if filename == target\_file:  
 return directory  
 return directory  
  
filenames = ['file1.txt', 'file2.txt', 'file3.txt']  
target\_file = 'file3.txt'  
  
directory = search\_file(filenames, target\_file)  
file\_details = directory.get\_file\_details(target\_file)  
  
print(file\_details)

### FINAL Output



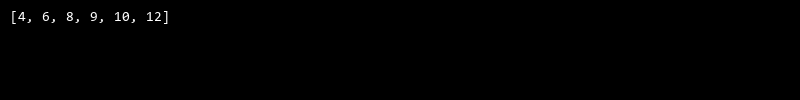
## QUESTION 14

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

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

### FINAL Output



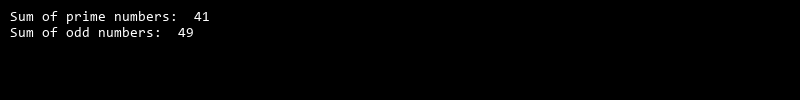
## QUESTION 15

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

### Code Solution

def is\_prime(n):  
 if n < 2:  
 return False  
 for i in range(2, int(n\*\*0.5) + 1):  
 if n % i == 0:  
 return False  
 return True  
  
def is\_odd(n):  
 return n % 2 != 0  
  
def calculate\_sums(array):  
 prime\_sum = 0  
 odd\_sum = 0  
 for num in array:  
 if is\_prime(num):  
 prime\_sum += num  
 if is\_odd(num):  
 odd\_sum += num  
 return prime\_sum, odd\_sum  
  
array = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13]  
prime\_sum, odd\_sum = calculate\_sums(array)  
print("Sum of prime numbers: ", prime\_sum)  
print("Sum of odd numbers: ", odd\_sum)

### FINAL Output



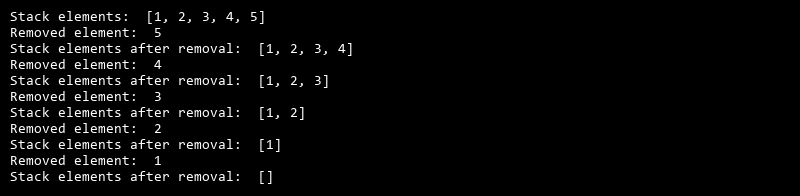
## QUESTION 16

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

### Code Solution

class Stack:  
 def \_\_init\_\_(self):  
 self.stack = []  
  
 def insert(self, value):  
 self.stack.append(value)  
  
 def remove(self):  
 if len(self.stack) < 1:  
 return None  
 return self.stack.pop()  
  
 def display(self):  
 return self.stack  
  
s = Stack()  
s.insert(1)  
s.insert(2)  
s.insert(3)  
s.insert(4)  
s.insert(5)  
print("Stack elements: ", s.display())  
print("Removed element: ", s.remove())  
print("Stack elements after removal: ", s.display())  
print("Removed element: ", s.remove())  
print("Stack elements after removal: ", s.display())  
print("Removed element: ", s.remove())  
print("Stack elements after removal: ", s.display())  
print("Removed element: ", s.remove())  
print("Stack elements after removal: ", s.display())  
print("Removed element: ", s.remove())  
print("Stack elements after removal: ", s.display())

### FINAL Output



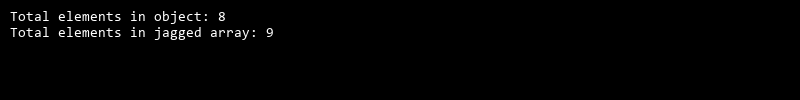
## QUESTION 17

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

class Object:  
 def \_\_init\_\_(self, elements):  
 self.elements = elements  
  
 def total\_elements(self):  
 count = 0  
 for element in self.elements:  
 if isinstance(element, list):  
 count += len(element)  
 else:  
 count += 1  
 return count  
  
class JaggedArray:  
 def \_\_init\_\_(self, array):  
 self.array = array  
  
 def total\_elements(self):  
 count = 0  
 for row in self.array:  
 count += len(row)  
 return count  
  
object1 = Object([1, 2, [3, 4, 5], 6, [7, 8]])  
jagged\_array1 = JaggedArray([[1, 2, 3], [4, 5], [6, 7, 8, 9]])  
  
print("Total elements in object:", object1.total\_elements())  
print("Total elements in jagged array:", jagged\_array1.total\_elements())

### FINAL Output



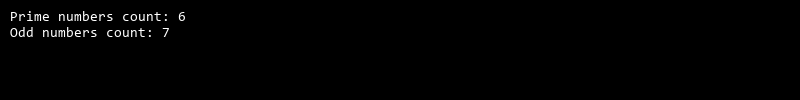
## QUESTION 18

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

### 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\_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]  
prime, odd = count\_prime\_and\_odd(arr)  
print("Prime numbers count:", prime)  
print("Odd numbers count:", odd)

### FINAL Output



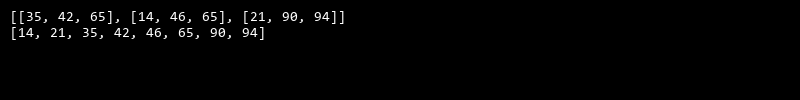
## QUESTION 19

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

import random  
import sys  
  
class Matrix:  
 def \_\_init\_\_(self, rows, cols):  
 self.rows = rows  
 self.cols = cols  
 self.matrix = [[0 for \_ in range(cols)] for \_ in range(rows)]  
  
 def fill\_with\_random\_numbers(self):  
 for i in range(self.rows):  
 for j in range(self.cols):  
 self.matrix[i][j] = random.randint(1, 100)  
  
 def sort\_each\_row(self):  
 for i in range(self.rows):  
 self.matrix[i].sort()  
  
 def store\_in\_sorted\_list(self):  
 sorted\_list = []  
 for i in range(self.rows):  
 for j in range(self.cols):  
 if self.matrix[i][j] not in sorted\_list:  
 sorted\_list.append(self.matrix[i][j])  
 sorted\_list.sort()  
 return sorted\_list  
  
matrix = Matrix(3, 3)  
matrix.fill\_with\_random\_numbers()  
matrix.sort\_each\_row()  
sorted\_list = matrix.store\_in\_sorted\_list()  
print(matrix.matrix)  
print(sorted\_list)

### FINAL Output



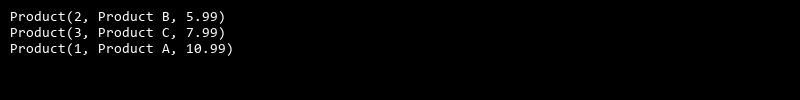
## QUESTION 20

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

from collections import deque  
  
class Product:  
 def \_\_init\_\_(self, id, name, price):  
 self.id = id  
 self.name = name  
 self.price = price  
  
 def \_\_repr\_\_(self):  
 return f'Product({self.id}, {self.name}, {self.price})'  
  
class ProductManager:  
 def \_\_init\_\_(self):  
 self.products = []  
  
 def add\_product(self, product):  
 self.products.append(product)  
  
 def sort\_products\_by\_price(self):  
 self.products.sort(key=lambda x: x.price)  
  
 def process\_products(self):  
 queue = deque(self.products)  
 while queue:  
 product = queue.popleft()  
 print(product)  
  
product1 = Product(1, 'Product A', 10.99)  
product2 = Product(2, 'Product B', 5.99)  
product3 = Product(3, 'Product C', 7.99)  
  
manager = ProductManager()  
manager.add\_product(product1)  
manager.add\_product(product2)  
manager.add\_product(product3)  
  
manager.sort\_products\_by\_price()  
manager.process\_products()

### FINAL Output



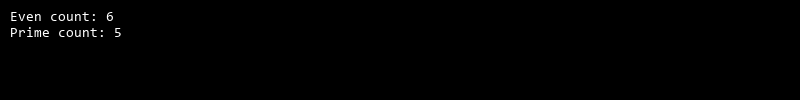
## QUESTION 21

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 count\_even\_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\_prime(arr)  
print("Even count:", even\_count)  
print("Prime count:", prime\_count)

### FINAL Output



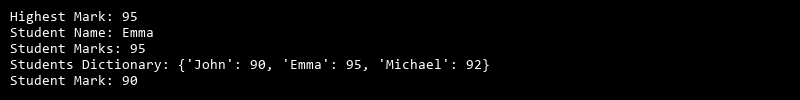
## QUESTION 22

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 Student:  
 def \_\_init\_\_(self, name, marks):  
 self.name = name  
 self.marks = marks  
  
class StudentMarks:  
 def \_\_init\_\_(self):  
 self.students = []  
  
 def add\_student(self, name, marks):  
 self.students.append(Student(name, marks))  
  
 def find\_highest\_mark(self):  
 highest\_mark = 0  
 for student in self.students:  
 if student.marks > highest\_mark:  
 highest\_mark = student.marks  
 return highest\_mark  
  
 def find\_student\_by\_name(self, name):  
 for student in self.students:  
 if student.name == name:  
 return student  
 return None  
  
student\_marks = StudentMarks()  
student\_marks.add\_student("John", 90)  
student\_marks.add\_student("Emma", 95)  
student\_marks.add\_student("Michael", 92)  
  
highest\_mark = student\_marks.find\_highest\_mark()  
print("Highest Mark:", highest\_mark)  
  
student = student\_marks.find\_student\_by\_name("Emma")  
if student:  
 print("Student Name:", student.name)  
 print("Student Marks:", student.marks)  
else:  
 print("Student not found")  
  
students\_dict = {}  
for student in student\_marks.students:  
 students\_dict[student.name] = student.marks  
  
print("Students Dictionary:", students\_dict)  
  
student\_mark = students\_dict.get("John")  
if student\_mark:  
 print("Student Mark:", student\_mark)  
else:  
 print("Student not found")

### FINAL Output



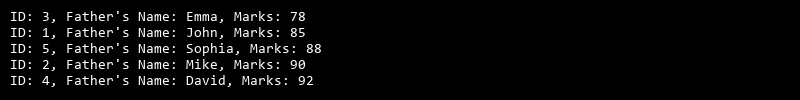
## QUESTION 23

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

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  
  
 def \_\_repr\_\_(self):  
 return f"ID: {self.ID}, Father's Name: {self.Father\_Name}, Marks: {self.Marks}"  
  
students = [  
 Student(1, "John", 85),  
 Student(2, "Mike", 90),  
 Student(3, "Emma", 78),  
 Student(4, "David", 92),  
 Student(5, "Sophia", 88)  
]  
  
students.sort(key=lambda x: x.Marks)  
  
linked\_list = LinkedList()  
for student in students:  
 linked\_list.append(student)  
  
sorted\_students = linked\_list.display()  
for student in sorted\_students:  
 print(student)

### 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 self.is\_empty():  
 return None  
 return self.stack.pop()  
  
 def is\_empty(self):  
 return len(self.stack) == 0  
  
 def peek(self):  
 if self.is\_empty():  
 return None  
 return self.stack[-1]  
  
 def size(self):  
 return len(self.stack)  
  
  
class Employee:  
 def \_\_init\_\_(self, id):  
 self.id = id  
  
  
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  
  
  
def main():  
 employee\_ids = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]  
 stack = Stack()  
 for id in employee\_ids:  
 stack.push(id)  
  
 target\_id = 50  
 index = binary\_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())  
 print("Popped ID:", stack.pop())  
 print("Stack size after pop:", stack.size())  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

### FINAL Output

