```
'''
//==============================================================================
====================
// Name         : 21465_Pract1.py
// Author       : Chaitanya Paraskar
// Roll No.     : 21465
// Aim          : In second year computer engineering class, group A student’s
play cricket,
                 group B students play badminton and group C students play
football.
                 Write a Python program using functions to compute following: -
                    a) List of students who play both cricket and badminton
                    b) List of students who play either cricket or badminton but
not both\
                    c) Number of students who play neither cricket nor badminton

                    d) Number of students who play cricket and football but not
badminton.
                    (Note- While realizing the group, duplicate entries should
be avoided,
                    Do not use SET built-in functions)
//==============================================================================
====================

'''


def present(v, li):
    for i in range(len(li)):
        if v == li[i]:
            return True
    else:
        return False


def not_present(v, li):
    for i in range(len(li)):
        if v == li[i]:
            return False
    else:
        return True


def intersection(l1, l2):
    res = []
    for e in l1:
        if present(e, l2) and not_present(e, res):
            res.append(e)
    return res


def union(l1, l2):
    res = []
    for e in l1:
        res.append(e)

    for e in l2:
        if not_present(e, res):
            res.append(e)
    return res


def difference(l1, l2):
    res = []
    for e in l1:
        if not_present(e, l2):
            res.append(e)
    return res


len_A = int(input("Enter No. of students who play Cricket :-"))
A = []
for i in range(len_A):
    name = input(f"Enter name of student {i+1}: ")
    if present(name, A):
        print(name, "already present !!")
    else:
        A.append(name)

len_B = int(input("Enter No. of students who play Badminton :-"))
B = []
for i in range(len_B):
    name = input(f"Enter name of student {i+1}: ")
    if present(name, B):
        print(name, "already present !!")
    else:
        B.append(name)

len_C = int(input("Enter No. of students who play Football :-"))
C = []
for i in range(len_C):
    name = input(f"Enter name of student {i+1}: ")
    if present(name, C):
        print(name, "already present !!")
    else:
        C.append(name)

print("Students who play both Cricket and Badminton : ",
      intersection(A, B))

print("Students who play either cricket or badminton but not both : ",
      difference(union(A, B), intersection(A, B)))

print("Number of students who play neither cricket nor badminton : ",
      len(difference(union(union(A, B), C), union(A, B))))

print("Number of students who play cricket and football but not badminton : ",
      len(difference(intersection(A, C), B)))
```

```
'''

OUTPUT :-

$ python pract1.py
Enter No. of students who play Cricket :-4
Enter name of student 1: 1
Enter name of student 2: 2
Enter name of student 3: 3
Enter name of student 4: 7
Enter No. of students who play Badminton :-4
Enter name of student 1: 1
Enter name of student 2: 5
Enter name of student 3: 7
Enter name of student 4: 6
Enter No. of students who play Football :-4
Enter name of student 1: 1
Enter name of student 2: 3
Enter name of student 3: 4
Enter name of student 4: 5
Students who play both Cricket and Badminton :  ['1', '7']
Students who play either cricket or badminton but not both :  ['2', '3', '5',
'6']
Number of students who play neither cricket nor badminton :  1
Number of students who play cricket and football but not badminton :  1

'''
```

```
'''
//===============================================================================
// Name        : 21465_Pract2.py
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Write a Python program to compute following operations on
String:
                a) To display word with the longest length
                b) To determines the frequency of occurrence of particular
character in the string
                c) To check whether given string is palindrome or not
                d) To display index of first appearance of the substring
                e) To count the occurrences of each word in a given string
                (Do not use string built-in functions)
//===============================================================================

'''


class Str_Op:
    def present(self, v, li):
        for i in range(len(li)):
            if v == li[i]:
                return True
        else:
            return False

    def not_present(self, v, li):
        for i in range(len(li)):
            if v == li[i]:
                return False
        else:
            return True

    def uniqueChars(self, str):
        res = []
        for i in range(0, self.length(str)):
            if self.not_present(str[i], res):
                res.append(str[i])
        return res

    def uniqueWords(self, str):
        res = []
        li = self.Sep_Word(str)
        for i in range(0, self.lengthList(li)):
            if self.not_present(li[i], res):
                res.append(li[i])
        return res

    def Sep_Word(self, str):
        res = []
        word = ""
        str = str + " "
        for c in str:
            if c != " ":
                word = word + c
            else:
                res.append(word)
                word = ""
        return res

    def longest(self):
        str = input("Enter your String : ")

        li = self.Sep_Word(str)

        longest_word = ""
        longest_len = self.length(longest_word)
        current_word = ""
        current_len = self.length(current_word)

        print(li)

        for word in li:
            current_word = word
            current_len = self.length(word)

            if current_len >= longest_len:
                longest_word = current_word
                longest_len = current_len

        print("Longest Word =", longest_word)
        print("Length of longest word =", longest_len)

    def occurance_char(self):
        str = input("Enter your String : ")

        chars = self.uniqueChars(str)
        count = [0] * self.lengthList(chars)

        for i in range(0, self.lengthList(chars)):
            for j in range(0, self.length(str)):
                if str[j] == chars[i]:
                    count[i] = count[i] + 1

        for i in range(0, self.lengthList(chars)):
            print(f"Count of {chars[i]} : {count[i]}")

    def occurance_word(self):
        str = input("Enter your String : ")

        li = self.Sep_Word(str)

        words = self.uniqueWords(str)
        count = [0] * self.lengthList(words)

        for i in range(0, self.lengthList(words)):
            for j in range(0, self.lengthList(li)):
                if li[j] == words[i]:
                    count[i] = count[i] + 1

        for i in range(0, self.lengthList(words)):
            print(f"Count of {words[i]} : {count[i]}")

    def substring(self):
        str = input("Enter your String : ")
        p = input("Enter Sub String to Search : ")

        strl = self.length(str)
        pl = self.length(p)

        found = False
        index = -1

        for i in range(strl - pl + 1):
            match = True
            for j in range(pl):
                if str[i + j] != p[j]:
                    match = False
                    break
                if match:
                    index = i
                    found = True

        if found:
            print(f"{p} found at {index}th index")
        else:
            print(f"Not Found !!")

    def palindrome(self):
        str = input("Enter your String : ")
        isPalindrome = True

        for i in range(0, self.length(str) // 2):
            if not str[i] == str[-1-i]:
                isPalindrome = False
                break
        if isPalindrome:
            print("String is a Palindrome !!")
        else:
            print("String is not a Palindrome !!")

    def length(self, str):
        str = str + "\0"
        i = 0
        while str[i] != "\0":
            i = i + 1
        return i

    def lengthList(self, li):
        li.append("\0")
        i = 0
        while li[i] != "\0":
            i = i + 1
        return i


def main():
    while True:
        obj = Str_Op()

        print("1. Display Longest Word")
        print("2. Display Frequency of occurence of Characters")
        print("3. Check Whether given string is palindrome or not")
        print("4. Display index of 1st appearance of sub string")
        print("5. Display frequency of occurence of Words")
        print("6. Exit")

        ch = int(input("Enter choice"))

        if ch == 1:
            obj.longest()
        elif ch == 2:
            obj.occurance_char()
        elif ch == 3:
            obj.palindrome()
        elif ch == 4:
            obj.substring()
        elif ch == 5:
            obj.occurance_word()
        elif ch == 6:
            break
        else:
            print("Enter Valid Input")


main()


'''
OUTPUT:

$ python pract2.py
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice1
Enter your String : Hello!! I am Chaitanya
['Hello!!', 'I', 'am', 'Chaitanya']
Longest Word = Chaitanya
Length of longest word = 9
1. Display Longest Word
2. Display Frequency of occurence of Characters
```

```
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice2
Enter your String : Heleleleleleo
Count of H : 1
Count of e : 6
Count of l : 5
Count of o : 1
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice3
Enter your String : IammaI
String is a Palindrome !!
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice1
Enter your String : Helloeleleleeello
['Helloeleleleeello']
Longest Word = Helloeleleleeello
Length of longest word = 16
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice3
Enter your String : rhrh
String is not a Palindrome !!
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice4
Enter your String : nandnandan
Enter Sub String to Search : an
an found at 8th index
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
```

```
6. Exit
Enter choice5
Enter your String : Hello I am Chaitanya, I am a Student I am currently admitted
to PICT
Count of Hello : 1
Count of I : 3
Count of am : 3
Count of Chaitanya, : 1
Count of a : 1
Count of Student : 1
Count of currently : 1
Count of admitted : 1
Count of to : 1
Count of PICT : 1
1. Display Longest Word
2. Display Frequency of occurence of Characters
3. Check Whether given string is palindrome or not
4. Display index of 1st appearance of sub string
5. Display frequency of occurence of Words
6. Exit
Enter choice6

'''
```

```
'''
//===============================================================================
// Name        : 21465_Pract3.py
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Write a python program to compute following computation on
matrix:
                a) Addition of two matrices
                b) Subtraction of two matrices
                c) Multiplication of two matrices
                d) Transpose of a matrix
//===============================================================================

'''


class Matrix_Op:

    def __init__(self, m1, m2, r, c):
        self.m1 = m1
        self.m2 = m2
        self.r = r
        self. c = c


def add(m1, m2, r, c):
    res = []
    r1 = []
    for i in range(r):
        for j in range(c):
            v = m1[i][j] + m2[i][j]
            r1.append(v)
        res.append(r1)
        r1 = []
    return res


def sub(m1, m2, r, c):
    res = []
    r1 = []
    for i in range(r):
        for j in range(c):
            v = m1[i][j] - m2[i][j]
            r1.append(v)
        res.append(r1)
        r1 = []
    return res


def mult(m1, m2, r, c):
    res = []

    for i in range(0, r):
        row = []
        for j in range(0, c):
            sum_val = 0
            for k in range(0, r):
                sum_val += m1[i][k] * m2[k][j]
            row.append(sum_val)
        res.append(row)

    return res


def trans(m):
    res = [[0]*len(m)]*len(m[0])
    for i in range(len(m)):
        for j in range(len(m[0])):
            res[i][j] = m[j][i]
    return res


r = int(input("Enter No. of rows : "))
c = int(input("Enter No. of columns : "))

m1 = []
m2 = []
r1 = []

print("Enter Matrix elements for Matrix 1 =>")

for i in range(r):
    for j in range(c):
        v = int(input(f"Enter value for row {i + 1} column {j + 1} : "))
        r1.append(v)
    m1.append(r1)
    r1 = []

print(m1)

print("Enter Matrix elements for Matrix 2 =>")

for i in range(r):
    for j in range(c):
        v = int(input(f"Enter value for row {i + 1} column {j + 1} : "))
        r1.append(v)
    m2.append(r1)
    r1 = []

print(m2)

print("Addition of 2 Matrices = ", add(m1, m2, r, c))
print("Subtraction of 2 Matrices = ", sub(m1, m2, r, c))
print("Multiplication of 2 Matrices = ", mult(m1, m2, r, c))
print("Transpose of  Matrix 1 = ", trans(m1))


'''
```

OUTPUT :

```
$ python pract3.py
Enter No. of rows : 3
Enter No. of columns : 1
Enter Matrix elements for Matrix 1 =>
Enter value for row 1 column 1 : 1
Enter value for row 2 column 1 : 2
Enter value for row 3 column 1 : 3
[[1], [2], [3]]
Enter Matrix elements for Matrix 2 =>
Enter value for row 1 column 1 : 1
Enter value for row 2 column 1 : 2
Enter value for row 3 column 1 : 3
[[1], [2], [3]]
Addition of 2 Matrices =  [[2], [4], [6]]
Subtraction of 2 Matrices =  [[0], [0], [0]]
Transpose of Matrix 1 = [[3, 2, 1]]

'''
```

```
'''
//===============================================================================
// Name         : 21465_Pract4.py
// Author       : Chaitanya Paraskar
// Roll No.     : 21465
// Aim          : a) Write a Python program to store roll numbers of student in
array
                    who attended training program in random order.
                    Write function for searching whether particular student
attended
                    training program or not, using Linear search and Sentinel
search.
                 b) Write a Python program to store roll numbers of student
array
                    who attended training program in sorted order.
                    Write function for searching whether particular student
attended
                    training program or not, using Binary search and Fibonacci
search.
//===============================================================================

'''


class Sentinal:
    def __init__(self):
        n = int(input("Enter No. of elements in Array : "))
        self.arr = [None] * (n+1)
        self.n = n

        for i in range(n):
            print("i = ", i)
            e = int(input(f"Enter {i+1}th element: "))
            self.arr[i] = e

    def search(self, x):
        self.arr[self.n] = x
        i = 0

        while self.arr[i] != x:
            i = i+1

        if i != self.n:
            return i
        return -1

#
--------------------------------------------------------------------------------
-----------------------


class Sequential:
    def __init__(self):
        n = int(input("Enter No. of elements in Array : "))
        self.arr = [None] * n
        self.n = n
        for i in range(n):
            e = int(input(f"Enter {i+1}th element: "))
            self.arr[i] = e

    def search(self, x):
        i = 0
        for i in range(self.n):
            if self.arr[i] == x:
                return i
        return -1

#
--------------------------------------------------------------------------------
-----------------------


class Binary:
    def __init__(self):
        n = int(input("Enter No. of elements in Array : "))
        self.arr = [None] * n
        self.n = n
        for i in range(n):
            e = int(input(f"Enter {i+1}th element: "))
            self.arr[i] = e
        self.sort()

    def sort(self):
        swapped = False

        for i in range(self.n-1):
            for j in range(0, self.n-i-1):
                if self.arr[j] > self.arr[j + 1]:
                    swapped = True
                    self.arr[j], self.arr[j + 1] = self.arr[j + 1], self.arr[j]

            if not swapped:
                return

    def search(self, x):
        min = 0
        max = self.n-1
        mid = (min + max)//2
        found = False

        while min <= max:
            mid = (min + max)//2

            if self.arr[mid] == x:
                found = True
                break
            elif self.arr[mid] > x:
                max = mid - 1
            elif self.arr[mid] < x:
                min = mid + 1

        if found:
            return mid

        return -1

#
--------------------------------------------------------------------------------
----------------------


class Fibonacci:
    def __init__(self):
        n = int(input("Enter No. of elements in Array : "))
        self.arr = [None] * n
        self.n = n
        for i in range(n):
            e = int(input(f"Enter {i+1}th element: "))
            self.arr[i] = e
        print("arr = ", self.arr)
        self.sort()
        print("arr = ", self.arr)

    def sort(self):
        swapped = False

        for i in range(self.n-1):
            for j in range(0, self.n-i-1):
                if self.arr[j] > self.arr[j + 1]:
                    swapped = True
                    self.arr[j], self.arr[j + 1] = self.arr[j + 1], self.arr[j]

            if not swapped:
                return

    def fibo(self, m):
        a = 0
        b = 1
        c = -1

        if m < 0:
            return -1
        if m == 0:
            return a
        if m == 1:
            return b
        else:
            for i in range(2, m+1):
                temp = a + b
                a = b
                b = temp
            return b

    def search(self, x):

        m = 0
        # increment m till fibo(m) is less than n
        while self.fibo(m) < self.n:
            m = m+1

        # init offset to -1
        offset = -1

        # iterate till fibo(m) is greater than 1
        while self.fibo(m) > 1:

            # init mid to minimum of given 2 terms
            mid = self.min(offset+self.fibo(m - 2), self.n - 1)

            # if x is greater than arr[mid] set offset = mid and decrement m by
1
            if x > self.arr[mid]:
                offset = mid
                m = m - 1
            # if x is smaller than arr[mid] decrement m by 2
            elif x < self.arr[mid]:
                m = m - 2
            # if x is equal to arr[mid]
            # i.e. x is found
            elif x == self.arr[mid]:
                return mid

        if (not (self.fibo(m - 1) > 0)) and (self.arr[offset + 1] == x):
            return offset + 1
        return -1

    def min(self, a, b):
        if a > b:
            return a
        else:
            return b

#
--------------------------------------------------------------------------------
----------------------


def main():
    while True:
        print("1. Sequential Search")
        print("2. Sentinal Search")
        print("3. Binary Search")
        print("4. Fibonacci Search")
        print("5. Exit")
        ch = int(input("Enter choice"))

        if ch == 1:
            obj = Sequential()
            x = int(input("Enter Element to be Searched :"))
            e = obj.search(x)
            if e == -1:
                print("Element not Found !!")
            else:
                print(f"Element found at {e}th index")
        elif ch == 2:
            obj = Sentinal()
            x = int(input("Enter Element to be Searched :"))
            e = obj.search(x)
            if e == -1:
                print("Element not Found !!")
            else:
                print(f"Element found at {e}th index")
        elif ch == 3:
            obj = Binary()
            x = int(input("Enter Element to be Searched :"))
            e = obj.search(x)
            if e == -1:
                print("Element not Found !!")
            else:
                print(f"Element found at {e}th index")
        elif ch == 4:
            obj = Fibonacci()
            x = int(input("Enter Element to be Searched :"))
            e = obj.search(x)
            if e == -1:
                print("Element not Found !!")
            else:
                print(f"Element found at {e}th index")
        elif ch == 5:
            break
        else:
            print("Enter Valid Input")


main()

'''

OUTPUT :

$ python pract4.py
1. Sequential Search
2. Sentinal Search
3. Binary Search
4. Fibonacci Search
5. Exit
```

```
Enter choice1
Enter No. of elements in Array : 5
Enter 1th element: 1
Enter 2th element: 2
Enter 3th element: 3
Enter 4th element: 4
Enter 5th element: 5
Enter Element to be Searched :4
Element found at 3th index
1. Sequential Search
2. Sentinal Search
3. Binary Search
4. Fibonacci Search
5. Exit
Enter choice2
Enter No. of elements in Array : 5
Enter 1th element: 1
Enter 2th element: 2
Enter 3th element: 3
Enter 4th element: 4
Enter 5th element: 5
Enter Element to be Searched :2
Element found at 1th index
1. Sequential Search
2. Sentinal Search
3. Binary Search
4. Fibonacci Search
5. Exit
Enter choice3
Enter No. of elements in Array : 5
Enter 1th element: 1
Enter 2th element: 2
Enter 3th element: 3
Enter 4th element: 4
Enter 5th element: 5
Enter Element to be Searched :6
Element not Found !!
1. Sequential Search
2. Sentinal Search
3. Binary Search
4. Fibonacci Search
5. Exit
Enter choice4
Enter No. of elements in Array : 1
Enter 1th element: 4
arr =  [4]
Enter Element to be Searched :1
Element not Found !!
1. Sequential Search
2. Sentinal Search
3. Binary Search
4. Fibonacci Search
5. Exit
Enter choice4
Enter No. of elements in Array : 5
Enter 1th element: 1
Enter 2th element: 2
Enter 3th element: 3
Enter 4th element: 4
Enter 5th element: 5
arr =  [1, 2, 3, 4, 5]
arr =  [1, 2, 3, 4, 5]
Enter Element to be Searched :1
Element found at 0th index
1. Sequential Search
2. Sentinal Search
3. Binary Search
4. Fibonacci Search
5. Exit
Enter choice5

'''
```

```
'''
//=============================================================================
// Name        : 21465_Pract5.py
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Write a python program to store second year percentage of
students in array.
                 Write function for sorting array of floating point numbers in
ascending order using-
                 a) Insertion sort
                 b) Shell Sort
                 and display top five scores
//=============================================================================
'''


class Array:

    def __init__(self):
        self.n = int(input("Enter Total No. of elements in Array : "))
        self.arr = [-1] * self.n

        for i in range(0, self.n):
            e = int(input(f"Enter {i}th element : "))
            self.arr[i] = e

        print(f"Entered array = {self.arr}")
        self.selectionSort()

    def bubbleSort(self):
        swap = False
        i = 0

        while i < self.n-1:
            j = 0
            while j < self.n - i - 1:
                if self.arr[j] > self.arr[j+1]:
                    temp = self.arr[j]
                    self.arr[j] = self.arr[j+1]
                    self.arr[j+1] = temp
                    swap = True
                j = j+1

            print(f"array after pass {i+1} : {self.arr}")

            if not swap:
                break

            i = i+1

        print(f"Sorted Array : {self.arr}")
        return

    def selectionSort(self):
        swap = False
        for i in range(self.n):

            min_idx = i

            for j in range(i+1, self.n):
                if self.arr[min_idx] > self.arr[j]:
                    min_idx = j
                    swap = True

            self.arr[i], self.arr[min_idx] = self.arr[min_idx], self.arr[i]

            if not swap:
                break

            print(f"array after pass {i+1} : {self.arr}")

        print(f"Sorted Array : {self.arr}")

    def insertionSort(self):
        i = 0

        while i < self.n-1:

            if self.arr[i] > self.arr[i+1]:

                temp = self.arr[i]
                self.arr[i] = self.arr[i+1]
                self.arr[i+1] = temp

                j = i

                while j > 0:

                    if self.arr[j] < self.arr[j-1]:

                        temp = self.arr[j]
                        self.arr[j] = self.arr[j-1]
                        self.arr[j-1] = temp

                        j = j-1
                    else:
                        break
            i = i+1
            print(f"array after pass {i} : {self.arr}")

        print(f"Sorted Array : {self.arr}")

    def ShellSort(self):
        gap = self.n//2
        while gap > 0:
            j = gap
            while j < self.n:
                i = j-gap
                while i >= 0:
                    if self.arr[i+gap] > self.arr[i]:

                        break
                    else:
                        self.arr[i+gap], self.arr[i] = self.arr[i],
self.arr[i+gap]

                    i = i-gap
                print(f"array after pass : {self.arr}")
                j += 1
            gap = gap//2

        print(f"Sorted Array : {self.arr}")


a = Array()


'''

OUTPUT

INSERTION SORT :

$ python pract5.py
Enter Total No. of elements in Array : 5
Enter 0th element : 5
Enter 1th element : 4
Enter 2th element : 3
Enter 3th element : 2
Enter 4th element : 1
Entered array = [5, 4, 3, 2, 1]
array after pass 1 : [4, 5, 3, 2, 1]
array after pass 2 : [3, 4, 5, 2, 1]
array after pass 3 : [2, 3, 4, 5, 1]
array after pass 4 : [1, 2, 3, 4, 5]
Sorted Array : [1, 2, 3, 4, 5]


BUBBLE SORT :

$ python pract5.py
Enter Total No. of elements in Array : 5
Enter 0th element : 5
Enter 1th element : 4
Enter 2th element : 3
Enter 3th element : 2
Enter 4th element : 1
Entered array = [5, 4, 3, 2, 1]
```

```
array after pass 1 : [4, 3, 2, 1, 5]
array after pass 2 : [3, 2, 1, 4, 5]
array after pass 3 : [2, 1, 3, 4, 5]
array after pass 4 : [1, 2, 3, 4, 5]
Sorted Array : [1, 2, 3, 4, 5]


SHELL SORT :

$ python pract5.py
Enter Total No. of elements in Array : 5
Enter 0th element : 5
Enter 1th element : 4
Enter 2th element : 3
Enter 3th element : 2
Enter 4th element : 1
Entered array = [5, 4, 3, 2, 1]
array after pass : [3, 4, 5, 2, 1]
array after pass : [3, 2, 5, 4, 1]
array after pass : [1, 2, 3, 4, 5]
array after pass : [1, 2, 3, 4, 5]
array after pass : [1, 2, 3, 4, 5]
array after pass : [1, 2, 3, 4, 5]
array after pass : [1, 2, 3, 4, 5]
Sorted Array : [1, 2, 3, 4, 5]

OUTPUT SELECTION SORT :

$ python pract5.py
Enter Total No. of elements in Array : 5
Enter 0th element : 5
Enter 1th element : 4
Enter 2th element : 3
Enter 3th element : 2
Enter 4th element : 1
Entered array = [5, 4, 3, 2, 1]
array after pass 1 : [1, 4, 3, 2, 5]
array after pass 2 : [1, 2, 3, 4, 5]
array after pass 3 : [1, 2, 3, 4, 5]
array after pass 4 : [1, 2, 3, 4, 5]
array after pass 5 : [1, 2, 3, 4, 5]
Sorted Array : [1, 2, 3, 4, 5]

'''
```

```
'''
//===============================================================================
// Name         : 21465_Pract6.py
// Author       : Chaitanya Paraskar
// Roll No.     : 21465
// Aim          : Write a python program to store first year percentage of
students in array.
                 Write function for sorting array of floating  point numbers in
ascending order using
                 quick sort and display top five scores.
//===============================================================================

'''


class Array:
    def __init__(self):
        self.n = int(input("Enter Total No. of elements in Array : "))
        self.arr = [-1] * self.n
        for i in range(0, self.n):
            e = int(input(f"Enter {i}th element : "))
            self.arr[i] = e
        print(f"Entered array = {self.arr}")
        self.i = 0
        self.quicksort(0, self.n-1)

    def partition(self, arr, low, high):
        pivot = arr[high]
        i = low - 1
        for j in range(low, high):
            if arr[j] <= pivot:
                i = i + 1
                (arr[i], arr[j]) = (arr[j], arr[i])
        (arr[i + 1], arr[high]) = (arr[high], arr[i + 1])
        return i + 1

    def quicksort(self, low, high):
        self.i = self.i + 1
        print(f"Array after Recursive Call {self.i} : ",
              self.arr, " low = ", low, " high = ", high)
        if low < high:
            pi = self.partition(self.arr, low, high)
            self.quicksort(low, pi - 1)
            self.quicksort(pi + 1, high)


a = Array()

'''
OUTPUT:

$ python pract6.py
```

```
Enter Total No. of elements in Array : 5
Enter 0th element : 5
Enter 1th element : 4
Enter 2th element : 3
Enter 3th element : 2
Enter 4th element : 1
Entered array = [5, 4, 3, 2, 1]
Array after Recursive Call 1 :  [5, 4, 3, 2, 1]  low =  0  high =  4
Array after Recursive Call 2 :  [1, 4, 3, 2, 5]  low =  0  high =  -1
Array after Recursive Call 3 :  [1, 4, 3, 2, 5]  low =  1  high =  4
Array after Recursive Call 4 :  [1, 4, 3, 2, 5]  low =  1  high =  3
Array after Recursive Call 5 :  [1, 2, 3, 4, 5]  low =  1  high =  0
Array after Recursive Call 6 :  [1, 2, 3, 4, 5]  low =  2  high =  3
Array after Recursive Call 7 :  [1, 2, 3, 4, 5]  low =  2  high =  2
Array after Recursive Call 8 :  [1, 2, 3, 4, 5]  low =  4  high =  3
Array after Recursive Call 9 :  [1, 2, 3, 4, 5]  low =  5  high =  4

'''
```

```
//===============================================================================
// Name         : 21465_Pract7.cpp
// Author       : Chaitanya Paraskar
// Roll No.     : 21465
// Aim          : Write C++ program for storing binary number using doubly linked
lists.
//                Write functions- a) To compute 1's and 2's complement
//                                 b) Add two binary numbers
//===============================================================================

#include "iostream"
using namespace std;

class Node
{
private:
    int data;
    Node *next;

    friend class Number;
};

class Number
{
private:
    Node *start;

public:
    Number(string num);
    void display();
    void reverseDisplay(Node *ptr);
    Number *add(Number *n2);
    void onecomp();
    void twocomp();
};

Number::Number(string num)
{
    for (int i = 0; i < num.length(); i++)
    {
        char ch = num[i];
        Node *n = new Node();

        if (ch == '0')
            n->data = 0;
        else
            n->data = 1;

        n->next = this->start;
        this->start = n;
    }
}

void Number::display()
```

```
{
    // Node *ptr = this->start;

    // cout << "Number -> ";
    // while (ptr != NULL)
    // {
    //     cout << ptr->data << " -> ";
    //     ptr = ptr->next;
    // }
    // cout << "NULL" << endl;

    this->reverseDisplay(this->start);
    cout << endl;
}

void Number::reverseDisplay(Node *ptr)
{
    if (ptr->next != NULL)
    {
        this->reverseDisplay(ptr->next);
        cout << ptr->data;
    }
    else
    {
        cout << ptr->data;
    }
}

void Number::onecomp()
{
    Node *ptr = this->start;

    while (ptr != NULL)
    {
        if (ptr->data == 0)
            ptr->data = 1;
        else
            ptr->data = 0;

        ptr = ptr->next;
    }
}

void Number::twocomp()
{
    cout << "After One's Complement :-" << endl;
    this->onecomp();
    cout << "n1 => ";
    this->display();

    Node *ptr = this->start;
    int carry = 1;

    while (ptr != NULL)
    {
        if (ptr->data == 0)
        {
            ptr->data = carry;
            carry = 0;
        }
        if (ptr->data == 1 && carry == 0)
        {
        }
        if (ptr->data == 1 && carry == 1)
        {
            ptr->data = 0;
            carry = 1;
        }

        ptr = ptr->next;
    }

    if (carry == 1)
    {
        Node *n = new Node();
        n->data = 1;
        n->next = this->start;
        this->start = n;
    }
}

Number *Number::add(Number *n)
{
    string str = "";
    int carry = 0;
    Node *n1 = this->start;
    Node *n2 = n->start;

    while (n1 != NULL && n2 != NULL)
    {
        if (n1->data == 0 && n2->data == 0)
        {
            if (carry == 0)
                str = "0" + str;
            else
                str = "1" + str;

            carry = 0;
        }
        if (n1->data == 0 && n2->data == 1)
        {
            if (carry == 0)
            {
                str = "1" + str;
                carry = 0;
            }
            else
            {
                str = "0" + str;
                carry = 1;
            }
        }
        if (n1->data == 1 && n2->data == 0)
        {
            if (carry == 0)
            {
                str = "1" + str;
                carry = 0;
            }
            else
            {
                str = "0" + str;
                carry = 1;
            }
        }
        if (n1->data == 1 && n2->data == 1)
        {
            if (carry == 0)
            {
                str = "0" + str;
                carry = 1;
            }
            else
            {
                str = "1" + str;
                carry = 1;
            }
        }
        n1 = n1->next;
        n2 = n2->next;
    }

    if (carry == 1)
    {
        str = "1" + str;
    }

    Number *res = new Number(str);

    return res;
}

int main()
{
    Number *n1 = new Number("101101");
    Number *n2 = new Number("001101");

    cout << "n1 => ";
    n1->display();
    cout << "n2 => ";
    n2->display();
    Number *n3 = n1->add(n2);
    cout << "n3 => ";
    n3->display();
    cout << "********************************" << endl;
    cout << "n1 => ";
    n1->display();
    cout << "After One's Complement :-" << endl;
    n1->onecomp();
    cout << "n1 => ";
    n1->display();
    cout << "********************************" << endl;
    cout << "n1 => ";
    n1->display();
    n1->twocomp();
    cout << "After Two's Complement :-" << endl;
    cout << "n1 => ";
    n1->display();
    return 0;
}

/*

Output:

$ g++ Pract7Binary.cpp -o out && ./out
n1 => 101101
n2 => 001101
n3 => 111010
********************************
n1 => 101101
After One's Complement :-
n1 => 010010
********************************
n1 => 010010
After One's Complement :-
n1 => 101101
After Two's Complement :-
n1 => 101110

*/
```

```
//============================================================================
// Name        : 21465_Pract8.cpp
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Second year Computer Engineering class, set A of students like
Vanilla Icecream and set B
//               of students like butterscotch ice-cream. Write C++ program to
store two sets using linked list.
//               compute and display - a) Set of students who like both vanilla
and butterscotch
//                                     b) Set of students who like either
vanilla or butterscotch or not both
//                                     c) Number of students who like neither
vanilla nor butterscotch
//============================================================================

#include "iostream"
using namespace std;

class Student
{
public:
    int rno;
    Student *next;
    Student(int rno)
    {
        this->rno = rno;
    }
};

class Set
{
    Student *head;
    int count = 0;

    void add(int rno)
    {
        Student *s = new Student(rno);
        s->next = this->head;
        head = s;

        count++;
    }

    friend Set *initUniversal();

public:
    Set() {}
    Set(Set *R)
    {
        int n;

        cout << "No. of Students : ";
        cin >> n;
        int i = 0;

        while (i < n)
        {
            int rno;

            cout << "Enter Roll No. of " << i + 1 << "th student : ";
            cin >> rno;

            if (this->not_in(rno) && R->in(rno))
            {
                this->add(rno);
                i++;
            }
            else
            {
                cout << "Already Present in Set !!" << endl;
            }
        }
    }

    void display()
    {
        cout << "Set : [";

        Student *ptr = this->head;

        while (ptr != NULL)
        {
            cout << ptr->rno;

            if (ptr->next != NULL)
                cout << ", ";

            ptr = ptr->next;
        }

        cout << "]" << endl;
    }

    bool in(int n)
    {
        Student *ptr = this->head;

        while (ptr != NULL)
        {
            if (ptr->rno == n)
                return true;
            ptr = ptr->next;
        }

        return false;
    }
    bool not_in(int n)
    {
        Student *ptr = this->head;

        while (ptr != NULL)
        {
            if (ptr->rno == n)
                return false;
            ptr = ptr->next;
        }

        return true;
    }

    int getCount()
    {
        return this->count;
    }

    static Set *And(Set *A, Set *B);
    static Set *Or(Set *A, Set *B);
    static Set *Difference(Set *A, Set *B);
};

Set *Set::And(Set *A, Set *B)
{
    Set *R = new Set();

    Student *a = A->head;

    while (a != NULL)
    {
        if (B->in(a->rno))
            R->add(a->rno);

        a = a->next;
    }

    return R;
}

Set *Set::Or(Set *A, Set *B)
{
    Set *R = new Set();

    Student *a = A->head;

    while (a != NULL)
    {
        if (R->not_in(a->rno))
            R->add(a->rno);
        a = a->next;
    }
    Student *b = B->head;

    while (b != NULL)
    {
        if (R->not_in(b->rno))
            R->add(b->rno);
        b = b->next;
    }

    return R;
}

Set *Set::Difference(Set *A, Set *B)
{
    Set *R = new Set();

    Student *a = A->head;

    while (a != NULL)
    {
        if (B->not_in(a->rno))
            R->add(a->rno);
        a = a->next;
    }

    return R;
}

Set *initUniversal()
{

    cout << "Enter Universal Set : " << endl;
    Set *U = new Set();
    int n;

    cout << "No. of Students : ";
    cin >> n;

    int i = 0;

    while (i < n)
    {
        int rno;

        cout << "Enter Roll No. of " << i + 1 << "th student : ";
        cin >> rno;

        if (U->not_in(rno))
        {
            U->add(rno);
            i++;
        }
        else
        {
            cout << "Already Present in Set !!" << endl;
        }
    }

    return U;
}

int main()
{

    Set *U = initUniversal();

    cout << "Enter Set of Roll No. of Students who like Vanilla : " << endl;
    Set *Vanilla = new Set(U);
    cout << "Enter Set of Roll No. of Students who like Butter Scotch : " <<
endl;
    Set *ButterScotch = new Set(U);

    cout << "Roll No. of Students who Like Vanilla : " << endl;
    Vanilla->display();

    cout << "Roll No. of Students who Like Butter Scotch : " << endl;
    ButterScotch->display();

    cout << "Students who like both Vanilla and Butter Scotch : " << endl;
    Set *a = Set::And(Vanilla, ButterScotch);
    a->display();

    cout << "Students who like Either Vanilla or Butter Scotch but not both : "
<< endl;
    Set *x = Set::Or(Vanilla, ButterScotch);
    Set *y = Set::And(Vanilla, ButterScotch);
    Set *b = Set::Difference(x, y);
    b->display();

    Set *c = Set::Difference(U, x);
    int count = c->getCount();
    cout << "No. of Students who like neither Vanilla, nor Butter Scotch : " <<
count << endl;

    return 0;
}

/*

Output:

$ g++ Pract8.cpp -o out && ./out
Enter Universal Set :
No. of Students : 4
Enter Roll No. of 1th student : 111
Enter Roll No. of 2th student : 222
Enter Roll No. of 3th student : 333
Enter Roll No. of 4th student : 444
Enter Set of Roll No. of Students who like Vanilla :
No. of Students : 2
Enter Roll No. of 1th student : 111
Enter Roll No. of 2th student : 222
Enter Set of Roll No. of Students who like Butter Scotch :
No. of Students : 2
Enter Roll No. of 1th student : 111
Enter Roll No. of 2th student : 333
Roll No. of Students who Like Vanilla :
Set : [222, 111]
Roll No. of Students who Like Butter Scotch :
Set : [333, 111]
Students who like both Vanilla and Butter Scotch :
Set : [111]
Students who like Either Vanilla or Butter Scotch but not both :
Set : [222, 333]
No. of Students who like neither Vanilla, nor Butter Scotch : 1

*/
```

```
//============================================================================
// Name         : 21465_Pract9.cpp
// Author       : Chaitanya Paraskar
// Roll No.     : 21465
// Aim          : In any language program mostly syntax error occurs due to
unbalancing delimiter such as(), {}, [].
//               Write C++ program using stack to check whether given expression
is well parenthesized or not.
//============================================================================

#include <iostream>
using namespace std;

class StackOp
{
private:
    char stack[10];

public:
    void checkPara()
    {
        char exp[10], ch;
        int flag = 0;
        int top = -1;
        cout << "Enter the expression to check " << endl;
        cin >> exp;

        for (int i = 0; exp[i] != '\0'; i++)
        {
            if (exp[i] == '{' || exp[i] == '[' || exp[i] == '(')
            {
                top++;
                stack[top] = exp[i];
            }
            else if (exp[i] == '}' || exp[i] == ']' || exp[i] == ')')
            {
                ch = stack[top];
                top--;

                switch (exp[i])
                {
                case '}':
                    if (ch != '{')
                    {
                        flag = 1;
                    }
                    break;

                case ')':

                    if (ch != '(')
                    {
                        flag = 1;
                    }
                    break;

                case ']':

                    if (ch != '[')
                    {
                        flag = 1;
                    }
                    break;
                }

                if (flag == 1)
                {
                    break;
                }
            }
            else
            {
                // continue for character
            }
        }

        if (top != -1 && flag == 1)
        {
            cout << "The given expression is not well paranthesized " << endl;
        }
        else
        {
            cout << "The given expression is well parathesized " << endl;
        }
    }
};

int main()
{
    StackOp obj;
    obj.checkPara();

    return 0;
}

/**

Output:

$ g++ Pract9.cpp -o out && ./out
Enter the expression to check
({{[a+b)]}
The given expression is not well paranthesized

$ g++ Pract9.cpp -o out && ./out
Enter the expression to check
[{(a+b)}]
The given expression is well parathesized
```

```
*/
```

```
//==============================================================================
// Name         : 21465_Pract10.cpp
// Author       : Chaitanya Paraskar
// Roll No.     : 21465
// Aim          : Implement C++ program for expression conversion as infix to
postfix and its evaluation using stack based on given
//                conditions : 1. Operands and operator, both must be single
character.
//                             2. Input Postfix expression must be in a desired
format.
//                             3. Only '+', '-', '*' and '/ ' operators are
expected.
//==============================================================================

#include <iostream>
using namespace std;

class Stack
{
private:
    char arr[20];
    int top = -1;

public:
    void display()
    {
        cout << "Stack : ";
        for (int i = 0; i < 20; i++)
        {
            char ch = arr[i];
            cout << ch << " ";
        }
        cout << "\n";
    }

    void push(char ch)
    {
        if (top < 19)
        {
            top++;
            arr[top] = ch;
        }
        else
        {
            cout << "Stack is Full !!" << endl;
        }
    }

    char pop()
    {
        char ch = arr[top];
        top--;
        return ch;
    }
    char getTop()
    {
        return arr[top];
    }

    bool isEmpty()
    {
        if (top == -1)
        {
            return true;
        }

        return false;
    }

    void clear()
    {
        while (this->top != -1)
            this->pop();
    }
};

class Node
{
public:
    char key;
    int val;
    Node *next;
};

class LinkedList
{
private:
    Node *start;

public:
    void add(char var, int val)
    {
        Node *n = new Node();
        n->key = var;
        n->val = val;

        n->next = start;
        start = n;
    }

    int getVal(char ch)
    {
        Node *ptr = this->start;

        while (ptr != NULL)
        {
            if (ptr->key == ch)
```

```
                break;

            ptr = ptr->next;
        }

        return ptr->val;
    }

    bool in(char n)
    {
        Node *ptr = this->start;

        while (ptr != NULL)
        {
            if (ptr->key == n)
                return true;
            ptr = ptr->next;
        }

        return false;
    }

    int getFirst()
    {
        return this->start->val;
    }

    void display()
    {
        cout << "LinkedList ";
        Node *ptr = this->start;

        while (ptr != NULL)
        {
            cout << "-> (" << ptr->key << ", " << ptr->val << ")";
            ptr = ptr->next;
        }
        cout << " -> NULL" << endl;
    }
};

class Expr
{
private:
    string inex = "";
    string postex = "";
    Stack *stack;
    LinkedList *list;
    static char custChar;

public:
    Expr()
    {
        this->stack = new Stack();
        this->list = new LinkedList();

        cout << "Enter Your Equation : ";
        getline(cin, this->inex);

        cout << "Infix Expression : " << this->inex << endl;
        this->toPost();
        cout << "Postfix Expression : " << this->postex << endl;
        this->eval();
    }

    static int prec(char ch)
    {
        int pr = 0;

        switch (ch)
        {
        case '*':
            pr = 2;
            break;
        case '-':
            pr = 1;
            break;
        case '+':
            pr = 1;
            break;
        case '/':
            pr = 2;
            break;
        case '%':
            pr = 2;
            break;
        case '^':
            pr = 3;
            break;
        }

        return pr;
    }

    void assoc(char ch)
    {
        if (ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == '%')
        {
            char c = this->stack->pop();
            this->postex += c;
            this->stack->push(ch);
        }
        else if (ch == '^')
        {
            this->stack->push(ch);
        }
    }
    void toPost()
    {
        int l = this->inex.length(), i = 0;

        while (i < l)
        {

            char ch = this->inex[i];

            if (ch == '(')
            {
                this->stack->push(ch);
            }
            else if (ch == ')')
            {
                while (this->stack->getTop() != '(')
                {
                    char c = this->stack->pop();
                    postex += c;
                }

                this->stack->pop();
            }
            else if (isalpha(ch))
            {
                postex += ch;

                if (!this->list->in(ch))
                {
                    int val;

                    cout << "Enter Value for variable " << ch << " : ";
                    cin >> val;

                    list->add(ch, val);
                }
            }
            else
            {
                int pch = Expr::prec(ch);
                int ptop = Expr::prec(this->stack->getTop());

                if (pch > ptop)
                {
                    this->stack->push(ch);
                }
                else if (pch < ptop)
                {
                    char c = this->stack->pop();
                    postex += c;

                    //  This Line causes the loop to reiterate for same char in
the expr
                    //  without incrementing value of i.
                    continue;
                }
                else
                {
                    this->assoc(ch);
                }
            }

            cout << "Char : " << ch << endl;
            this->stack->display();
            this->list->display();

            i++;
        }

        while (!this->stack->isEmpty())
        {
            char c = this->stack->pop();

            postex += c;
        }
    }

    void eval()
    {
        this->stack->clear();

        int l = this->postex.length(), i = 0;

        while (i < l)
        {
            char ch = this->postex[i];

            if (isalpha(ch))
            {
                this->stack->push(ch);
            }
            else
            {
                char a = this->stack->pop(), b = this->stack->pop();

                int aval = this->list->getVal(a);
                int bval = this->list->getVal(b);

                int res = Expr::perfOp(aval, bval, ch);

                this->list->add(Expr::custChar, res);
                this->stack->push(Expr::custChar);
                Expr::custChar = (char)((int)Expr::custChar + 1);
            }

            cout << "Char : " << ch << endl;
            this->stack->display();
            this->list->display();
```

```
            i++;
        }

        int res = this->list->getFirst();

        cout << "Evaluation of this Equation is : " << res << endl;
    }

    static int perfOp(int a, int b, char ch)
    {
        int res = 0;

        switch (ch)
        {
        case '*':
            res = b * a;
            break;
        case '-':
            res = b - a;
            break;
        case '+':
            res = b + a;
            break;
        case '/':
            res = b / a;
            break;
        case '%':
            res = b % a;
            break;
        case '^':
            res = b ^ a;
            break;
        }

        return res;
    }
};

char Expr::custChar = 'A';

int main()
{
    Expr *a = new Expr();
    delete a;
    return 0;
}

/*

Output:

$ g++ Pract10.cpp -o out && ./out
Enter Your Equation : (a+b)
```

```
***********************************************
Infix Expression : (a+b)
***********************************************
Char : a
Stack : (
LinkedList  -> NULL
Enter Value for variable a : 1
***********************************************
Char : +
Stack : (
LinkedList -> (a, 1) -> NULL
***********************************************
Char : b
Stack : ( +
LinkedList -> (a, 1) -> NULL
Enter Value for variable b : 2
***********************************************
Char : )
Stack : ( +
LinkedList -> (b, 2)-> (a, 1) -> NULL
***********************************************
Postfix Expression : ab+
***********************************************
Char : a
Stack : a
LinkedList -> (b, 2)-> (a, 1) -> NULL
***********************************************
Char : b
Stack : a b
LinkedList -> (b, 2)-> (a, 1) -> NULL
***********************************************
Char : +
Stack : A b
LinkedList -> (A, 3)-> (b, 2)-> (a, 1) -> NULL
***********************************************
Evaluation of this Equation is : 3
***********************************************

*/
```

```
//=============================================================================
// Name        : 21465_Pract11.cpp
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Queues are frequently used in computer programming, and a
typical example is the creation
//               of a job queue by an operating system. If the operating system
does not use priorities, then
//               the jobs are processed in the order they enter the system.
Write C++ program for simulating job queue.
//               Write functions to add job, display job and delete job from
queue.
//=============================================================================

#include <iostream>
using namespace std;

class p_queue
{
    int pid;
    int priority;
    p_queue *next;

public:
    p_queue(int p, int pr)
    {
        pid = p;
        priority = pr;
        next = NULL;
    }
    friend class Schedule;
};

class Schedule
{
    p_queue *front;
    p_queue *rear;

public:
    Schedule()
    {
        front = NULL;
        rear = NULL;
    }
    void push()
    {
        int p, q;
        cout << "enter priority of process";
        cin >> q;
        cout << "process id ";
        cin >> p;

        p_queue *n = new p_queue(p, q);
        if (front == NULL)
        {
            front = n;
            return;
        }
        else if (front->next == NULL)
        {
            if (n->priority < front->priority)
            {
                n->next = front;
                front = n;
                return;
            }
        }

        p_queue *temp = front;
        if (q < temp->priority)
        {
            n->next = front;
            front = n;
            return;
        }
        while (temp->next != NULL)
        {
            if (temp->next->priority > n->priority)
            {
                n->next = temp->next;
                temp->next = n;
                return;
            }
            temp = temp->next;
        }
        temp->next = n;
    }
    void traverse()
    {
        p_queue *temp = front;
        while (temp != NULL)
        {
            cout << temp->priority << " " << temp->pid << " " << endl;
            temp = temp->next;
        }
    }
    void del()
    {
        p_queue *temp = front;
        cout << "processs in exe " << temp->pid << " " << temp->priority <<
endl;
        front = front->next;
        delete temp;
    }
};
int main()
{
    Schedule obj;
    char c;
    do
    {
        int ch;
        cout << "1.Insert process\n2.Delete process\n enter your choice";
        cin >> ch;

        if (ch == 1)
        {
            obj.push();
            obj.traverse();
        }
        else if (ch == 2)
        {
            obj.del();
            obj.traverse();
        }
        else
            cout << "enter valid data<<endl";

        cout << "Do you want to continue(y)";
        cin >> c;

    } while (c == 'y');

    return 0;
}

/*

Output:

$ g++ Pract11.cpp -o out && ./out
1.Insert process
2.Delete process
enter your choice 1
enter priority of process 2
process id 111
2 111
Do you want to continue(y) y
1.Insert process
2.Delete process
enter your choice 1
enter priority of process 1
process id 222
1 222
2 111
Do you want to continue(y) y
1.Insert process
2.Delete process
enter your choice 1
enter priority of process 3
```

```
process id 333
1 222
2 111
3 333
Do you want to continue(y) y
1.Insert process
2.Delete process
enter your choice 2
processs in exe 222 1
2 111
3 333
Do you want to continue(y) y
1.Insert process
2.Delete process
enter your choice 2
processs in exe 111 2
3 333
Do you want to continue(y) y
1.Insert process
2.Delete process
enter your choice 2
processs in exe 333 3
Do you want to continue(y) y
1.Insert process
2.Delete process
enter your choice

*/
```

```
//============================================================================
// Name        : 21465_Pract12.cpp
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Write program to implement a priority queue in C++ using an
order list / array to store
//               the items in the queue.Create a class that includes the data
items(which should be template)
//               and the priority(which should be int).The order list / array
should contain these objects,
//               with operator<= overloaded so that the items with highest
priority appear at the beginning of
//               the list / array(which will make it relatively easy to retrieve
the highest item.)
//============================================================================

#include <iostream>
using namespace std;

template <class T>
class p_queue
{
    T pid;
    int priority;
    p_queue *next;

public:
    p_queue(T p, int pr)
    {
        pid = p;
        priority = pr;
        next = NULL;
    }
    friend class Schedule;

    bool operator<(const p_queue<T> *other)
    {
        return this->priority < other->priority;
    }

    bool operator>(const p_queue<T> *other)
    {
        return this->priority > other->priority;
    }
};

class Schedule
{
    p_queue<int> *front;
    p_queue<int> *rear;

public:
    Schedule()
    {
        front = NULL;
        rear = NULL;
    }
    void push()
    {
        int p, q;
        cout << "enter priority of process";
        cin >> q;
        cout << "process id ";
        cin >> p;

        p_queue<int> *n = new p_queue(p, q);

        if (front == NULL)
        {
            front = n;
            return;
        }
        else if (front->next == NULL)
        {
            if (n < front)
            {
                n->next = front;
                front = n;
                return;
            }
        }

        p_queue<int> *temp = front;
        if (q < temp->priority)
        {
            n->next = front;
            front = n;
            return;
        }
        while (temp->next != NULL)
        {
            if (temp->next > n)
            {
                n->next = temp->next;
                temp->next = n;
                return;
            }
            temp = temp->next;
        }
        temp->next = n;
    }
    void traverse()
    {
        p_queue<int> *temp = front;
        while (temp != NULL)
        {
            cout << temp->priority << " " << temp->pid << " " << endl;
            temp = temp->next;
        }
    }
    void del()
    {
        p_queue<int> *temp = front;
        cout << "processs in exe " << temp->pid << " " << temp->priority <<
endl;
        front = front->next;
        delete temp;
    }
};
int main()
{

    Schedule obj;
    char c;
    do
    {
        int ch;
        cout << "1.Insert process\n2.Delete process\n enter your choice";
        cin >> ch;

        if (ch == 1)
        {
            obj.push();
            obj.traverse();
        }
        else if (ch == 2)
        {
            obj.del();
            obj.traverse();
        }
        else
            cout << "enter valid data<<endl";

        cout << "Do you want to continue(y)";
        cin >> c;

    } while (c == 'y');

    return 0;
}

/*
Output:

$ g++ Praact13.cpp -o out && ./out
Size of Queue = 10
Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
```

```
6. Exit
Enter your choice: 1
Deque is empty

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 2
Enter data to push front: 123
123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 2
Enter data to push front: 234
234 123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 2
Enter data to push front: 3
3 234 123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 3
Enter data to push back: 456
3 234 123 456

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
```

```
6. Exit
Enter your choice: 4
234 123 456

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 5
234 123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 4
123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 5
Deque is empty

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 6
Exiting Deque Menu
Deque Deleted

*/
```

```
//================================================================================
// Name        : 21465_Pract13.cpp
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : A double-ended queue (deque) is a linear list in which
additions and deletions may be
//               made at either end.Obtain a data representation mapping a deque
into a one dimensional array.
//               Write C++ program to simulate deque with functions to add and
delete elements from either
//               end of the deque.
//================================================================================

#include <iostream>
using namespace std;

class deque
{
    int size;
    int *arr;
    int front;
    int back;

public:
    deque(int _size) : front(-1), back(-1), size(_size)
    {
        arr = new int[_size];
    }

    void push_back(int _data)
    {
        if ((front == 0 && back == size) || (front == back + 1))
        {
            cout << "Deque OverFlow" << endl;
            return;
        }
        else if (front == -1)
            front = back = 0;
        else if (front != 0 && back == size)
            back = 0;
        else
            back++;

        arr[back] = _data;
    }

    void push_front(int _data)
    {
        if ((front == 0 && back == size) || (front == back + 1))
        {
            cout << "Deque OverFlow" << endl;
            return;
        }
        else if (front == -1)
            front = back = 0;
        else if (front == 0 && back != size)
            front = size;
        else
            front--;
        arr[front] = _data;
    }

    void pop_back()
    {
        if (front == -1)
            cout << "Deque is Empty" << endl;
        else if (front == back)
            front = back = -1;
        else if (back == 0)
            back = size;
        else
            back--;
    }

    void pop_front()
    {
        if (front == -1)
            cout << "Deque is Empty" << endl;
        else if (front == back)
            front = back = -1;
        else if (front == size)
            front = 0;
        else
            front++;
    }

    int getFront()
    {
        if (front == -1)
        {
            cout << "Deque if Empty" << endl;
            return -1;
        }
        else
            return arr[front];
    }

    void print()
    {
        if (front == -1)
        {
            cout << "Deque is empty" << endl;
            return;
        }
        else
        {
            int start = front;
            int end = back;
```

```
            while (start != back)
            {
                cout << arr[start] << " ";
                if (start == size)
                    start = 0;
                else
                    start++;
            }
            cout << arr[end] << endl;
        }
    }

    ~deque()
    {
        cout << "Deque Deleted" << endl;
        delete[] arr;
    }
};

int main()
{
    int dequeSize = 10;

    cout << "Size of Queue = 10";

    deque myDeque(dequeSize - 1);

    int choice;
    do
    {
        cout << "\nDeque Menu:\n";
        cout << "1. Display Deque from Front to Back\n";
        cout << "2. Push Front\n";
        cout << "3. Push Back\n";
        cout << "4. Pop Front\n";
        cout << "5. Pop Back\n";
        cout << "6. Exit\n";

        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice)
        {
        case 1:
            myDeque.print();
            break;

        case 3:
            int dataPushBack;
            cout << "Enter data to push back: ";
            cin >> dataPushBack;
            myDeque.push_back(dataPushBack);
            myDeque.print();
            break;
        case 2:
            int dataPushFront;
            cout << "Enter data to push front: ";
            cin >> dataPushFront;
            myDeque.push_front(dataPushFront);
            myDeque.print();
            break;

        case 5:
            myDeque.pop_back();
            myDeque.print();
            break;

        case 4:
            myDeque.pop_front();
            myDeque.print();
            break;

        case 6:
            cout << "Exiting Deque Menu\n";
            break;

        default:
            cout << "Invalid choice. Please enter a valid option.\n";
        }

    } while (choice != 6);

    return 0;
}

/*

Output:

$ g++ Praact13.cpp -o out && ./out
Size of Queue = 10
Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 1
Deque is empty

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
```

```
6. Exit
Enter your choice: 2
Enter data to push front: 123
123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 2
Enter data to push front: 234
234 123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 2
Enter data to push front: 3
3 234 123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 3
Enter data to push back: 456
3 234 123 456

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 4
234 123 456

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
```

```
6. Exit
Enter your choice: 5
234 123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 4
123

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 5
Deque is empty

Deque Menu:
1. Display Deque from Front to Back
2. Push Front
3. Push Back
4. Pop Front
5. Pop Back
6. Exit
Enter your choice: 6
Exiting Deque Menu
Deque Deleted

*/
```