SET - B

Question 1:

1. John plans to buy a laptop. He has shortlisted some of the models based on his requirements.
He has also collected the list of models available in two stores A and B. Given his shortlist and the list of model numbers of the laptops available in A and B; help John to perform the following tasks.

1. List the model number of laptops available in at least one of the stores.
2. List the model number of laptops in John’s shortlist but not available in any of the stores.

Design :

main()

```
1. Read the size of shortlist and number of laptops in  A and B  into n1, n2, n3
2. Read the elements into 3 arrays S,A,B
3. for i = 0 to n2-1   // printing model numbers only in shop A
    a. if ( search (B , A[i] ,  n3) )
        i. continue
    b. else
        i. print( A[i] )  // add spaces while printing
4. for i = 0 to n3-1 // printing model numbers only in shop B
    a. if ( search (A , B[i] , n2) )
        i. continue
    b. else
        i. print( B[i] )  // add spaces while printing
5. for  i = 0 to n2-1  // printing model numbers that are common in both shops A and B
    a. if ( search (B,  A[i]  , n3) )
        i. print( A[i] )  // add spaces while printing
6. flag = 0
7. for i = 0 to n1-1 // model numbers that are in shopping list but not in shops A or B
    a. if ( search (A,  S[i] , n2 ) or  search (B,  S[i]  , n3) )
```

```
            i.  continue
        b. else
            i.  print(S[i])  // add spaces while printing
            ii. flag =1
    8. if(flag == 0)
        a. print( -1 )
```

Evaluation criteria : **[4 marks]**

1 mark for line 3

1 mark for line 4

0.5 marks for line 5

1 mark for line 7

0.5 marks for line 8

search(A, key,n):

//returns true if element is found in array A else returns false

```
1. for  i = 0 to n-1
    a. if(key == A[i])
        i.  return true
2. return false
```

Evaluation criteria : **[2 mark]**

Question 2:

John has collected the prices of the laptops from different stores and prepared a list of prices. To select a laptop, he decided to sort the list in the non-decreasing order of price. He observed that in the list there were sequences of prices in non-decreasing order. He wanted to prepare the final sorted list of prices.

Design:

Question 2

# Declare global variables start,end.

print_prices(A, i, j):

```
1. for k = i to j
    a. print(A[k]) // add spaces while printing
```

Evaluation criteria : **[0.5 mark]**

longest_sorted_sequence(A, n):

```
1. max_length = 1
2. start = 0 // global variable
3. end  = 0  // global variable
4. length = 1
5. index = 0
6. For i=1 to n-1
    a. if ( A[i-1] <= A [i])
        i. length ++
    b. else
        i. if(length >= max_length) // selects the rightmost sequence in the array
            1. max_length = length
            2. start = index
            3. end  = start + max_length-1
        ii. length = 1
        iii. index = i
7. if(length > =max_length)
    a. max_length = length
```

```
        b.  start = index
        c.  end  = start + max_length-1
```

Evaluation criteria : **[1 mark]**

sort_prices(A, n):

// sort function to sort the prices of the given array (students are allowed to use any sorting algorithm)

```
    1.  merge_sort(A , 0 , n-1)
```

merge_sort(A , l , r):

// merge sort for sorting a given array

```
    1.  if l<r
        a.  q = ( l+ r)/2
        b.  merge_sort (A, l, q)
        c.  merge_sort (A, q+1,r)
        d.  merge ( A, l, q, r)
```

merge(A,p,q,r):

// merge procedure for merge sort

```
    1.  $n_1$ = q-p+1
    2.  $n_2$ = r-q
    3.  let  L[0  .....  $n_1$] and R [ 0 .....  $n_2$]  be new arrays
    4.  for i = 0 to $n_1$ -1
        a.  L[i] = A [ p+ i]
    5.  for j = 0 to $n_2$ -1
        a.  R[j] = A[ q + j+1]
    6.  L[$n_1$] = $\infty$
    7.  R[$n_2$] = $\infty$
    8.  i = 0
    9.  j = 0
```

```
10. for k = p to r
        a.  if L [i] $\leq$ R[j]
                i.   A[k] = L[ i]
                ii.  i = i +1
        b.  else
                i.   A[k] = R[j]
                ii.  j = j+1
```

Evaluation criteria : **[1 marks]**

three_way_merge(A, d1, d2, n):

// extending the two way merge function to merge the given 3 arrays

```
1.  $n_1$ = d1 + 1
2.  $n_2$ = d2 - d1
3.  $n_3$ = n- d2 -1
4.  let  L[0.....$n_1$] , M [0.....$n_2$ ]  ,R[0…..$n_3$] be new arrays
5.  for i = 0 to $n_1$  - 1
        a.  L[i] = A [ i ]
6.  for j = 0 to $n_2$ -1
        a.  M[j] =  A[ d1 + j +1]
7.  for k = 0 to $n_3$ - 1
        a.  R[k] = A[ d2 + k +1 ]
8.  L[$n_1$] = $\infty$
9.  M[$n_2$] = $\infty$
10. R[$n_3$] =  $\infty$
11. i = 0
12. j = 0
13. k = 0
14. For l = 0 to n -1
        a.  if L [i] $\leq$ M[j]
        i.   If $L[i] \le R[k]$
                 1. A[l] = L[ i]
                 2. i = i +1
                 3. print (1)
        ii.  else
                 1. A[l] = R[k]
                 2. k = k +1
                 3. print (3)
    b.  else
        i.   if $M[j] \le R[k]$
                 1. A[l] = M[j]
                 2. j = j+1
                 3. print (2)
        ii.  else
                 1. A[l] = R[k]
                 2. k = k+1
                 3. print(3)
```

Evaluation criteria : **[1.5 marks]**

//No marks shall be awarded if the two way merge is used two times to merge the given three arrays