Rajalakshmi Engineering College

Name: Naveed Sheriff
Email: 240701348@rajalakshmi.edu.in
Roll no: 240701348
Phone: 9025573780
Branch: REC
Department: I CSE FD
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_CY_Updated

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Ravi is given an array of integers and is tasked with sorting it uniquely. He needs to sort the elements in such a way that the elements at odd positions are in descending order, and the elements at even positions are in ascending order.

Your task is to help Ravi create a program that uses insertion sort to sort the array as per the specified conditions and then print the sorted array. Position starts from 1.

Example

Input:

Size of the array = 10

Array elements = 25 36 96 58 74 14 35 15 75 95

Output:

Resultant array = 96 14 75 15 74 36 35 58 25 95

Explanation:

Initial Array: 25 36 96 58 74 14 35 15 75 95

Elements at odd positions (1, 3, 5, 7, 9): 25 96 74 35 75

Elements at odd positions sorted descending order: 96 75 74 35 25

Elements at even positions (2, 4, 6, 8, 10): 36 58 14 15 95

Elements at even positions sorted ascending order: 14 15 36 58 95

So, the final array is 96 14 75 15 74 36 35 58 25 95.

Input Format

The first line contains an integer N, representing the number of elements in the array.

The second line contains N space-separated integers, representing the elements of the array.

Output Format

The output displays integers, representing the sorted array elements separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
3 1 4 2
Output: 4 1 3 2

Answer

```
#include<stdio.h>
```

```
#include<stdlib.h>

void insertionsorteven(int a[],int n)
{
   for(int i=1;i<n;i++)
   {
      int temp = a[i];
      int j = i-1;
      while(j>=0 && a[j]>temp)
      {
         a[j+1] = a[j];
         j--;
      }
      a[j+1] = temp;
   }
}
void insertionsortodd(int a[],int n)
{
   for(int i=1;i<n;i++)
   {
      int temp = a[i];
      int j = i-1;
      while(j>=0 && a[j]<temp)
      {
         a[j+1] = a[j];
         j--;
      }
      a[j+1] = temp;
   }
}

int main()
{
   int n;
   scanf("%d",&n);
   int a[n];
   for(int i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
   int odd[100];
   int oddcount = 0;
```

```
    int even[100];
    int evencount = 0;
    for(int i=0;i<n;i++)
    {
        if((i+1)%2 != 0)
        {
            odd[oddcount] = a[i];
            oddcount++;
        }
        else
        {
            even[evencount] = a[i];
            evencount++;
        }
    }
    insertionsortodd(odd,oddcount);
    insertionsorteven(even,evencount);

    int r[n];
    int o = 0,e = 0;
    for(int i=0;i<n;i++)
    {
        if((i+1)%2 != 0)
        r[i] = odd[o++];

        else
        r[i] = even[e++];
    }
    for(int i=0;i<n;i++)
    {
        printf("%d ",r[i]);
    }
}
```

Status : Correct **Marks : 10/10**

2. Problem Statement

Sheela wants to distribute cookies to her children, but each child will only be happy if the cookie size meets or exceeds their individual greed factor. She has a limited number of cookies and wants to make as many children

happy as possible. Priya decides to sort both the greed factors and cookie sizes using QuickSort to efficiently match cookies with children. Your task is to help Sheela determine the maximum number of children that can be made happy.

Input Format

The first line of input consists of an integer n, representing the number of children.

The second line contains n space-separated integers, where each integer represents the greed factor of a child.

The third line contains an integer m, representing the number of cookies.

The fourth line contains m space-separated integers, where each integer represents the size of a cookie.

Output Format

The output prints a single integer, representing the maximum number of children that can be made happy.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1 2 3
2
1 1
Output: The child with greed factor: 1

Answer

```
#include<stdio.h>

int partition(int a[],int lb,int ub)
{
   int pivot = a[lb];
   int start = lb;
   int end = ub;
```

```
    if(start < end)
    {
        while(a[start] <= pivot)
        {
            start++;
        }
        while(a[end] > pivot)
        {
            end--;
        }
        if(start < end)
        {
            int temp = a[start];
            a[start] = a[end];
            a[end] = temp;
        }
        int temp = a[lb];
        a[lb] = a[end];
        a[end] = temp;
    }
    return end;
}

void quicksort(int a[],int lb,int ub)
{
    if(lb < ub)
    {
        int loc = partition(a,lb,ub);
        quicksort(a,lb,loc-1);
        quicksort(a,loc+1,ub);
    }
}
int main()
{
    int n;
    scanf("%d",&n);
    int a[n];
    for(int i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    quicksort(a,0,n-1);
```

```
    int m;
    scanf("%d",&m);
    int b[m];
    for(int i=0;i<m;i++)
    {
        scanf("%d",&b[i]);
    }
    quicksort(b,0,m-1);
    int count = 0;
    int i=0 , j=0;
    while(i<n && j<m)
    {
        if(b[i] >= a[i])
        {
            count++;
            i++;
        }
        j++;
    }
    printf("The child with greed factor: %d",count);
}
```

Status : Correct **Marks : 10/10**

3. Problem Statement

Meera is organizing her art supplies, which are represented as a list of integers: red (0), white (1), and blue (2). She needs to sort these supplies so that all items of the same color are adjacent, in the order red, white, and blue. To achieve this efficiently, Meera decides to use QuickSort to sort the items. Can you help Meera arrange her supplies in the desired order?

Input Format

The first line of input consists of an integer n, representing the number of items in the list.

The second line consists of n space-separated integers, where each integer is either 0 (red), 1 (white), or 2 (blue).

Output Format

The output prints the sorted list of integers in a single line, where integers are arranged in the order red (0), white (1), and blue (2).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
2 0 2 1 1 0
Output: Sorted colors:
0 0 1 1 2 2

Answer

```
#include<stdio.h>

int main()
{
   int n;
   scanf("%d",&n);
   int a[n];
   for(int i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
   for(int i=1;i<n;i++)
   {
      int temp = a[i];
      int j = i-1;
      while(j>=0 && a[j]>temp)
      {
         a[j+1] = a[j];
         j--;
      }
      a[j+1] = temp;
   }
   printf("Sorted colors:\n");
   for(int i=0;i<n;i++)
   {
      printf("%d ",a[i]);
   }
```

```
}
```

Status : Correct ***Marks : 10/10***