Rajalakshmi Engineering College

Name: Soameshwaran D
Email: 240701520@rajalakshmi.edu.in
Roll no: 240701520
Phone: 7358671540
Branch: REC
Department: I CSE FE
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

Reshma is passionate about sorting algorithms and has recently learned about the merge sort algorithm. She wants to implement a program that utilizes the merge sort algorithm to sort an array of integers, both positive and negative, in ascending order.

Help her in implementing the program.

Input Format

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

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

Output Format

The output prints N space-separated integers, representing the array elements sorted in ascending order.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 9
5 -3 0 12 7 -8 2 1 6
Output: -8 -3 0 1 2 5 6 7 12

Answer

```
// You are using GCC
#include <stdio.h>

void merge(int arr[], int left, int mid, int right) {
   int i, j, k;
   int n1 = mid - left + 1;
   int n2 = right - mid;

   // Temporary arrays
   int L[n1], R[n2];

   // Copy data to temporary arrays L[] and R[]
   for (i = 0; i < n1; i++)
      L[i] = arr[left + i];
   for (j = 0; j < n2; j++)
      R[j] = arr[mid + 1 + j];

   // Merge the temporary arrays back into arr[left..right]
   i = 0; // Initial index of L[]
   j = 0; // Initial index of R[]
   k = left; // Initial index of merged subarray

   while (i < n1 && j < n2) {
      if (L[i] <= R[j]) {
         arr[k++] = L[i++];
      } else {
            arr[k++] = R[j++];
        }
    }

    // Copy the remaining elements of L[], if any
    while (i < n1) {
        arr[k++] = L[i++];
    }

    // Copy the remaining elements of R[], if any
    while (j < n2) {
        arr[k++] = R[j++];
    }
}

void mergeSort(int arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;

        // Sort first and second halves
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);

        // Merge the sorted halves
        merge(arr, left, mid, right);
    }
}

int main() {
    int N;
    scanf("%d", &N);

    int arr[25];
    for (int i = 0; i < N; i++) {
        scanf("%d", &arr[i]);
    }

    mergeSort(arr, 0, N - 1);

    for (int i = 0; i < N; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

Status : Correct **_Marks : 10/10_**

2. 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

```
// You are using GCC
#include <stdio.h>

// Function to perform insertion sort on specific indices
void insertion_sort(int arr[], int indices[], int count, int ascending) {
    for (int i = 1; i < count; i++) {
        int key = arr[indices[i]];
        int j = i - 1;

        while (j >= 0 && ((ascending && arr[indices[j]] > key) || (!ascending &&
arr[indices[j]] < key))) {
            arr[indices[j + 1]] = arr[indices[j]];
            j--;
        }
        arr[indices[j + 1]] = key;
    }
}

int main() {
    int n;
    scanf("%d", &n);

    int arr[10]; // Since 1 ≤ N ≤ 10
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Arrays to store indices of odd and even positions (1-based)
    int odd_indices[10], even_indices[10];
    int odd_count = 0, even_count = 0;

    // Separate indices based on position
    for (int i = 0; i < n; i++) {
        if ((i + 1) % 2 == 1) { // 1-based odd position
            odd_indices[odd_count++] = i;
        } else { // 1-based even position
            even_indices[even_count++] = i;
        }
    }

    // Sort odd-position elements in descending order
    insertion_sort(arr, odd_indices, odd_count, 0);

    // Sort even-position elements in ascending order
    insertion_sort(arr, even_indices, even_count, 1);

    // Print the sorted array
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```

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

3. Problem Statement

Marie, the teacher, wants her students to implement the ascending order of numbers while also exploring the concept of prime numbers.

Students need to write a program that sorts an array of integers using the merge sort algorithm while counting and returning the number of prime integers in the array. Help them to complete the program.

Input Format

The first line of input consists of an integer N, representing the number of array elements.

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

Output Format

The first line of output prints the sorted array of integers in ascending order.

The second line prints the number of prime integers in the array.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
5 3 6 8 9 7 4

Output: Sorted array: 3 4 5 6 7 8 9
Number of prime integers: 3

Answer

```
// You are using GCC
#include <stdio.h>
#include <stdbool.h>

// Function to check if a number is prime
bool isPrime(int num) {
   if (num < 2) return false;
   for (int i = 2; i * i <= num; i++) {
      if (num % i == 0)
         return false;
   }
   return true;
```

```
}

// Merge function
void merge(int arr[], int left, int mid, int right) {
   int i, j, k;
   int n1 = mid - left + 1;
   int n2 = right - mid;

   // Temporary arrays
   int L[10], R[10];

   for (i = 0; i < n1; i++)
      L[i] = arr[left + i];
   for (j = 0; j < n2; j++)
      R[j] = arr[mid + 1 + j];

   // Merge back
   i = 0; j = 0; k = left;

   while (i < n1 && j < n2) {
      if (L[i] <= R[j]) arr[k++] = L[i++];
      else arr[k++] = R[j++];
   }

   while (i < n1) arr[k++] = L[i++];
   while (j < n2) arr[k++] = R[j++];
}

// Merge Sort function
void mergeSort(int arr[], int left, int right) {
   if (left < right) {
      int mid = left + (right - left) / 2;

      mergeSort(arr, left, mid);
      mergeSort(arr, mid + 1, right);

      merge(arr, left, mid, right);
   }
}

int main() {
   int N, i, count = 0;
```

```
    int arr[10];

    // Input
    scanf("%d", &N);
    for (i = 0; i < N; i++) {
        scanf("%d", &arr[i]);
    }

    // Sort the array
    mergeSort(arr, 0, N - 1);

    // Count primes
    for (i = 0; i < N; i++) {
        if (isPrime(arr[i])) {
            count++;
        }
    }

    // Output
    printf("Sorted array: ");
    for (i = 0; i < N; i++) {
        printf("%d ", arr[i]);
    }
    printf("\nNumber of prime integers: %d\n", count);

    return 0;
}
```

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