Rajalakshmi Engineering College

Name: pooja D
Email: 240701385@rajalakshmi.edu.in
Roll no: 240701385
Phone: 9677250159
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

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 l, int m, int r) {
    int n1 = m - l + 1, n2 = r - m;
    int left[n1], right[n2];
    for (int i = 0; i < n1; i++) left[i] = arr[l + i];
    for (int j = 0; j < n2; j++) right[j] = arr[m + 1 + j];
    int i = 0, j = 0, k = l;
    while (i < n1 && j < n2) {
        if (left[i] <= right[j]) arr[k++] = left[i++];
        else arr[k++] = right[j++];
    }
    while (i < n1) arr[k++] = left[i++];
    while (j < n2) arr[k++] = right[j++];
}

void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = l + (r - l) / 2;
        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);
        merge(arr, l, m, r);
    }
}
```

```
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

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

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

void swap(int *a, int *b) {
    int t = *a;
    *a = *b;
    *b = t;
}

int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = low - 1;
    for (int j = low; j < high; j++) {
        if (arr[j] <= pivot) {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return i + 1;
}

void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
```

```
int maxChildrenHappy(int greed[], int n, int cookies[], int m) {
    int i = 0, j = 0, count = 0;
    while (i < n && j < m) {
        if (cookies[j] >= greed[i]) {
            count++;
            i++;
            j++;
        } else {
            j++;
        }
    }
    return count;
}

int main() {
    int n, m;
    scanf("%d", &n);
    int greed[100];
    for (int i = 0; i < n; i++) scanf("%d", &greed[i]);

    scanf("%d", &m);
    int cookies[100];
    for (int i = 0; i < m; i++) scanf("%d", &cookies[i]);

    quickSort(greed, 0, n - 1);
    quickSort(cookies, 0, m - 1);

    int result = maxChildrenHappy(greed, n, cookies, m);
    printf("The child with greed factor: %d\n", result);

    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>

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;
}

void merge(int arr[], int l, int m, int r) {
    int n1 = m - l + 1, n2 = r - m;
```

```
    int left[n1], right[n2];
    for (int i=0; i<n1; i++) left[i] = arr[l+i];
    for (int i=0; i<n2; i++) right[i] = arr[m+1+i];
    int i=0, j=0, k=l;
    while(i<n1 && j<n2) {
        if (left[i] <= right[j]) arr[k++] = left[i++];
        else arr[k++] = right[j++];
    }
    while(i<n1) arr[k++] = left[i++];
    while(j<n2) arr[k++] = right[j++];
}

void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = l + (r-l)/2;
        mergeSort(arr, l, m);
        mergeSort(arr, m+1, r);
        merge(arr, l, m, r);
    }
}

int main() {
    int N;
    scanf("%d", &N);
    int arr[10];
    for (int i=0; i<N; i++) scanf("%d", &arr[i]);
    mergeSort(arr, 0, N-1);
    int primeCount = 0;
    for (int i=0; i<N; i++) if (isPrime(arr[i])) primeCount++;
    printf("Sorted array: ");
    for (int i=0; i<N; i++) printf("%d ", arr[i]);
    printf("\nNumber of prime integers: %d\n", primeCount);
    return 0;
}
```

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