

COMPETITIVE PROGRAMMING

ASSIGNMENT-2

Name: V. Vyshnavi

Hallticket No.:2303A51968

Batch-14

Problem Statement :

A warehouse records package priority scores as integers. To dispatch efficiently, you must sort the scores in non-decreasing order using merge sort (divide and conquer). For each test case, output the sorted list.

Input Format:

The first line contains integer T.

For each test case:

- First line: N
- Second line: N integers (priority scores)

Output Format:

For each test case, print the sorted array in one line (space-separated).

Constraints:

- $1 \leq T \leq 20$
- $1 \leq N \leq 200000$ (sum of N over all test cases ≤ 200000)
- $-10^9 \leq A_i \leq 10^9$

Sample Input

```
1
7
4 1 6 2 5 3 2
```

Expected Output

```
1 2 2 3 4 5 6
```

Python Code:

```
def merge_sort(arr):
    if len(arr) <= 1:
```

```
    return arr
```

```
    mid = len(arr) // 2
```

```
    left = merge_sort(arr[:mid])
```

```
    right = merge_sort(arr[mid:])
```

```
    return merge(left, right)
```

```
def merge(left, right):
```

```
    result = []
```

```
    i = j = 0
```

```
    while i < len(left) and j < len(right):
```

```
        if left[i] <= right[j]:
```

```
            result.append(left[i])
```

```
            i += 1
```

```
        else:
```

```
            result.append(right[j])
```

```
            j += 1
```

```
    result.extend(left[i:])
```

```
    result.extend(right[j:])
```

```
    return result
```

```
T = int(input())
```

```
for _ in range(T):
```

```
    N = int(input())
```

```
    arr = list(map(int, input().split()))
```

```
sorted_arr = merge_sort(arr)
```

```
print(*sorted_arr)
```



The screenshot shows the OnlineGDB interface with a Python 3 file named 'main.py'. The code defines a recursive merge sort function. The left sidebar contains navigation links like 'Create New Project', 'My Projects', and 'Classroom'. The top toolbar includes buttons for 'Run', 'Debug', 'Stop', 'Share', 'Save', and 'Beautify'.

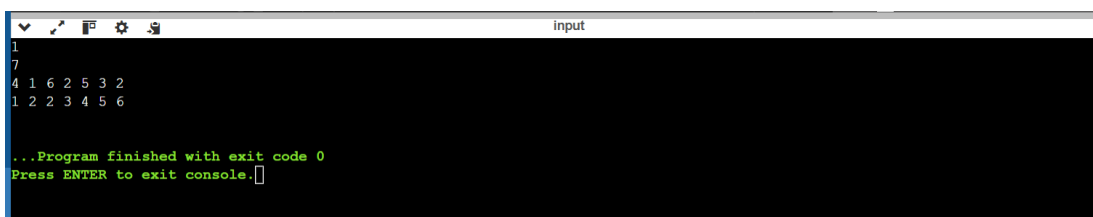
```
1 def merge_sort(arr):
2     if len(arr) <= 1:
3         return arr
4
5     mid = len(arr) // 2
6     left = merge_sort(arr[:mid])
7     right = merge_sort(arr[mid:])
8
9     return merge(left, right)
10
11
12 def merge(left, right):
13     result = []
14     i = j = 0
15
16     while i < len(left) and j < len(right):
17         if left[i] <= right[j]:
18             result.append(left[i])
19             i += 1
20         else:
21             result.append(right[j])
22             j += 1
23
24     result.extend(left[i:])
25     result.extend(right[j:])
26     return result
27
```



This screenshot shows the continuation of the 'main.py' file in the OnlineGDB editor. It includes the main logic to read input, convert it to a list, and call the merge sort function. The output area on the right is currently empty.

```
12 def merge(left, right):
13     result = []
14     i = j = 0
15
16     while i < len(left) and j < len(right):
17         if left[i] <= right[j]:
18             result.append(left[i])
19             i += 1
20         else:
21             result.append(right[j])
22             j += 1
23
24     result.extend(left[i:])
25     result.extend(right[j:])
26     return result
27
28
29 T = int(input())
30
31 for _ in range(T):
32     N = int(input())
33     arr = list(map(int, input().split()))
34
35     sorted_arr = merge_sort(arr)
36     print(*sorted_arr)
37
```

Output:



The screenshot shows the console output of the program. It displays two test cases: the first with input [4, 1, 6, 2, 5, 3, 2] and output [1, 2, 2, 3, 4, 5, 6], and the second with input [1, 2, 2, 3, 4, 5, 6] and output [1, 2, 2, 3, 4, 5, 6]. The program finished with exit code 0.

```
1
7
4 1 6 2 5 3 2
1 2 2 3 4 5 6

...Program finished with exit code 0
Press ENTER to exit console.
```

Java Code:

```
import java.util.*;

public class Main {

    static void mergeSort(long[] a, int l, int r) {
        if (l >= r) return;

        int m = (l + r) / 2;
        mergeSort(a, l, m);
        mergeSort(a, m + 1, r);
        merge(a, l, m, r);
    }

    static void merge(long[] a, int l, int m, int r) {
        long[] t = new long[r - l + 1];
        int i = l, j = m + 1, k = 0;

        while (i <= m && j <= r)
            t[k++] = (a[i] <= a[j]) ? a[i++] : a[j++];

        while (i <= m) t[k++] = a[i++];
        while (j <= r) t[k++] = a[j++];

        for (i = l, k = 0; i <= r; i++, k++)
            a[i] = t[k];
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int T = sc.nextInt();

        while (T-- > 0) {
```

```

int N = sc.nextInt();

long[] a = new long[N];

for (int i = 0; i < N; i++)

    a[i] = sc.nextLong();

mergeSort(a, 0, N - 1);

for (int i = 0; i < N; i++)

    System.out.print(a[i] + " ");

System.out.println();
}

sc.close();
}
}

```

The screenshot shows the OnlineGDB web interface. On the left is a sidebar with navigation links: 'Create New Project', 'My Projects', 'Classroom' (marked as 'new'), 'Learn Programming', 'Programming Questions', 'Upgrade', and 'Logout'. The main area displays a Java code editor with the following code:

```

1 ~ import java.util.*;
2
3 ~ public class Main {
4
5 ~     static void mergeSort(long[] a, int l, int r) {
6         if (l >= r) return;
7
8         int m = (l + r) / 2;
9         mergeSort(a, l, m);
10        mergeSort(a, m + 1, r);
11        merge(a, l, m, r);
12    }
13
14 ~    static void merge(long[] a, int l, int m, int r) {
15        long[] t = new long[r - l + 1];
16        int i = l, j = m + 1, k = 0;
17
18        while (i <= m && j <= r)
19            t[k++] = (a[i] <= a[j]) ? a[i++] : a[j++];
20
21        while (i <= m) t[k++] = a[i++];
22        while (j <= r) t[k++] = a[j++];
23
24        for (i = l, k = 0; i <= r; i++, k++)
25            a[i] = t[k];
26    }
27 }

```

The interface includes a top toolbar with buttons for Run, Debug, Stop, Share, Save, and Beautify. The language is set to Java. A 'close ad' button is visible in the top right corner of the code editor area.



```
OnlineGDB
online compiler and debugger for c/c++

Welcome, 2303A51968

Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Upgrade
Logout

Main.java
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

for (int i = 0; i < N; i++)
    a[i] = t[k];

}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int T = sc.nextInt();

    while (T-- > 0) {
        int N = sc.nextInt();
        long[] a = new long[N];

        for (int i = 0; i < N; i++)
            a[i] = sc.nextLong();

        mergeSort(a, 0, N - 1);

        for (int i = 0; i < N; i++)
            System.out.print(a[i] + " ");
        System.out.println();
    }
    sc.close();
}
```

Output:



```
input
1
7
4 1 6 2 5 3 2
1 2 2 3 4 5 6

...Program finished with exit code 0
Press ENTER to exit console.
```

Maximum Profit Streak (Divide and Conquer)

Problem Statement:

A company tracks daily profit changes as an array A (values can be negative). A “profit streak” is any non-empty contiguous subarray. Find the maximum possible sum of a profit streak using a divide-and-conquer approach (split into left, right, and crossing subproblems).

Input Format:

The first line contains integer T. For each test case:

- First line: N
- Second line: N integers A1.

AN Output Format: For each test case, print one integer: maximum subarray sum.

Constraints

- $1 \leq T \leq 20$
- $1 \leq N \leq 200000$ (sum of N over all test cases ≤ 200000)
- $-10^9 \leq A_i \leq 10^9$

Sample Input

```
1
9
```

-2 1 -3 4 -1 2 1 -5 4

Expected Output : 6

Python code:

```
def max_sum(arr, l, r):  
    if l == r: return arr[l]  
    m = (l + r) // 2  
    left = max_sum(arr, l, m)  
    right = max_sum(arr, m+1, r)  
    cross = sum_ = 0  
    c = -10**18  
    for i in range(m, l-1, -1):  
        sum_ += arr[i]  
        c = max(c, sum_)  
    sum_ = 0  
    d = -10**18  
    for i in range(m+1, r+1):  
        sum_ += arr[i]  
        d = max(d, sum_)  
    return max(left, right, c+d)
```

```
T = int(input())  
for _ in range(T):  
    N = int(input())  
    arr = list(map(int, input().split()))  
    print(max_sum(arr, 0, N-1))
```



OnlineGDB
online compiler and debugger for c/c++

Welcome, 2303A51968 ▲

Create New Project

My Projects

Classroom **new**

Learn Programming

Programming Questions

Upgrade

Logout ▾

main.py

```
1 def max_sum(arr, l, r):
2     if l == r: return arr[l]
3     m = (l + r) // 2
4     left = max_sum(arr, l, m)
5     right = max_sum(arr, m+1, r)
6     cross = sum_ = 0
7     c = -10**18
8     for i in range(m, l-1, -1):
9         sum_ += arr[i]
10        c = max(c, sum_)
11    sum_ = 0
12    d = -10**18
13    for i in range(m+1, r+1):
14        sum_ += arr[i]
15        d = max(d, sum_)
16    return max(left, right, c+d)
17
18 T = int(input())
19 for _ in range(T):
20     N = int(input())
21     arr = list(map(int, input().split()))
22     print(max_sum(arr, 0, N-1))
23
```

close ad [x]

Output:



```
1
9
-2 1 -3 4 -1 2 1 -5 4
6

...Program finished with exit code 0
Press ENTER to exit console.
```

Java Code:

```
import java.util.*;

public class Main {

    static long maxSum(long[] a, int l, int r) {

        if(l==r) return a[l];

        int m=(l+r)/2;

        long left=maxSum(a,l,m), right=maxSum(a,m+1,r);

        long sum=0,c=-Long.MAX_VALUE;

        for(int i=m;i>=l;i--){sum+=a[i]; c=Math.max(c,sum);}

        sum=0; long d=-Long.MAX_VALUE;

        for(int i=m+1;i<=r;i++){sum+=a[i]; d=Math.max(d,sum);}

        return Math.max(Math.max(left,right),c+d);

    }

    public static void main(String[] args){

        Scanner sc=new Scanner(System.in);

        int T=sc.nextInt();
```



```

while(T-->0){

    int N=sc.nextInt();

    long[] a=new long[N];

    for(int i=0;i<N;i++) a[i]=sc.nextLong();

    System.out.println(maxSum(a,0,N-1));

}

sc.close();

}

}

```



The screenshot shows the OnlineGDB IDE interface. On the left is a sidebar with the user's name '2303A51968' and various navigation links. The main area displays a Java file named 'Main.java' with the following code:

```

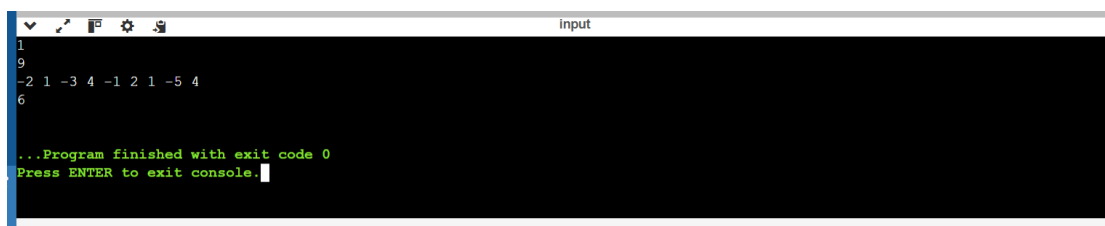
import java.util.*;

public class Main {
    static long maxSum(long[] a, int l, int r) {
        if(l==r) return a[l];
        int m=(l+r)/2;
        long left=maxSum(a,l,m), right=maxSum(a,m+1,r);
        long sum=0,c=-Long.MAX_VALUE;
        for(int i=m;i>=l;i--){sum+=a[i]; c=Math.max(c,sum);}
        sum=0; long d=-Long.MAX_VALUE;
        for(int i=m+1;i<=r;i++){sum+=a[i]; d=Math.max(d,sum);}
        return Math.max(Math.max(left,right),c+d);
    }

    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        int T=sc.nextInt();
        while(T-->0){
            int N=sc.nextInt();
            long[] a=new long[N];
            for(int i=0;i<N;i++) a[i]=sc.nextLong();
            System.out.println(maxSum(a,0,N-1));
        }
        sc.close();
    }
}

```

Output:



The screenshot shows the console output of the program. The input is as follows:

```

1
9
-2 1 -3 4 -1 2 1 -5 4
6

```

The output of the program is:

```

...Program finished with exit code 0
Press ENTER to exit console.

```