DAA LAB

Name: Atharva Jogiji

Section: A5_B2

Roll no: 34

Practical 4: TASK-1

Code:

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

int max(int a, int b)
{
   if(a > b)
      return a;
   else
      return b;
}
int recursiveSplitWithConstraint(int arr[], int n, int constraint)
{
   if(n == 0)
   {
      return 0;
   }
   else if(n == 1)
   {
      if(arr[0] <= constraint)
      {
         return arr[0];
      }
      else
      {
         return 0;
      }
   }
   int mid = n/2;
   int leftSum = recursiveSplitWithConstraint(arr, mid, constraint);
   int rightSum = recursiveSplitWithConstraint(arr + mid, n - mid, constraint);
   int leftCross = INT_MIN;
   int rightCross = INT_MIN;
   int currentSum = 0;
```

```
    for(int i=mid-1; i>=0; i--)
    {
        currentSum += arr[i];
        if(currentSum <= constraint && currentSum > leftCross)
        {
            leftCross=currentSum;
        }
    }
    currentSum = 0;
    for(int i=mid; i<n; i++)
    {
        currentSum += arr[i];
        if(currentSum <= constraint && currentSum > rightCross)
        {
            rightCross=currentSum;
        }
    }
    int crossSum=0;
    if(leftCross!=INT_MIN && rightCross!=INT_MIN && leftCross + rightCross <= constraint)
    {
        crossSum=leftCross+rightCross;
    }
    int best=max(leftSum, rightSum);
    best=max(best, crossSum);
    return best;
}

int main()
{
    int case1[]={2, 1, 3, 4};
    int case2[]={2, 2, 2, 2};
    int case3[]={1, 5, 2, 3};
    int case4[]={6, 7, 8};
    int case5[]={1, 2, 3, 2, 1};
    int case6[]={1, 1, 1, 1, 1};
    int case7[]={4, 2, 3, 1};
    int case8[]={};
    int case9[]={1, 2, 3};

    printf("Test Case 1: %d\n", recursiveSplitWithConstraint(case1, 4, 5));
    printf("Test Case 2: %d\n", recursiveSplitWithConstraint(case2, 4, 4));
    printf("Test Case 3: %d\n", recursiveSplitWithConstraint(case3, 4, 5));
    printf("Test Case 4: %d\n", recursiveSplitWithConstraint(case4, 3, 5));
    printf("Test Case 5: %d\n", recursiveSplitWithConstraint(case5, 5, 5));
    printf("Test Case 6: %d\n", recursiveSplitWithConstraint(case6, 5, 4));
    printf("Test Case 7: %d\n", recursiveSplitWithConstraint(case7, 4, 5));
    printf("Test Case 8: %d\n", recursiveSplitWithConstraint(case8, 0, 10));
```

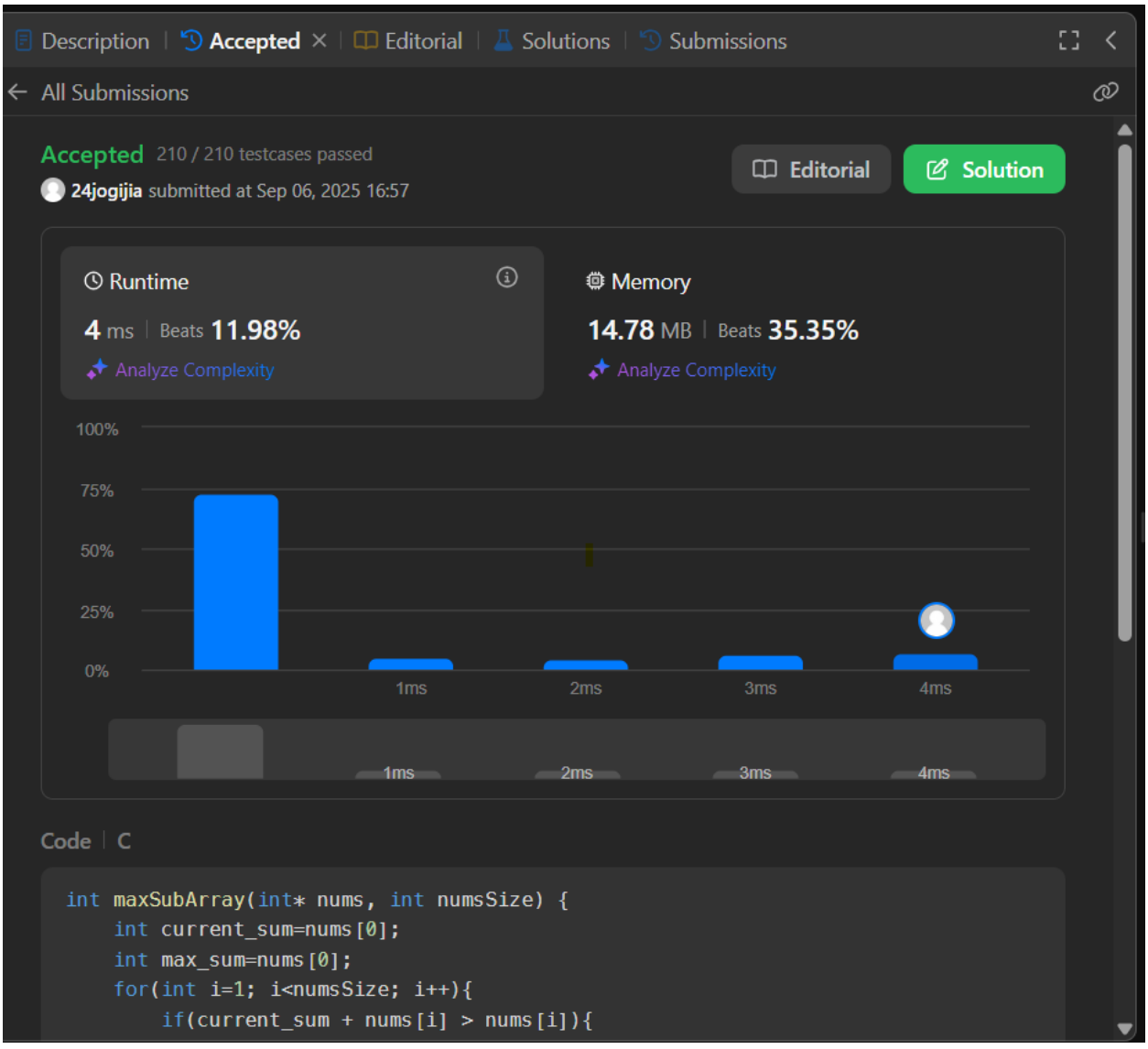
```
    printf("Test Case 9: %d\n", recursiveSplitWithConstraint(case9, 3, 0));

    int n=100000;
    int Long[n];
    for(int i=0; i<n; i++)
    {
        Long[i]=i+1;
    }
    printf("Test Case 10: %d\n", recursiveSplitWithConstraint(Long, n, 1000000000));
    return 0;
}
```

Output:

```
PS C:\Users\Dell> cd "c:\Users\Dell\Downloads\" ; if ($?) { gcc p4.c -o p4 } ; if ($?) { .\p4 }
Test Case 1: 4
Test Case 2: 4
Test Case 3: 5
Test Case 4: 0
Test Case 5: 3
Test Case 6: 3
Test Case 7: 4
Test Case 8: 0
Test Case 9: 0
Test Case 10: 937512500
PS C:\Users\Dell\Downloads>
```

TASK-2

Leetcode:


Description | Accepted × | Editorial | Solutions | Submissions
All Submissions
Accepted 210 / 210 testcases passed
24jogijia submitted at Sep 06, 2025 16:57
Editorial
Solution
Runtime
4 ms | Beats 11.98%
Analyze Complexity
Memory
14.78 MB | Beats 35.35%
Analyze Complexity
100%
75%
50%
25%
0%
1ms
2ms
3ms
4ms
Code | C
int maxSubArray(int* nums, int numsSize) {
int current_sum=nums[0];
int max_sum=nums[0];
for(int i=1; i<numsSize; i++){
if(current_sum + nums[i] > nums[i]){