CSCI 445 Max Subarray Problem: Design Document

Lucas Furlong

Davenport University

CSCI 445 Max Subarray Problem: Design Document

**Overview**

The problem that was asked was to take the max subarray problem, being that you must find a subarray from an array that contains the maximum sum and utilize the divide and conquer algorithm to solve it.

The divide and conquer algorithm divides the array into two halves and then returns the maximum of the subarray sum on the right and left half and the subarray sum when the subarray crosses the midpoint. It utilizes recursion to find the sum for the left and right half.

This program will take an input from a user of an array, which can consist of various sizes ranging from 1 to 2,147,483,647 (the max value for int32 in C#) and find the max subarray in the main array. This document will go over the specific inputs, outputs, the components and properties of the code, the processing, and the testing of the program.

**Logic (Components and Properties)**

|  |
| --- |
| Program |
| **-**leftsum : double  -rightsum: double  -sum : double  -s : string  -size : int  -c : string  -numbers : double  -maxsum : double |
| **+**MaxCrossingSubarray( a[] : double, l : int, m : int, r : int) : double  +MaxSubarray(a[] : double, l : int, r : int) : double |

**Properties in the Program**

***leftsum***

This property represents the max left sum which is then assigned to the sum when calculating in the MaxCrossingSubarray method.

***rightsum***

This property represents the max right sum which is then assigned to the sum when calculating in the MaxCrossingSubarray method.

***sum***

This property represents the value that is returned to the MaxSubarray method when calculation is complete in the MaxCrossingSubarray method.

***s***

This property represents the string value that is obtained from the user for the size of the array.

***size***

This property represents the integer value of the size that is used to initialize the array when it is created.

***c***

This property represents the string value of each number that is entered into the program by the user.

***numbers***

This property represents the double value of each number that is used when initializing the array after it is created.

***maxsum***

This property represents the max sum of the subarray that is found.

**Methods in the Program**

***MaxSubarray***

This method halves the array until there is the ability to calculate a max subarray that can be compared to the other parts of the array that are subsequentially calculated.

***MaxCrossingSubarray***

This is the method that does the calculating of the Subarrays which then is passed back to be compared again in the MaxSubarray method.

**Input**

This section with cover the inputs that are permitted by the user to be processed in the program

**Two Sections (Size and Numbers)**

The program will ask two questions when the user starts it. The first question that is asked is what the size of the array is. The user can enter a number from 1 to the max value of Int32 value type. The second question is what the numbers are of the array, and the user can enter any double type ranging from the min to the max value of the double value type.

The point of this is to establish what the program will take as an array so it can calculate as the max subarray.

Text

Description automatically generated

A picture containing background pattern

Description automatically generated

**Output**

The output of the program is a string that writes out the max sum of the subarray.

A computer screen capture

Description automatically generated with medium confidence

**Testing**

**Scenario 1 (array of 6 elements)**

Array is: [9, -2, 6, -7, 1, 4]

Text

Description automatically generated

Max subarray is [9, -2, 6] which sum is 13

**Scenario 2 (array of 5 elements with decimals)**

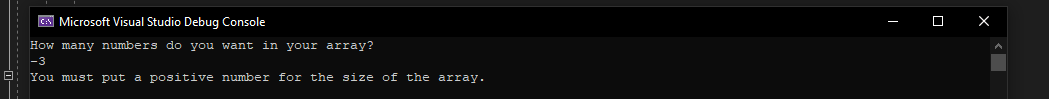
Array is [1.2, -4.2, 9.3, -2.55, 3.2]

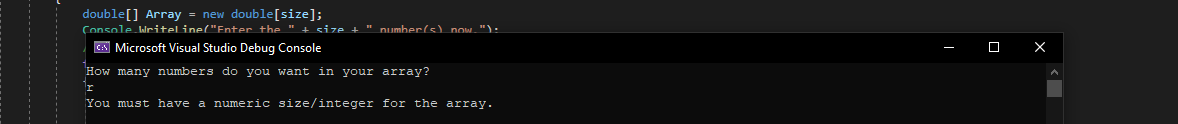
A picture containing background pattern

Description automatically generated

Max subarray is [9.3, -2.55, 3.2] which sum is 9.95

**Scenario 3 (bad cases)**





Background pattern

Description automatically generated