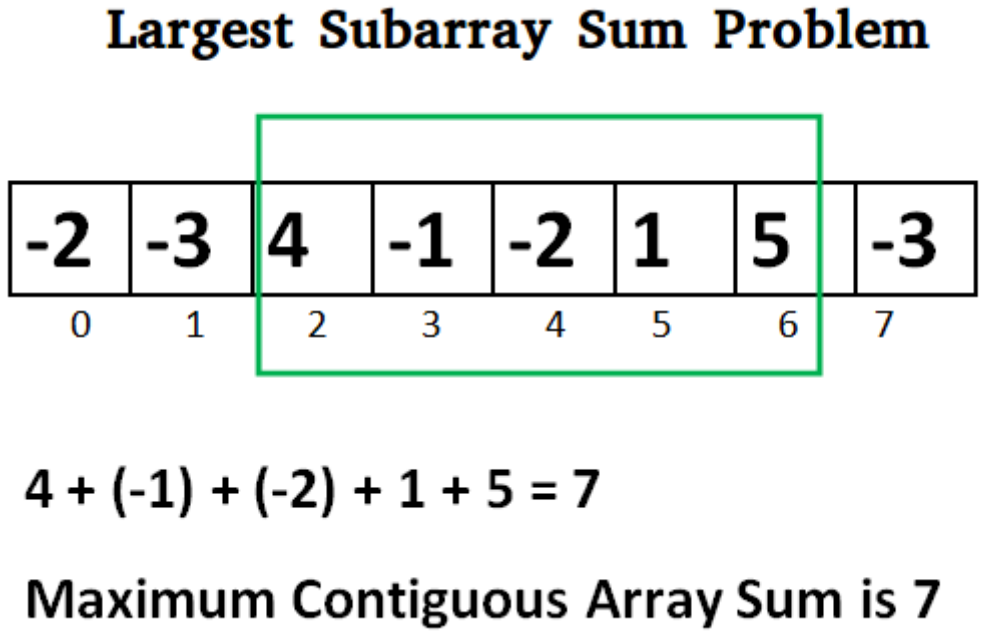
Largest Sum Contiguous Subarray

Write an efficient C program to find the sum of contiguous subarray within a one-dimensional array of numbers which has the largest sum.



Recommended: Please solve it on "*PRACTICE* " first, before moving on to the solution.

Kadane's Algorithm:

```
Initialize:
    max_so_far = 0
    max_ending_here = 0

Loop for each element of the array
  (a) max_ending_here = max_ending_here + a[i]
  (b) if(max_ending_here < 0)
            max_ending_here = 0
  (c) if(max_so_far < max_ending_here)
```

```
            max_so_far = max_ending_here
return max_so_far
```

Explanation:

Simple idea of the Kadane's algorithm is to look for all positive contiguous segments of the array (max_ending_here is used for this). And keep track of maximum sum contiguous segment among all positive segments (max_so_far is used for this). Each time we get a positive sum compare it with max_so_far and update max_so_far if it is greater than max_so_far

```
    Lets take the example:
    {-2, -3, 4, -1, -2, 1, 5, -3}

    max_so_far = max_ending_here = 0

    for i=0,  a[0] =  -2
    max_ending_here = max_ending_here + (-2)
    Set max_ending_here = 0 because max_ending_here < 0

    for i=1,  a[1] =  -3
    max_ending_here = max_ending_here + (-3)
    Set max_ending_here = 0 because max_ending_here < 0

    for i=2,  a[2] =  4
    max_ending_here = max_ending_here + (4)
    max_ending_here = 4
    max_so_far is updated to 4 because max_ending_here greater
    than max_so_far which was 0 till now

    for i=3,  a[3] =  -1
    max_ending_here = max_ending_here + (-1)
    max_ending_here = 3

    for i=4,  a[4] =  -2
    max_ending_here = max_ending_here + (-2)
    max_ending_here = 1

    for i=5,  a[5] =  1
    max_ending_here = max_ending_here + (1)
    max_ending_here = 2

    for i=6,  a[6] =  5
    max_ending_here = max_ending_here + (5)
    max_ending_here = 7
    max_so_far is updated to 7 because max_ending_here is
    greater than max_so_far

    for i=7,  a[7] =  -3
    max_ending_here = max_ending_here + (-3)
    max_ending_here = 4
```

Program:

C++

```cpp
// C++ program to print largest contiguous array sum
#include<iostream>
#include<climits>
using namespace std;

int maxSubArraySum(int a[], int size)
{
    int max_so_far = INT_MIN, max_ending_here = 0;

    for (int i = 0; i < size; i++)
    {
        max_ending_here = max_ending_here + a[i];
        if (max_so_far < max_ending_here)
            max_so_far = max_ending_here;

        if (max_ending_here < 0)
            max_ending_here = 0;
    }
    return max_so_far;
}

/*Driver program to test maxSubArraySum*/
int main()
{
    int a[] = {-2, -3, 4, -1, -2, 1, 5, -3};
    int n = sizeof(a)/sizeof(a[0]);
    int max_sum = maxSubArraySum(a, n);
    cout << "Maximum contiguous sum is " << max_sum;
    return 0;
}
```

Run on IDE

Java

```java
import java.io.*;
// Java program to print largest contiguous array sum
import java.util.*;

class Kadane
{
    public static void main (String[] args)
    {
        int [] a = {-2, -3, 4, -1, -2, 1, 5, -3};
        System.out.println("Maximum contiguous sum is " +
                                        maxSubArraySum(a));
    }

    static int maxSubArraySum(int a[])
    {
        int size = a.length;
        int max_so_far = Integer.MIN_VALUE, max_ending_here = 0;

        for (int i = 0; i < size; i++)
        {
            max_ending_here = max_ending_here + a[i];
            if (max_so_far < max_ending_here)
                max_so_far = max_ending_here;
            if (max_ending_here < 0)
                max_ending_here = 0;
        }
        return max_so_far;
    }
}
```

Run on IDE

Python

```
# Python program to find maximum contiguous subarray

# Function to find the maximum contiguous subarray
from sys import maxint
def maxSubArraySum(a,size):

    max_so_far = -maxint - 1
    max_ending_here = 0

    for i in range(0, size):
        max_ending_here = max_ending_here + a[i]
        if (max_so_far < max_ending_here):
            max_so_far = max_ending_here

        if max_ending_here < 0:
            max_ending_here = 0
    return max_so_far

# Driver function to check the above function
a = [-13, -3, -25, -20, -3, -16, -23, -12, -5, -22, -15, -4, -7]
print "Maximum contiguous sum is", maxSubArraySum(a,len(a))

#This code is contributed by _Devesh Agrawal_
```

Run on IDE

Output:

```
Maximum contiguous sum is 7
```

Above program can be optimized further, if we compare max_so_far with max_ending_here only if max_ending_here is greater than 0.

C++

```
int maxSubArraySum(int a[], int size)
{
   int max_so_far = 0, max_ending_here = 0;
   for (int i = 0; i < size; i++)
   {
       max_ending_here = max_ending_here + a[i];
       if (max_ending_here < 0)
           max_ending_here = 0;

       /* Do not compare for all elements. Compare only
          when  max_ending_here > 0 */
       else if (max_so_far < max_ending_here)
           max_so_far = max_ending_here;
   }
   return max_so_far;
}
```

Run on IDE

Python

```
def maxSubArraySum(a,size):

    max_so_far = 0
    max_ending_here = 0

    for i in range(0, size):
        max_ending_here = max_ending_here + a[i]
```

```
        if max_ending_here < 0:
            max_ending_here = 0

        # Do not compare for all elements. Compare only
        # when  max_ending_here > 0
        elif (max_so_far < max_ending_here):
            max_so_far = max_ending_here

    return max_so_far
```

Run on IDE

Time Complexity: O(n)

Algorithmic Paradigm: Dynamic Programming

Following is another simple implementation suggested by **Mohit Kumar**. The implementation handles the case when all numbers in array are negative.

C++

```
#include<iostream>
using namespace std;

int maxSubArraySum(int a[], int size)
{
   int max_so_far = a[0];
   int curr_max = a[0];

   for (int i = 1; i < size; i++)
   {
        curr_max = max(a[i], curr_max+a[i]);
        max_so_far = max(max_so_far, curr_max);
   }
   return max_so_far;
}

/* Driver program to test maxSubArraySum */
int main()
{
   int a[] =  {-2, -3, 4, -1, -2, 1, 5, -3};
   int n = sizeof(a)/sizeof(a[0]);
   int max_sum = maxSubArraySum(a, n);
   cout << "Maximum contiguous sum is " << max_sum;
   return 0;
}
```

Run on IDE

Java

```
// Java program to print largest contiguous
// array sum
import java.io.*;

class GFG {

    static int maxSubArraySum(int a[], int size)
    {
    int max_so_far = a[0];
    int curr_max = a[0];

    for (int i = 1; i < size; i++)
    {
           curr_max = Math.max(a[i], curr_max+a[i]);
```

```
        max_so_far = Math.max(max_so_far, curr_max);
    }
    return max_so_far;
    }

    /* Driver program to test maxSubArraySum */
    public static void main(String[] args)
    {
    int a[] = {-2, -3, 4, -1, -2, 1, 5, -3};
    int n = a.length;
    int max_sum = maxSubArraySum(a, n);
    System.out.println("Maximum contiguous sum is "
                       + max_sum);
    }
}

// This code is contributd by Prerna Saini
```

Run on IDE

Python

```
# Python program to find maximum contiguous subarray

def maxSubArraySum(a,size):

    max_so_far =a[0]
    curr_max = a[0]

    for i in range(1,size):
        curr_max = max(a[i], curr_max + a[i])
        max_so_far = max(max_so_far,curr_max)

    return max_so_far

# Driver function to check the above function
a = [-2, -3, 4, -1, -2, 1, 5, -3]
print"Maximum contiguous sum is" , maxSubArraySum(a,len(a))

#This code is contributed by _Devesh Agrawal_
```

Run on IDE

Output:

```
Maximum contiguous sum is 7
```

To print the subarray with the maximum sum, we maintain indices whenever we get the maximum sum.

CPP

```
// C++ program to print largest contiguous array sum
#include<iostream>
#include<climits>
using namespace std;

int maxSubArraySum(int a[], int size)
{
    int max_so_far = INT_MIN, max_ending_here = 0,
       start =0, end = 0, s=0;

    for (int i=0; i< size; i++ )
```

```
    {
        max_ending_here += a[i];

        if (max_so_far < max_ending_here)
        {
            max_so_far = max_ending_here;
            start = s;
            end = i;
        }

        if (max_ending_here < 0)
        {
            max_ending_here = 0;
            s = i + 1;
        }
    }
    cout << "Maximum contiguous sum is "
        << max_so_far << endl;
    cout << "Starting index "<< start
        << endl << "Ending index "<< end << endl;
}

/*Driver program to test maxSubArraySum*/
int main()
{
    int a[] = {-2, -3, 4, -1, -2, 1, 5, -3};
    int n = sizeof(a)/sizeof(a[0]);
    int max_sum = maxSubArraySum(a, n);
    return 0;
}
```

Run on IDE

Java

```
// Java program to print largest
// contiguous array sum
class GFG {

    static void maxSubArraySum(int a[], int size)
    {
        int max_so_far = Integer.MIN_VALUE,
        max_ending_here = 0,start = 0,
        end = 0, s = 0;

        for (int i = 0; i < size; i++)
        {
            max_ending_here += a[i];

            if (max_so_far < max_ending_here)
            {
                max_so_far = max_ending_here;
                start = s;
                end = i;
            }

            if (max_ending_here < 0)
            {
                max_ending_here = 0;
                s = i + 1;
            }
        }
        System.out.println("Maximum contiguous sum is "
                           + max_so_far);
        System.out.println("Starting index " + start);
        System.out.println("Ending index " + end);
    }

    // Driver code
    public static void main(String[] args)
    {
```

```
        int a[] = { -2, -3, 4, -1, -2, 1, 5, -3 };
        int n = a.length;
        maxSubArraySum(a, n);
    }
}

// This code is contributed by  prerna saini
```

Run on IDE

Python3

```
# Python program to print largest contiguous array sum

from sys import maxsize

# Function to find the maximum contiguous subarray
# and print its starting and end index
def maxSubArraySum(a,size):

    max_so_far = -maxsize - 1
    max_ending_here = 0
    start = 0
    end = 0
    s = 0

    for i in range(0,size):

        max_ending_here += a[i]

        if max_so_far < max_ending_here:
            max_so_far = max_ending_here
            start = s
            end = i

        if max_ending_here < 0:
            max_ending_here = 0
            s = i+1

    print ("Maximum contiguous sum is %d"%(max_so_far))
    print ("Starting Index %d"%(start))
    print ("Ending Index %d"%(end))

# Driver program to test maxSubArraySum
a = [-2, -3, 4, -1, -2, 1, 5, -3]
maxSubArraySum(a,len(a))
```

Run on IDE

Output:

```
Maximum contiguous sum is 7
Starting index 2
Ending index 6
```

Now try below question

Given an array of integers (possibly some of the elements negative), write a C program to find out the *maximum product* possible by multiplying 'n' consecutive integers in the array where n <= ARRAY_SIZE. Also print the starting point of maximum product subarray.

References:

http://en.wikipedia.org/wiki/Kadane%27s_Algorithm

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GATE CS Corner Company Wise Coding Practice

Dynamic Programming | Amazon-Question | Visa-Question

Login to Improve this Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Recommended Posts:

Maximum subarray sum in O(n) using prefix sum
Ugly Numbers
Dynamic Programming | Set 3 (Longest Increasing Subsequence)
Dynamic Programming | Set 27 (Maximum sum rectangle in a 2D matrix)
Dynamic Programming | Set 4 (Longest Common Subsequence)
Longest Palindromic Substring | Set 1
Maximum number of trailing zeros in the product of the subsets of size k
Balanced expressions such that given positions have opening brackets
Check if it is possible to transform one string to another
Maximum difference of zeros and ones in binary string | Set 2 (O(n) time)

(Login to Rate)

2.6 Average Difficulty : **2.6/5.0**
Based on **495** vote(s)

Basic | Easy | Medium | Hard | Expert

Add to TODO List

Mark as DONE

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments | Share this post!