Lecture 16

Array Problems

Topics for today

Today is all about problem solving. We’ll give you problem statements and discuss solutions with you.

- You need to understand the pattern of the problem statement.
- All the problems ask you to solve some condition on a subarray of size K
- You can find similar problems in leetcode
- The general solution technique is
    - Initialize a state corresponding to the problem you need to solve.
    - Compute a new state for every subarray of size K.
    - Check if the state is a possible solution to the problem or not.
        - Replace the old state value with the new state value if new state is a possible solution.
    - Return the answer state.

Maximum Sum Subarray of Size K

Given an array of positive integers, and a positive number k, find the maximum sum of any contiguous subarray of size k.

Example 1 : Input: [3, 5, 2, 1, 7], k=2

Output: 8

Explanation: Subarray with maximum sum is [1, 7].

Example 2 : Input: a[] = {4, 2, 3, 5, 1, 2}, k = 3

Output: 10

Explanation: Subarray with maximum sum is [2, 3, 5]

Example

```
Students in class = {Naman, Dinesh, Mayank, Shubham, Vikas, Sumit, Nitin, Deepak}
Chocolates        =   5   , 2     , 1     , 2       , 10   , 5    , 1    , 1
3 class Monitors
We'll check for every 3 students next-to-each other
whichever group has max. chocolates will become the monitors

{Naman, Dinesh, Mayank} = 8
{Dinesh, Mayank, Shubham} = 5
{Mayank, Shubham, Vikas} = 13
{Shubham, Vikas, Sumit} = 17
{Vikas, Sumit, Nitin} = 16
{Sumit, Nitin, Deepak} = 7
**/
```

```
{4, 2, 3, 5, 1, 2} , k = 3
   {4, 2, 3} = 9,  startIndex = 0, endIndex = 2
       subarraySum = 0
       when j = 0, subarraySum = 0 + 4 = 4
       when j = 1, subarraySum = 4 + 2 = 6
       when j = 2, subarraySum = 6 + 2 = 8
   {2, 3, 5} = 10, startIndex = 1, endIndex = 3
       subarraySum = 0
       when j = 1, subarraySum = 0 + 2 = 2
       when j = 2, subarraySum = 2 + 3 = 5
       when j = 3, subarraySum = 5 + 5 = 10
   {3, 5, 1} = 9,  startIndex = 2, endIndex = 4
   {5, 1, 2} = 8,  startIndex = 3, endIndex = 5
   maximum sum = 10
```

Bruteforce Solution

```
private static int getMaximumSumOfContiguousSubArray(int[] arr, int k) {
    int maximumSum = -1;
    int subArraySum = 0;

    for(int i = 0; i <= arr.length - k; i++) { // i denotes the start of the sub-array
        subArraySum = 0;
        for(int j = i; j < i+k; j++) { // j denotes the end of the sub-array if i = 0 and k = 3 then j < 3, i = 1 and k = 3 then j < 4
            subArraySum += arr[j]; // subArraySum = subArraySum + arr[j]
            if(subArraySum > maximumSum ) {
                maximumSum = subArraySum;
            }
        }
    }

    return maximumSum;
}
Time Complexity = O(N * K)
Space Complexity =
```

Understanding the loop

```
private static void getMaximumSumOfContiguousSubArray(int[] arr, int k) {

    for(int i = 0; i <= arr.length - k; i++) { // i denotes the start of the sub-array
        subArraySum = 0;
        for(int j = i; j < i+k; j++) { // j denotes the end of the sub-array if i = 0
and k = 3 then j < 3, i = 1 and k = 3 then j < 4
            System.out.println("i = " + i + " , j = " + j);
        }
    }

}
```

Output of for-loop

```
i = 0 , j = 0

i = 0 , j = 1

i = 0 , j = 2

i = 1 , j = 1

i = 1 , j = 2

i = 1 , j = 3

i = 2 , j = 2

i = 2 , j = 3

i = 2 , j = 4

i = 3 , j = 3

i = 3 , j = 4

i = 3 , j = 5
```

Minimum Sum Subarray of Size K

Given an array of positive integers, and a positive number k, find the maximum sum of any contiguous subarray of size k.

Example 1 : Input: [3, 5, 2, 1, 7], k=2

Output: 3

Explanation: Subarray with minimum sum is [2,1].

Example 2 : Input: a[] = {4, 2, 3, 5, 1, 2}, k = 3

Output: 8

Explanation: Subarray with minimum sum is [5,1,2]

```java
private static int findSubarrayWithMinimumSum(int[] arr, int k) {
     int minimumSum = Integer.MAX_VALUE;
     int subArraySum = 0;
     int subArrayStart = -1;
     int subArrayEnd = -1;

     for(int i = 0; i <= arr.length - k; i++) {// i is for the starting position of the subarray
          subArraySum =0;
         for(int j = i; j < i + k; j++) {
              subArraySum += arr[j];
          }
         if(subArraySum < minimumSum) {
              minimumSum = subArraySum;
              subArrayStart = i;
              subArrayEnd = i + (k -1);
          }
      }

      System.out.println("i = " + subArrayStart +" , j = " + subArrayEnd);
     return minimumSum;
 }
```

Does Subarray of Size K with sum P exist or not

Given an array of positive integers, and a positive number k, find the maximum sum of any contiguous subarray of size k.

Example 1 : Input: [3, 5, 2, 1, 7], k=2 ,

When P = 8, Output: true , Explanation : {3,5} {1,7}

When P = 4, Output: false,

```java
private static boolean isSubArrayWithSizeKandSumPPresent(int[] arr, int k, int p) {
    boolean isSumPresent = false;
    int subArraySum = 0;

    for(int i = 0; i <= arr.length - k; i++) { // i is for the start of the subarray
        subArraySum = 0;
        for(int j = i; j < i + k; j++) {
            subArraySum += arr[j];

            if(subArraySum == p) {
                isSumPresent = true;
            }
        }
    }

    return isSumPresent;
}
```