

arr[] = {1, 2, 3, 4, 1}

res = 111

~~3~~ = 001

3 = 011

1 = 001

~~2~~ = 010

4 = 100

~~2~~ = 010

4

$\text{arr}[j] = 5 \rightarrow 3 \ 4 \ 2 \ 1 \}$
 $\text{row} = 111$
 $\text{gcd} = 1$
 $\text{for } (i = 0; i < n; i++) \{$
 $\text{if } (1 \text{ or } \text{arr}[i] \& \text{row}) = 1 \}$
 $\text{num1} = \text{num1} \wedge \text{arr}[i];$
 $\text{do } \{$
 $\text{num2} = \text{num2} \wedge \text{arr}[i];$
 $\}$
 $\}$

From the given Array of integer n , find the maximum length of subarray that are arranged in either even and odd or odd and even.

sample input: $[10, 12, 14, 4, 8]$

output: 3

Explanation: 14 and 7 and 7 and 8 are arranged in alternate pattern

Expected Time complexity as $O(n)$.

$arr[3] = \{3, 4, 1, 6\}$
 $total_sum = 14$, $left_sum = 0$
 $for (i = 0; i < n; i++) \{$
 $\quad total_sum = total_sum - arr[i];$
 $\quad if (total_sum = left_sum) \{$
 $\quad \quad \quad \}$
 $\quad left_sum = left_sum + arr[i];$
 $\}$

From the given Arrays of N integers demand find the maximum sum of sub-Arrays.

$arr = \{1, 2, 3, -2, 5\}$
 $int\ cur_sum = arr[0], max_sum = arr[0];$
 $for (i = 1; i < n; i++)$
 $\{$
 $\quad cur_sum = \max(arr[i], cur_sum + arr[i]);$
 $\quad max_sum = \max(cur_sum, max_sum);$
 $\}$
 $printf("%d", max_sum);$

Window size (w) = 4

Diagram illustrating the width of an array. The array contains elements 3, 8, 2, and 5. The indices 0, 1, 2, and 3 are shown below the elements. A bracket below the indices indicates the width $w = 4$.

$$3 + 8 + 2 + 5 = 18$$

```

cur-sum=0;
for(i=0; i<w; i++){
    cur-sum = cur-sum + arr[i];
}

```

Cur-Sum = 18
max-Sum = Cur-Sum;

3	8	2	5	7	6	12
---	---	---	---	---	---	----

Cur Sum = 18
max_sum = 18

Subtraction

8	2	5	7
---	---	---	---

↓
Adding r to the curSum.

$$\text{for } (i = \underline{1}, i \leq n - w, i \neq \tau) \{$$

```

cur_sum = cur_sum - arr[i-1] + arr[n-w+i];
if (cur_sum > max_sum) {
    max_sum = cur_sum;
}

```