Stickler thief

Solution:-

Loop for all elements in arr[] and maintain two sums sum1 and sum2l where sum1 = Max sum including the previous element and sum2 = Max sum excluding the previous element.

Max sum excluding the current element will be max(sum1, sum2) and max sum including the current element will be sum2 + current element (Note that only sum2 is considered because elements cannot be adjacent).

At the end of the loop return max of sum1 and sum2.

Example:

```
arr[] = {5,  5, 10, 40, 50, 35}
 sum1= 5
 sum2= 0

 For i = 1 (current element is 5)
 sum1=  (sum2+ arr[i])  = 5
 sum2=  max(5, 0) = 5

 For i = 2 (current element is 10)
 sum1 =  (sum2 + arr[i]) = 15
 sum2 =  max(5, 5) = 5

 For i = 3 (current element is 40)
 sum1 = (sum2+ arr[i]) = 45
 sum2 = max(5, 15) = 15

 For i = 4 (current element is 50)
 sum1= (sum2 + arr[i]) = 65
 sum2 =  max(45, 15) = 45

 For i = 5 (current element is 35)
 sum1=  (sum2 + arr[i]) = 80
 sum2 =  max(65, 45) = 65
And 35 is the last element. So, answer is max(incl, excl) = 80
```

Implementation:

C++

```
#include<bits/stdc++.h>

Using namespace std;

/*Function to return max sum such that no two elements  are adjacent */

int FindMaxSum(int arr[], int n)

(

int sum1 = arr[0];

  int sum2 = 0;

  int result,i;

  for (i = 1; i < n; i++)

  {

     /* current max excluding i */

     result = (sum1 > sum2)? sum1: sum2;

     /* current max including i */

     sum1 = sum2 + arr[i];

     sum2 = result;

  }

   /* return max of incl and excl */

   return ((sum1 > sum2)? sum1 : sum2);

}
```

```
/* Driver program to test above function */

int main()

{

   int t;

   cin>>t;

   while(t--)

  {

    cin>>n;

    int arr[n];

   for(int i=0;i<n;i++)

   cin>>a[i];

   cout<<FindMaxSum(arr,n);

}

  return 0;

}
```