Kickstart Round G 2021 - Staying Hydrated

2 , 5, 7

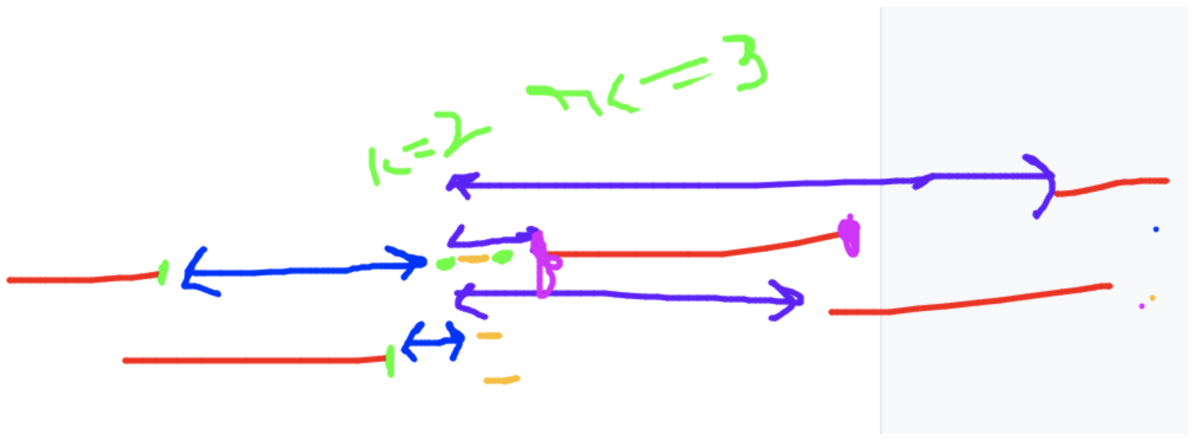
|x-2| + |x-5| + | x-7|

X1, y1   x2, y2

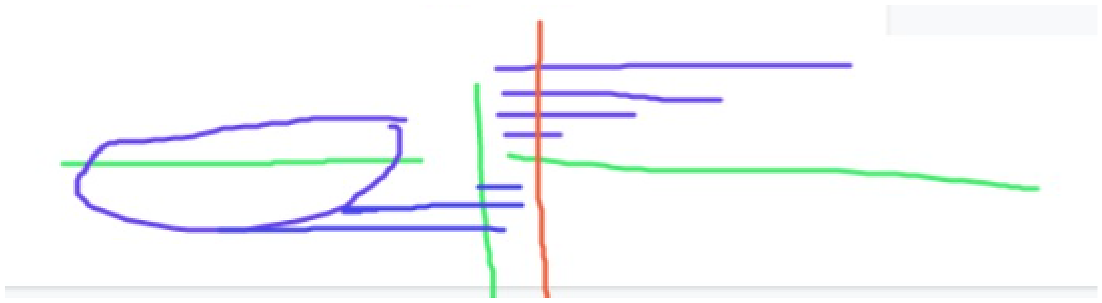
(|x-x1| + |y-y1|)

Other way to do this problem is to simply sort x(of size 2n) and y(of size 2n) separately and get answer as x[n-1] and y[n-1]
The idea is that we can take x and y separately as we are taking sum so we can group our x's and y's and minimize them separately
Now Consider this question, You have to find minimum value of X such that |X-x[i]| is minimum.
Then the answer would always be median among these sorted x[i]. You can go through link provided below:
https://math.stackexchange.com/questions/113270/the-median-minimizes-the-sum-of-absolute-deviations-the-ell-1-norm

Now, here in question we have even number of values 2*n so we have two medians, and at both those points and any point between them would give
the same optimal answer. But the question requires us to give the minimum among them hence we will choose x[n-1] and y[n-1] as the minimum.



Kickstart Round G 2021 - Banana Bunches



```
1 2 3 | 4 4 6 7
Sum ->
0 -> 0
1 -> 1
2 -> 1
3 -> 1
5 -> 2
6 -> 3
Right -> 4 4 => 2
```

Q) You are given a string which is a representation of a number base 2 print a string which is the binary representation of this number base 6

|S| <= 200

(input) 10001 (17 -> decimal) ---->   25 (output)

1010

000000004

101%6 101/6

25/6 25%6

1010 (10)

0000000 → 0000001 (*2 + 1) → 0000002 (*2) → 0000005 (*2 + 1) → 000000(10) -> 0000014 (*2)

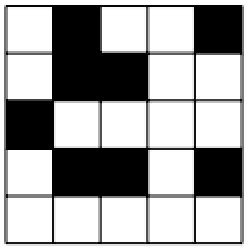
10100 (20)

0000000 → 0000001 (*2 + 1) → 0000002 (*2) → 0000005 (*2 + 1) → 000000(10) -> 0000014 (*2) → 0000001(8) -> 00000(1*2 + 1(carry))2 (*2) → 000032

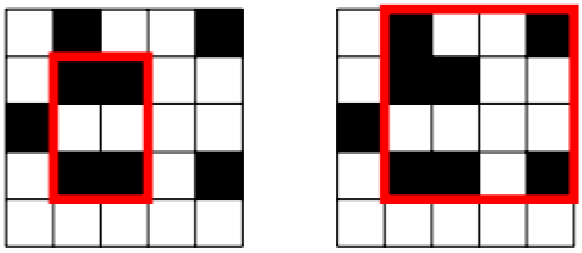
https://www.hackerrank.com/contests/goc-cdc-series-10/challenges/itsybitsy/problem

```cpp
int n;
cin>>n;
vector<int> a(n);
for(auto &i:a){
    cin>>i;
}


vector<int> res;
res.push_back(0);

auto multiply = [&](){
    int carry=0;
    for(int i=0;i<res.size();i++){
        int value = res[i]*2 + carry;
        res[i] = value%6;
        carry = value/6;
        if(carry>0&&res.size()==i+1) res.push_back(0);
    }
};

auto add = [&](){
    int carry=1;
    for(int i=0;i<res.size();i++){
        int value = res[i] + carry;
        res[i] = value%6;
        carry = value/6;
        if(carry>0&&res.size()==i+1) res.push_back(0);
    }
};

for(int i=0;i<n;i++){
    multiply();
    if(a[i]) add();
}

for(auto i:res) cout<<i<<" ";
cout<<"\n";
```

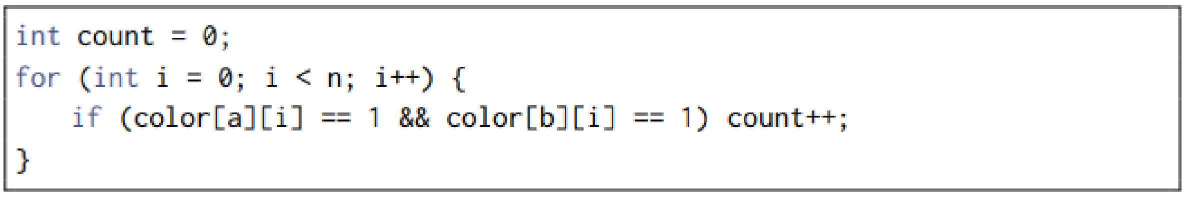
Count Subgrids



01001(9)
01100(12)
10000(16)
01101(13)
00000(0)



```
For a in rows
        For b in rows
```



```
int count = 0;
for (int i = 0; i < n; i++) {
    if (color[a][i] == 1 && color[b][i] == 1) count++;
}
```

Ans += cnt*(cnt-1)/2

Cnt = __builtin_popcount(color[a]&color[b])

120
Row -> 30 30 30 30
M/32
N*N*M/32
1000*1000*1000/32 == 4*10^7

bitset<1000> b;
M/32

```
For a in rows
        For b in rows
                Int cnt = bit[a]|bit[b]
```

Bitset

```
Const int N = 10;
bitset<N> bit(“1110”), sbit; // 0000001110

Bit[0] = 1; 0000001111

bit.set()
bit.reset();
bit.count() // Number of set bits
bit[i]|=(1<<6)       // 0001001110
bit.flip(); 1111110001
```

Q. Find if there exists a subset of items from N items with weight exactly W?
(N <=1000, W<=1000000)

4 -> 3 6 2 4 w[i]
5

```
// possible[i] stores whether we can take items of weight i or not
vector<bool> possible(W+1, false);

possible[0]=true;
for(int i=0;i<n;i++){
        for(int j=W;j>=w[i];j--){
                possible[j]|=possible[j-w[i]];
        }
}
O(N*W)
```

Possible 1 0 0 0 0 0
w[i]=3, 1 0 0 1 0 0

w[i]=2, 1 0 1 1 0 1

```
bitset<1000001> bit(0);
bit[0]=1;
for(int i=0;i<n;i++){
        bit = bit | (bit<<w[i]);
}
Bit => 1 0 0 0 0 0
bit<<3, 0 0 0 1 0 0
w[i]=3 => 1 0 0 1 0 0
bit<<2 => 0 0 1 0 0 1
w[i]=2 => 1 0 1 1 0 1
```

10^9/32 => 4*10^7

https://www.codechef.com/OCT20A/problems/ADDSQURE/