Problem A

He asks you to count how many in range , we can use prefix sum to minimize time for each query.

We can use prefix in this way:

Icon

Description automatically generatedWhen current index equal previous index we increment the value of prefix, otherwise we use previous value.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 1 | 1 | 2 | 3 |

For each query answer is:(How many in range - How many in range .

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | #include <iostream>  #include <bits/stdc++.h>  using namespace std;  int main ()  {  string s; cin>>s; int q, l , r,oo=s.size();  int arr [oo-1];  for (int i = 0; i < oo-1; ++i) {  if(s[i]==s[i+1]){arr[i+1]=arr[i]+1;}  else  {  arr[i+1]=arr[i];  }  }  cin>>q;  while (q--)  {  cin>>l>>r;  cout<<(arr[r-1]-arr[l-1])<<endl;  }  } |

Problem C

We can use prefix sum as problem A, for each number realize the condition we will increment the value of prefix:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | #include <iostream>  #include <bits/stdc++.h>  using namespace std;  int n;  long long arr[N];  long long pref[N];  int check(int num)  {  if(num%10==2||num%10==3||num%10==9){return 1;}  return 0;  }  int main ()  {  cin>>n;  for(int i=1;i<N;i++)  {  pref[i]=pref[i-1]+ check(i);  }  while (n--)  {  int l,r;cin>>l>>r;  cout<<pref[r]-pref[l-1]<<endl;  }  } |

Problem E

In this problem we need to choose optimally. that when we divide **the loss is as small as possible**, so we will use smallest values in array first:

Icon

Description automatically generated with low confidence

|  |  |  |
| --- | --- | --- |
| Sum | Division | loss |
| (200 + 300) = 500 | 250 | 250 |
| (300 + 500) = 800 | 400 | 400 |
| (200 + 500) = 700 | 350 | 350 |

To get first 2 smallest values every time, we will sort.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | #include<iostream>  #include <algorithm>  using namespace std;  int main()  {  int x;  cin>>x;  float arr[x];  for (int i=0;i<x;i++)  {  cin>>arr[i];  }  sort (arr,arr+x);  for (int i=1;i<x;i++)  {  arr[i]=(arr[i]+arr[i-1])/2;  }  cout<<arr[x-1];  return 0;} |

Problem F

A picture containing shape

Description automatically generatedwe need to move 1 from index to index so we need to move 2 horizontal, and 1 vertical.

If 1 in index so answer is:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | #include <iostream>  using namespace std;  int butifule\_matrix(int arr [5][5])  {    for ( int i =0 ; i< 5 ; i++)  {  for ( int j =0 ; j<5 ; j++)  {  if (arr[i][j]==1){ return abs(i-2)+abs(j-2);}  }  }  }  int main ()  {  int arr[5][5];  for ( int i =0 ; i< 5 ; i++)  {  for ( int j =0 ; j<5 ; j++)  {  cin>>arr[i][j];  }  }  cout<<butifule\_matrix(arr);  } |

Problem H

A picture containing text, clipart, vector graphics

Description automatically generated

In this example we have two cases:

* Two neighbors O: we have two options
* Take separately so cost will be
* Take both O so cost will be
* Only one O:
* Take this one with cost

So, we will take minimum when we have two neighbors.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | #include <iostream>  using namespace std;  int main(){  int tc ; cin>>tc;  while (tc--)  {  int n , m , x , y ;  cin>>n>>m>>x>>y;  char arr[n][m];  for (int i = 0; i < n; ++i) {  for (int j = 0; j <m ; ++j) {  cin>>arr[i][j];  }  }  int ans = 0 ;  for (int i = 0; i < n; ++i) {  for (int j = 0; j < m ; ++j) {  if(j+1 <m &&arr[i][j]=='.' && arr[i][j+1]=='.' )  {  ans += min(2\*x , y) ;  j++;  }  else if(arr[i][j]=='.')  {  ans+= x;  }  }  }  cout<<ans<<endl;  }  } |

Problem I

We will use prefix sum to answer each query in , so we need to know the index of character in alphabet.

We can do this using this formula:

Because , and

|  |  |
| --- | --- |
| ‘a’ – ‘a’ | 0 |
| ‘b’ – ‘a’ | 1 |
| ‘c’ – ‘a’ | 2 |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | #include <iostream>  using namespace std;  int main(){  int n , q ;  cin>>n>>q ;  int pref[n+1];  string s ; cin>>s;  pref[0] = 0;  for (int i = 0; i < s.size(); ++i) {  pref[i+1] = pref[i] + (s[i] - 'a'+1) ;  }  while (q--)  {  int l , r ;  cin>>l>>r ;  cout<<pref[r] - pref[l-1]<<endl;  }  } |

Problem J

We search for continuous element in array their sum as small as possible.

I can use prefix sum to get sum of .

Icon

Description automatically generated

I will minimize value .

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | #include <iostream>  using namespace std;  int val(int l , int r , int arr [])  {  if(l==0){return arr[r];}  else{return arr[r]-arr[l-1];}  }  int main(){  int n , k ,ind=0;cin>>n>>k;  int arr[n];  cin>>arr[0];  int minn= 1000000000;  for(int i = 1 ; i< n ;i++)  {  int x; cin>>x;  arr[i]=x+arr[i-1];  }  for(int i = 0 ; i< n-k+1 ; i++)  {  if(val(i,i+k-1,arr) <minn)  {  minn=val(i,i+k-1,arr);  ind=i;  }  }  cout<<ind+1;  } |

Problem K

In this problem we need to minimize the cost of string, first let’s define cost.

If we take any index I and any index L so the cost increase when:

A picture containing text

Description automatically generated

So if we have A B in string try to not have another A B.

By use this sequence:

A AB AC AD B BC BD C CD ….

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | #include <iostream>  using namespace std;  int main(){  int n , k;  cin>>n>>k;  while (n>0) {  for (int j = 0; j < k and n>0; ++j)  {  cout<<(char)(j + 'a'); n--;  for (int l = j+1; l < k and n>0; ++l)  {  cout<<(char)(j + 'a'); n--;  if(n>0)  {  cout<<(char)(l + 'a'); n--;  }  }  }  } |

Problem L

At first:

Subtract two equations:

B never equal C because if , .

So:

Now in first equation:

So always will be odd so is Odd.

if C = N (max value)

So will be any odd value between .

Count odd values in this range in by using .

Or iterate loop on odd values in .