Problem A

We need to find string: in input string, so we will make pointer in in and in input string.

A picture containing text

Description automatically generated

So, when we increment , otherwise increment .

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | int main() {  string trg ="hello",in;  int j=0;  cin>>in;  for(char i : in)  {  if(i==trg[j]){j++;}  }  if(j==5){cout<<"YES"<<endl;}  else{cout<<"NO"<<endl;}  } |

Problem B

We can represent data of laptops as such:

|  |  |
| --- | --- |
| First | Price |
| Second | Quality |

So we need to use array pairs.

Alex search for two laptops and such that

At first, we will sort laptops on Price, it guarantees that

Now we must check if Condition is met or not.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | int main() {  int n ; cin>>n;  pair<int,int> labtops [n] ;  for (int i = 0; i < n; ++i) {  cin>>labtops[i].first>>labtops[i].second;  }  sort(labtops,labtops+n);  for (int i = 0; i < n; ++i) {  if(labtops[i].first < labtops[i+1].first && labtops[i].second>labtops[i+1].second)  {  cout<<"Happy Alex"<<endl;  return 0;  }  }  cout<<"Poor Alex"<<endl;  return 0;  } |

Problem C

Number of substrings is

Proof:

A picture containing calendar

Description automatically generated

If is number of distinct characters in string. Answer is

Problem D

In each row if we find it means this car will turned over during the collision.

In each column if we find it means this car will turned over during the collision.

So, we check for each row if this row dose not contain either 3 or 1 we will count this row.

And we Must use vector to push indices and to print its value.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | int main() {  int n , x;  cin>>n;  vector<int> ans ;  for (int i = 0; i < n; ++i) {  bool check\_row = false ;  for (int j = 0; j <n ; ++j) {  cin>>x;  if(x==1 || x==3){check\_row = true;}  }  if(!check\_row){ans.emplace\_back(i+1);}  }  cout<<ans.size()<<endl;  for(int y : ans)  {cout<<y<<" ";}  } |

Problem E

mean:

**Subtract how many numerators until it is divisible by the denominator**.

So, if it means that we can subtract from numerator to be divisible by C.

Otherwise print -1.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | void solve(int tc = 0) {  int a , b , c ;  cin>>a>>b>>c ;  if(b%c <= (b-a)){cout<< b - (b%c) <<endl;}  else{cout<<-1<<endl;}  } |

Time complexity .

You can use loops and check each number if divisible by C in.

Problem F

A person with the eyes closed

Description automatically generated with low confidenceShape, square

Description automatically generatedThis card gives you points, and opportunity to play additional cards.

Idea: try to play cards as much as possible to increase number of cards.

Firstly, we will sort all pairs based on . it guarantees to take as much as possible cards and if two cards give same will card that have greater.

So, we will create pairs as:

|  |  |
| --- | --- |
| points | second |
| cards | first |

So, when we use sort it will sort pairs based on first, we need to take maximum element we can add another attribute greater<>() to sort function.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | int main() {  int n ; cin>>n;  pair<int,int> B[n];  int b=1 , a=0;  for (int i = 0; i < n; ++i) {  cin>>B[i].second>>B[i].first;  }  sort(B , B+n , greater<>());  int sum = 0 , nof =1;  for (int i = 0; i < n &&nof>0; ++i) {  sum+=B[i].second;  nof+= B[i].first -1;  }  cout<<sum<<endl;  } |

Problem G

Icon

Description automatically generated with medium confidencewe try to separate each digit in integer .

Let’s define first digit as .

And second digit as .

And third digit as .

Now we have last digit in c , we need to get second digit.

So, we will divide by 10 because 123/10 = 12.3 in C++ we equal 12.

Now we can get b by %10

Again, we will divide by 10 to get third digit

Now we have last digit in K so we can say or

To get values back we multiply number in his weight:

Problem J

|  |  |
| --- | --- |
| Description | Current |
| Firstly, we have let’s swap with first element.  now | Text  Description automatically generated with medium confidence |
| Here we have and second element 324. Swap values, | Text  Description automatically generated with low confidence |
| Here we have and fourth element 324. Swap values, | A picture containing text  Description automatically generated |
| Now array is sorted and | Text  Description automatically generated |

We will use function is\_sorted( arr , arr + n ).

While array not sorted, we will iterate over array and check if we have any value we will swap these values.

If we didn’t find any value and array not sorted it means we can’t sort this array with value so we will print

|  |  |
| --- | --- |
| 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 | signed main()  {  MOHARM  int tc;  cin>>tc;  while (tc--)  {  int n , k ;  cin>>n>>k ;  int arr[n];  in(arr,n)  int ans = 0 ;  while (!is\_sorted(arr,arr+n))  {  bool not\_sorted = true ;  for (int &temp : arr) {  if(temp>k){  swap(k ,temp );  not\_sorted = false;  ++ans;  break;  }  }    //NO CHANGE AND WHILE CONDITION NOT TRUE -> NOT SORTED AND CAN'T UPDATE ANY ELEMENT  if(not\_sorted){ ans = - 1 ;break;}  }  cout<<ans<<endl;    }  } |

Problem K

In this problem he has coin, and he want to buy item with coin.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | void solve(int tc = 0) {  int a , b ;  cin>>a>>b ;  int diff = (b - a) ;  if(diff<=0){cout<<0;}  else  {  cout<< ((b - a ) / 10) + (((b - a) % 10) != 0) ;  }  } |

Problem M

A picture containing whiteboard

Description automatically generated

We need to share cards one by one on people.

Let’s define number of cards and number of people .

If we will share cards to first A person.

If we will share cards to all persons and repeat this operation time until we have no card.

**Mod : Subtract how many numerators until it is divisible by the denominator**.

|  |  |  |
| --- | --- | --- |
| Cards | People | note |
| 3 | 3 | Each one will take cards so, we will divide 3 cards for 3 persons and remaining 0 cards so answer = 0 , so last person is 3 |
| 4 | 3 | Each one will take cards so, we will divide 3 cards for 3 persons and remaining 0 cards so answer = 1 , so last person is 1 |

Because we start from A person so we will add this value to number of cards and subtract one because C will take card too.

A picture containing company name

Description automatically generatedProblem N

Let’s create initial array with size 13 \* 13 with empty value.

A black circle with a white background

Description automatically generated with medium confidenceand fill values P and V from input.

* if current cell is W and is P we will make empty cell
* A picture containing text, clipart

  Description automatically generatedif current cell is W and is P we will make empty cell
* if current cell is W and is P we

will make empty cell

* if current cell is W and is P we will make empty cell

|  |  |
| --- | --- |
| 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 | void solve(int tc = 0) {  cin>>n>>m ;  char grid [ 12 ] [ 12 ] ;  //fill grid with .  for (int i = 0; i <= 11; ++i) {  for (int j = 0; j <= 11; ++j) {  grid[i][j] = '.';  }  }  //take input  for (int i = 1; i <= n; ++i) {  for (int j = 1; j <= m; ++j) {  cin>>grid[i][j];  }  }  //algorithm  for (int i = 1; i <= 11; ++i) {  for (int j = 1; j <= 11; ++j) {  if(grid[i][j]=='W' )  {  if(grid[i][j+1]=='P') { grid[i][j + 1] = '.'; }  else if (grid[i][j-1]=='P'){grid[i][j-1] = '.';}  else if(grid[i+1][j]=='P'){grid[i+1][j] = '.';}  else if(grid[i-1][j]=='P'){grid[i-1][j] = '.';}  else{ans--;} //we will increase ans all times ( +1 - 1 )  ans++;  }  }  }  cout<<ans<<endl;  } |