ACM (5 min)

4 stages
3 members
1 computer
5 hours

Intro + Upsolving (20 min)

Primitive Data Types:
- 1 byte: char (-128 → 127 -0 → 255) / bool (0-1)
- 2 bytes: short
- 4 bytes: int (2^63-1) 2*10^9 / float
- 8 bytes: long long int (10^19) - double

http://www.cplusplus.com/doc/tutorial/variables/

Helpful Library: (Needs a g++ compiler - CodeBlocks or Eclipse for C++)
#include “bits/stdc++.h”

Arrays:
Contiguous Memory -> Base index + shift
int ar[5]={}; // all initialized to 0
char ar[5]={}; // all initialized to 0

Popular C++ Functions:
#include “algorithm”
-sort
int a[10]={10,3,3,2,1};
sort(a,a+5);

-min/max
int x=2,y=3;
int z=min(x,y);//2
int zz=max(x,y);//3

-swap
int x=2,y=3;
swap(x,y); // x=3 , y=2

(No of changes) http://codeforces.com/problemset/problem/155/A

String Class (?)+ Complexity (?)

A string is a dynamic char array.
char a[]=”amaasas”; is equivalent to **string s=”amaasas”;**

Methods:
- Constructor:
string s;
string s(5,’a’);// string s=”aaaaa”;

- size() / length()
string s=”01”;
cout<<s.size(); // 2

- []
string s=”abc”;
cout<<s[1]; // b

- +=
string s=”Bassem”;
s+=” Ossama”;
cout<<s; //s = ”Bassem Ossama”

- find() -> returns index of first char if found , -1 if not found
sring s=”abc”;
int in=s.find(“c”); // 2 → -1
if(in==-1) cout<<”Not Found”;
int in2=s.find("a",1); // search for a starting from index 1

- substr()
string s=”abcde”;
string a1=s.substr(1,3);//bcd
string a2=s.substr(1);//bcde

- erase()
string s=”abcde”;
s.erase(1,3);//ae
s.erase(1,3);//a

- compare()
string s=”ab”,s1=”abc”
cout<<(int)s.compare(s1);

- getline()
getline(cin,s);

Properties:
-Array of characters
(First is capital) http://codeforces.com/problemset/problem/281/A
(Convert Case & Minimize Operations) http://codeforces.com/problemset/problem/59/A

-Substring:
(Has AB and BA) http://codeforces.com/problemset/problem/550/A

- No of substrings=??
string s=”abcd”; // 4
a ab abc abcd → length
b bc bcd → length-1
c cd → length-2
d → length-3

No of substrings=∑(length-i) i:0→ n = (n+1)*n – (n(n+1)/2) = 0.5 * n* (n+1)

How to generate all substrings in a string?

```
String s=”abcd”;
for(int startindx=0;startindx<s.size();startindx++)
{
          for(int endindx=startindx;endindx<s.size();endindx++)
          {
                    cout<<s.substr(startindx,endindx+1-startindx)<<endl;
          }
}
```

- Lexicographical order
aa<aab
ab<ac

- Palindromes
AbA a aa acca are palindromes.
ab Aa aren’t palindromes
(Add char to make palindrome)http://codeforces.com/problemset/problem/505/A

-Anagrams (Strings of the same chars)
abc cba cab acb

<u>Time Complexity:</u>

- 10^7 operation → 1 second
- Worst Case Scenario
- Big O notation : https://en.wikipedia.org/wiki/Big_O_notation

---

```
int a=0,b=2; // 2 operations
for(int i=0;i<1000;i++) // initialization 1 op - checks 1001 op - incrementation 1000 op
{
        a++; //1000 operation
}
```

Total No of Operations = 2+1+1001+1000+1000 = 3*1000 + 4
(Depend more on the higher factor -1000-)

O(n)+O(n) = O(n)
O(n)+O(log2(n)) = O(n) Ex:→ n=10^18
10^18 op , 64 op

```
for(int i=0;i<n;i++)
{
        for(int j=0;j<n;j++)
          ; //n operations
}
```

n*n operations -> O(n^2)
n=100 -> 1,0000 operations (OK)
n=10^5 → 10^10 operations (PROBLEM!!!)

Onsite Contest -> problem H
http://codeforces.com/group/dwsLKsQhIS/contest/208286/problem/H
Solution 1 :
```
for(int q=0;q<Q;q++)
{
cin>>l>>r;
long long sum=0;
for(int i=l;i<=r;i++)
{
        int val=i*i;
        sum+=val;
}
cout<<sum;
}
```
O(Q*10^5) -> Q<=10^5 -> Worst Case: 10^10 Operations (PROBLEM!!!!)

Solution 2:

$\sum_{1}^{n} i^{2} =$ (n)*(n+1)*(2*n+1) /6

```
long long findSum(long long n)
{
        return (n)*(n+1)*(2*n+1) /6; // O(1)
}

for(int i=0;i<Q;i++)
{

        long long int l,r;
        long long sum=findSum(r)-findSum(l-1)+(l*l);
        cout<<sum<<endl;
}
```

Solution 3:

```
long long sumTillIndexI[100005];
0→ I
sumTillIndexI[0]=0;
sumTillIndexI[1]=0 + 1;
sumTillIndexI[2]=0 + 1 + 4;
sumTillIndexI[3]=0+ 1 + 4 + 9;
sumTillIndexI[4]=0+ 1 + 4 + 9 + 16;
for(int i=1;i<100005;i++)
{
        sumTillIndexI[i]=(i*i)+sumTillIndexI[i-1];
}
O(10^5)
for(int q=0;q<Q;q++)
{
cin>>l>>r;
cout<<sumTillIndexI[r]-sumTillIndexI[l-1];
}

O(Q=10^5)
```

Draft:

-"Combinatorics: Permuations – Combinations"

Given:n

Output: Sum from 1 to n.

n*(n+1)/2

SUM(1 → 5)

s=1+2+3+4+5

s=5+4+3+2+1

6=n+1

2*s=(6)+(6)+(6)+(6)+(6)

2*s=(n+1)+(n+1)+(n+1)+(n+1)+(n+1)

2*s=n(n+1)

s=n(n+1)/2