My Tips for USACO

ACM; Jack Wu

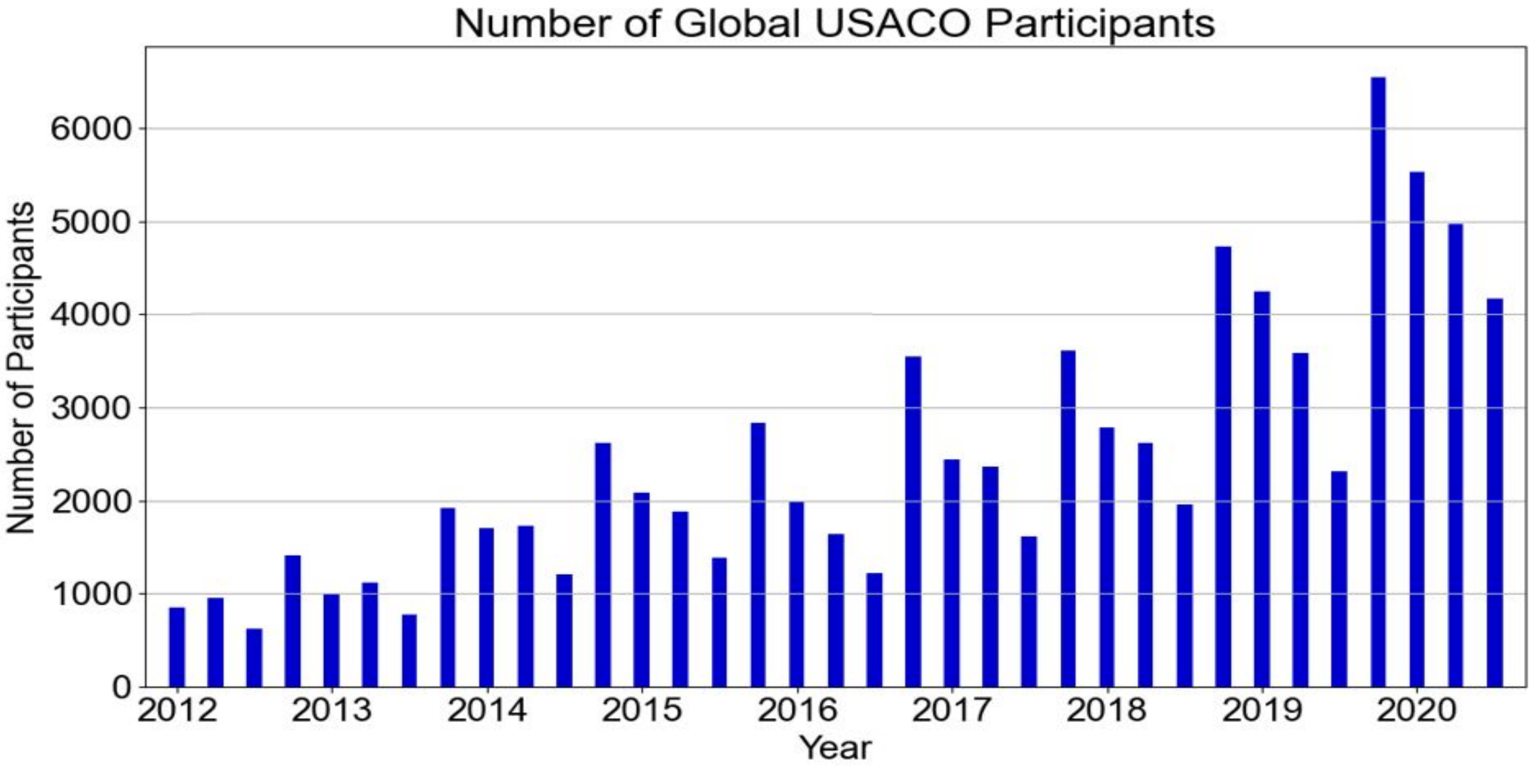
General Information

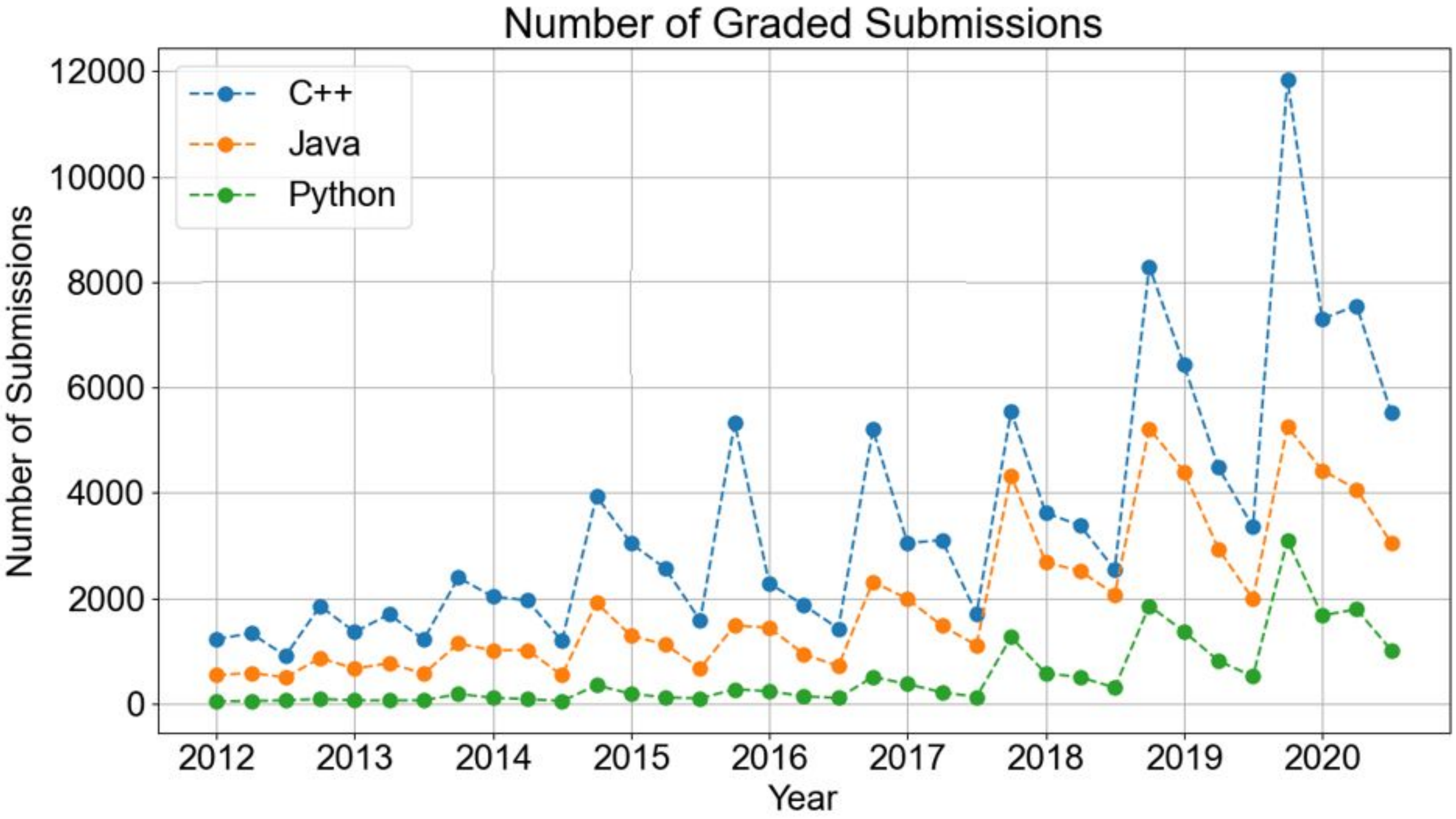
A) USACO is becoming increasingly popular

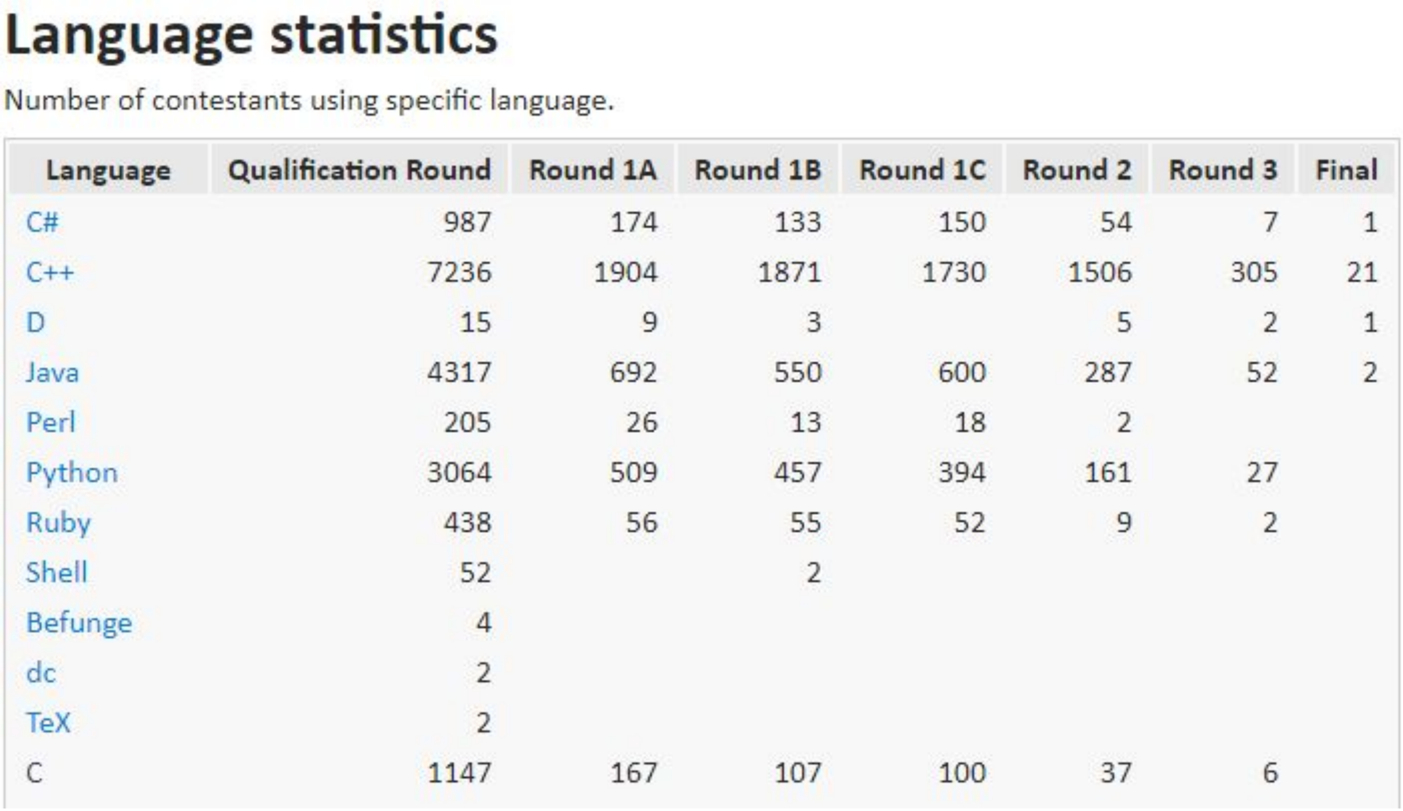
1. CS is becoming increasingly popular
2. Boosted by programs such as AP CS
3. USACO is an at-home competition, which makes it easy to join during COVID

B) December contest gets the most students

1. Beginning of a new contest season




Number of Graded Submissions
C++
Java
Python
Number of Submissions
0
2000
4000
6000
8000
10000
12000
2012
2013
2014
2015
2016
2017
2018
2019
2020
Year

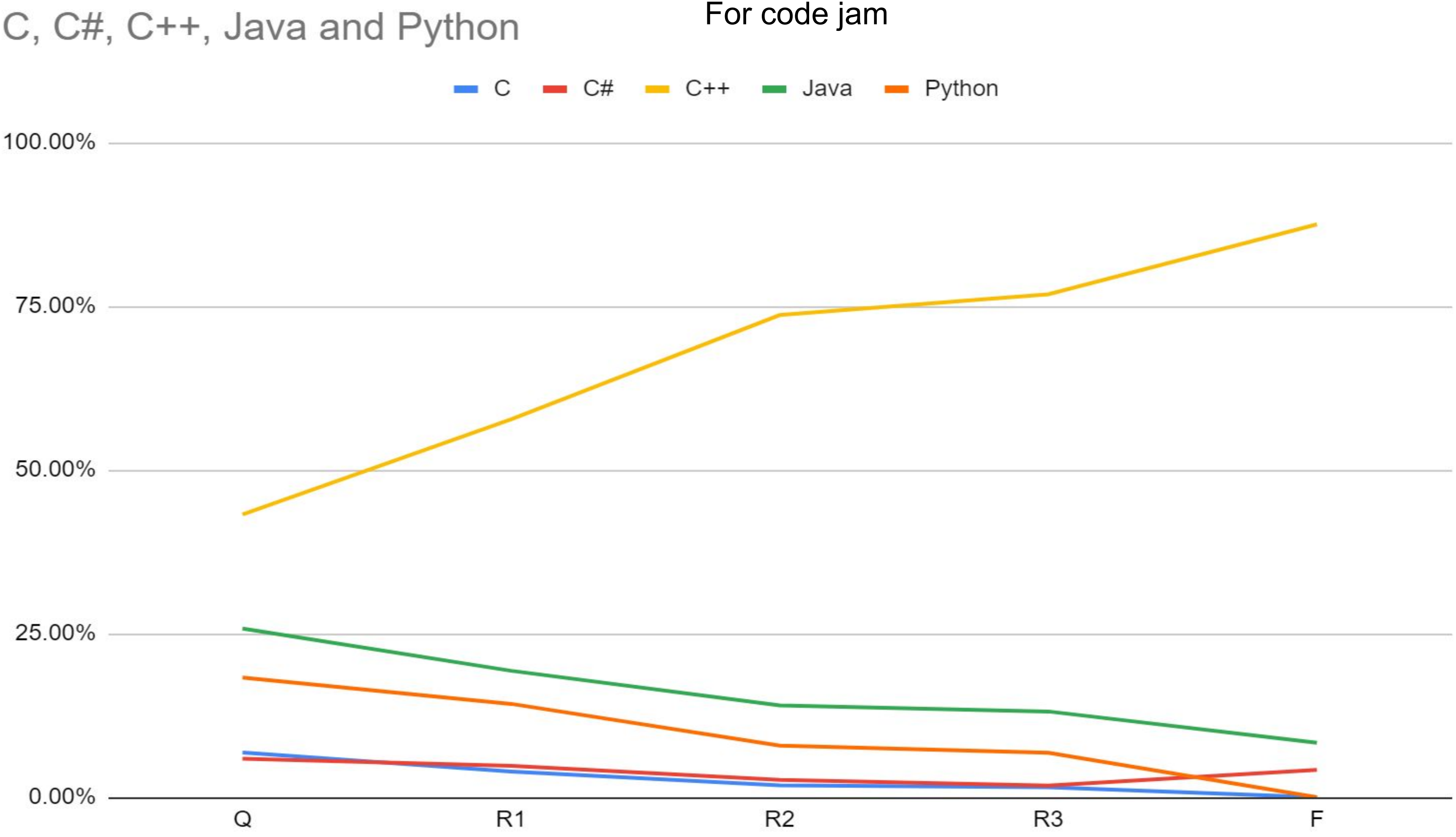


Language statistics

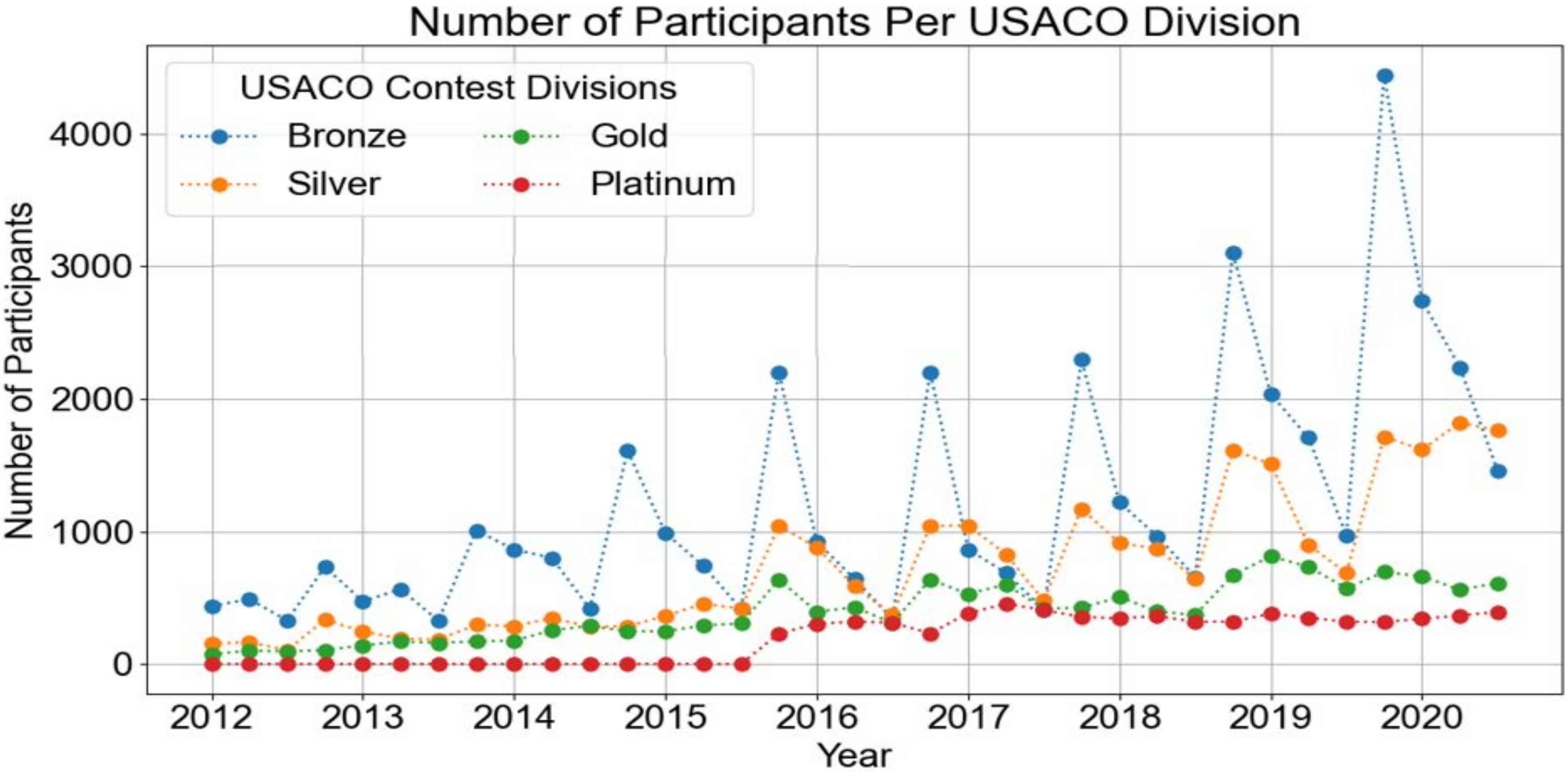
Number of contestants using specific language.

Language	Qualification Round	Round 1A	Round 1B	Round 1C	Round 2	Round 3	Final
C#	987	174	133	150	54	7	1
C++	7236	1904	1871	1730	1506	305	21
D	15	9	3		5	2	1
Java	4317	692	550	600	287	52	2
Perl	205	26	13	18	2		
Python	3064	509	457	394	161	27	
Ruby	438	56	55	52	9	2	
Shell	52		2				
Befunge	4						
dc	2						
TeX	2						
C	1147	167	107	100	37	6	

For code jam

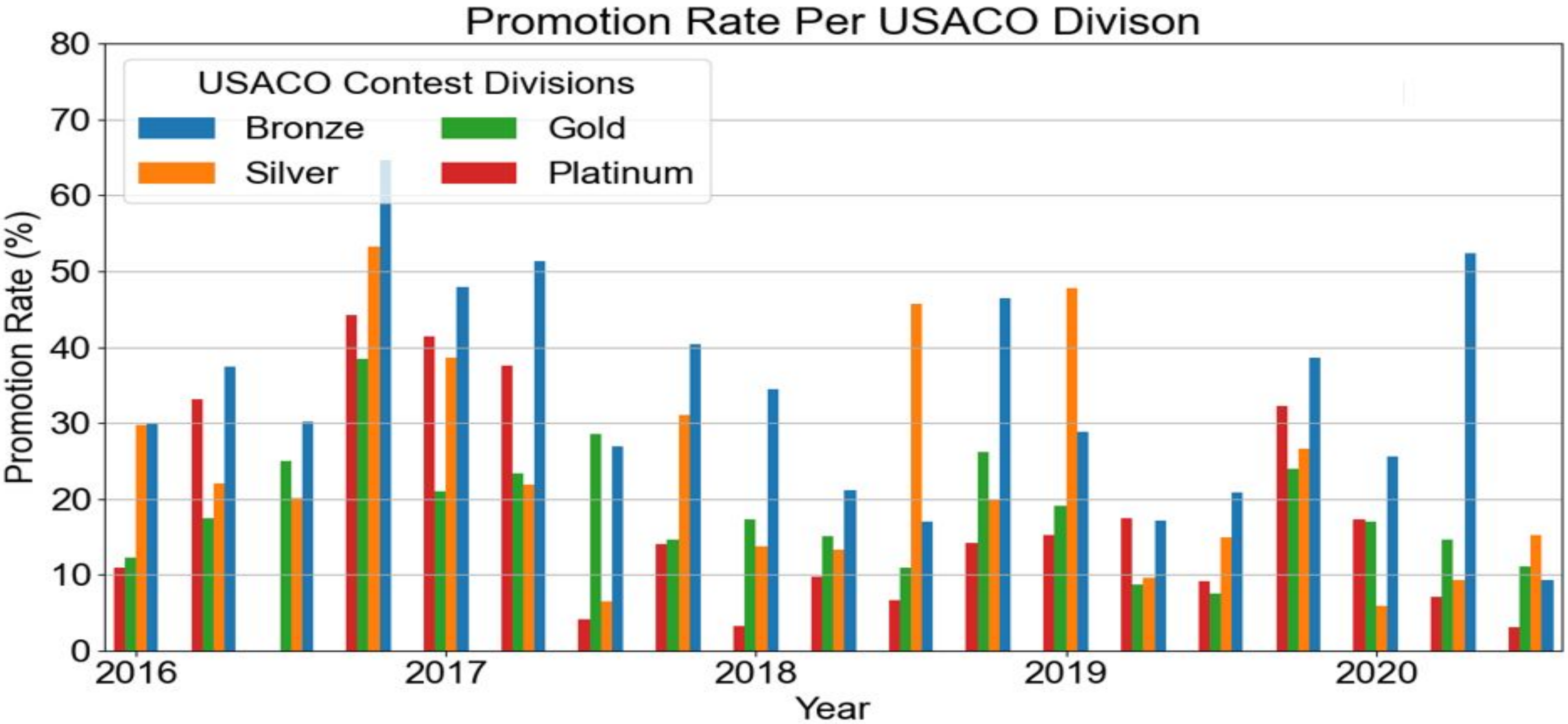

C, C#, C++, Java and Python
C
C#
C++
Java
Python
100.00%
75.00%
50.00%
25.00%
0.00%
Q
R1
R2
R3
F

# of Bronze = 2.5 # of Silver = 8 # of Gold = 20 # of Platinum



Promotion Rate: Bronze 30% > Silver 20% > Gold 10% > Platinum listing 5%

Accumulate: Bronze 30%, Silver 6%, Gold 0.6%, Platinum listing 0.03%



Comparison to Competitive Math

Math	AMC 8	AMC 10/12	AIME	USAMO	MOP	IMO
USACO	Bronze	Silver	Gold	Platinum	Camp	IOI

General Problem Solving Procedure

1. Determine a target time complexity
2. Solve the first sample test case on paper
3. Generalize your methods
4. Develop your algorithm
5. Consider edge cases
6. Begin testing as early as possible
7. Write complicated test cases

Computer Science’s Mathematical Basis

You’ve heard that computer science is heavily based on math

In USACO, especially at higher levels, is based on complex math and number theory

Complex problems require analyzing, generalizing, and simplifying to make them solvable by computers

USACO problems draw on very similar skills as do AMC and AIME problems

A simple application of the power of Math

How would you program a computer to calculate

$$2^{16}$$

as quickly as possible?

Mathematics needed:

- Euler’s Totient Function $$\varphi(n) = n \prod_{p|n} \left(1 - \frac{1}{p}\right)$$
- Mobius Function $$\sum_{n=1}^{\infty} \frac{\mu(n)}{n^s} = \frac{1}{\zeta(s)}$$
- Fermat’s Little Theorem $$a^p \equiv a \pmod{p}.$$ $$a^{p-1} \equiv 1 \pmod{p}.$$
- Binomial Transformation $$s_n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} a_k.$$
- Mobius Transformation $$f(n) = \sum_{d|n} \mu(d) g\left(\frac{n}{d}\right) \quad \text{for every integer } n \geq 1$$
- Dirichlet Convolution $1 = d * \mu$ $1 * \mu = \varepsilon$ $\phi * 1 = \mathrm{Id}$

The Formulas to Remember

φ Totient

μ Mobius

d Number of Factors

σ Sum of Factors

$$\varepsilon(x) = [x == 1]$$

$$I(x) = 1$$

$$Id(x) = x$$

$$h(n) = \sum_{d|n} f(d)g(n/d)$$

$$f * \varepsilon = f$$

$\mu * I = \varepsilon$ (mobius transformation)

$$f * \mu * I = f$$

$$\varphi * I = Id$$

$$n = \sum_{d|n} \varphi(d)$$

$$\mu * Id = \varphi$$

First, we have to convert the USACO problem into a math problem

$$\prod_{i=1}^{N}\prod_{j=1}^{N}\frac{lcm(i,j)}{gcd(i,j)}(mod\ p)$$

(where p is prime)

When

- N<10, it’s an elementary school problem
- 10<=N<20, it’s a middle school problem
- 20<=N<=100, it’s a high school problem
- 100<N<=5000, it’s an AMC/AIME problem
- N>5000, it’s a USACO problem

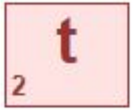
USACO: N=1,000,000 P=104857601 solve in 0.2 seconds

Time Complexity

The problem requires N <= 1000000

That means a brute force approach would require on the order of N^2, or 1000000000000, computations

This will result in a TLE (Time Limit Exceeded) Error:



“Booo! You need a better algorithm!”

Wolfram fails at n=100:

Product[Product[lcm(i,j)/gcd(i,j) , {i, 1, 100}],{j, 1, 100}] (mod 104857601)

We need a O(N log N) or better solution; one that exploits mathematical patterns to use much less computation.

$$\prod_{i=1}^{N}\prod_{j=1}^{N}\frac{lcm(i,j)}{gcd(i,j)}(mod\ p)$$

Calculate $\prod_{i=1}^{N}\prod_{j=1}^{N}\frac{lcm(i,j)}{gcd(i,j)}$ first

$$\prod_{i=1}^{N}\prod_{j=1}^{N}\frac{lcm(i,j)}{gcd(i,j)} = \prod_{i=1}^{N}\prod_{j=1}^{N}\frac{i*j}{(gcd(i,j))^{2}} = \frac{(N!)^{2N}}{(\boxed{\prod_{i=1}^{N}\prod_{j=1}^{N}gcd(i,j)})^{2}}$$

$$\prod_{i=1}^{N}\prod_{j=1}^{N}gcd(i,j) = \prod_{d=1}^{N}\prod_{i=1}^{N}\prod_{j=1}^{N}[gcd(i,j)==d] = \prod_{d=1}^{N}d^{\sum_{i=1}^{N}\sum_{j=1}^{N}[gcd(i,j)==d]} = \prod_{d=1}^{N}d^{\sum_{i=1}^{N/d}\sum_{j=1}^{N/d}[gcd(i,j)==1]}$$

2 methods with different complexity

$\varphi(mod) = mod - 1$ then O(nlogn)

Mobius Transformation O(n)

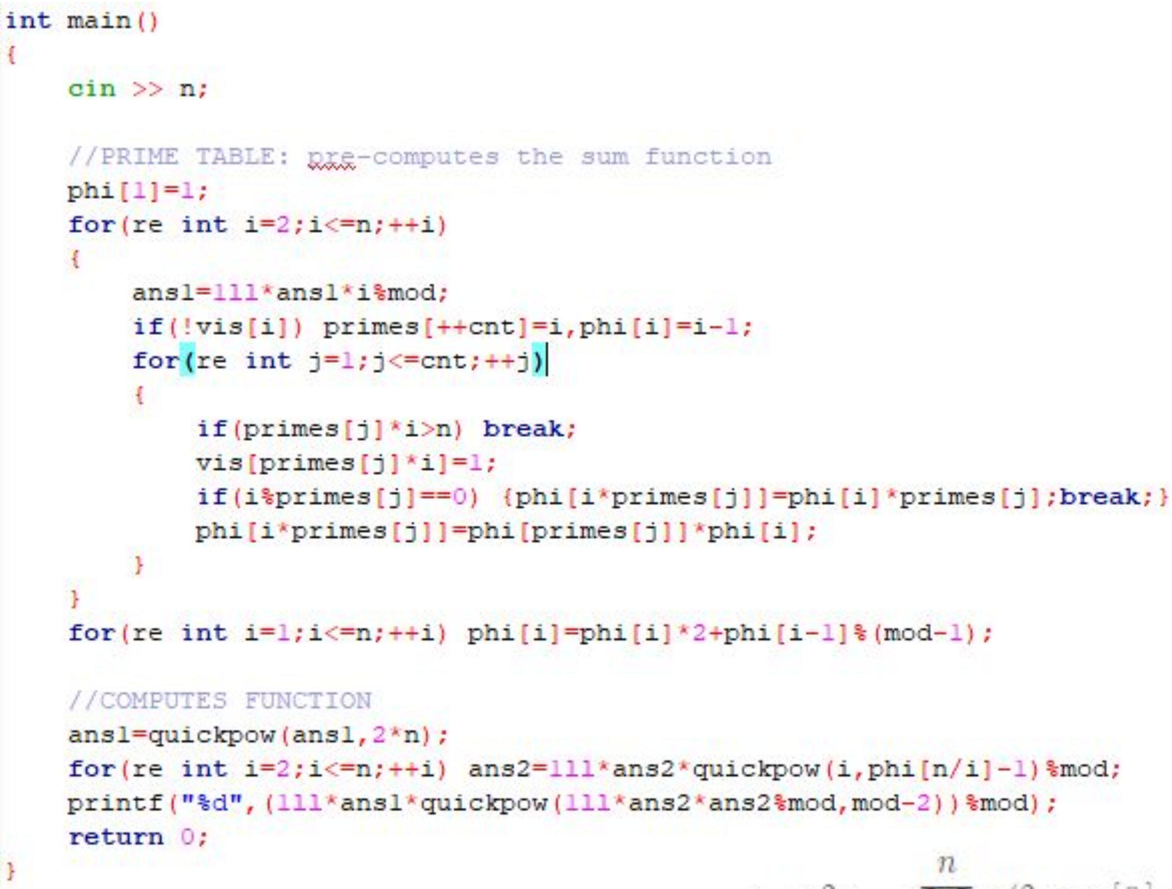
$$\varphi(mod) = mod - 1$$

$$(n!)^{2n} * (\prod_{d=1}^{n} d^{(2*sum[\frac{n}{d}]-1)\%(mod-1)})^{-2}$$

$$\sum_{i=1}^{N/d}\sum_{j=1}^{N/d}[gcd(i,j)==1] = 2sum[\tfrac{n}{d}] - 1\%(mod-1)$$

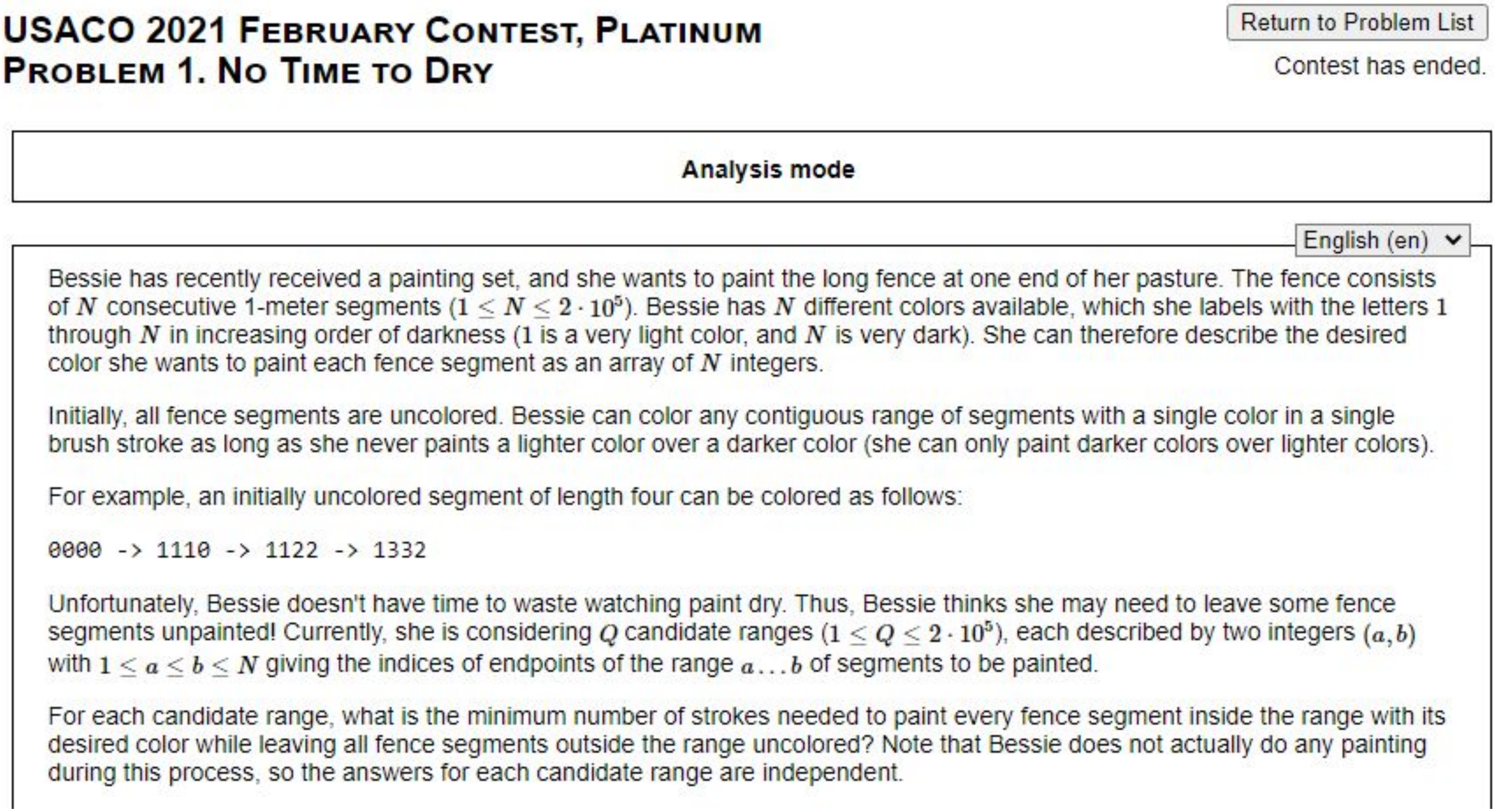
Mobius:

$$\sum_{i=1}^{N/d}\sum_{j=1}^{N/d}[gcd(i,j)==1] = \sum_{i=1}^{N/d}\sum_{j=1}^{N/d}\sum_{g|gcd(i,j)}\mu(i) = \sum_{g=1}^{N/d+1}\mu(i)\sum_{i=1}^{N/dg}\sum_{j=1}^{N/dg} = \sum_{g=1}^{N/d+1}\mu(i) * \frac{N}{dg} * \frac{N}{dg}$$



```
int main()
{
    cin >> n;

    //PRIME TABLE: pre-computes the sum function
    phi[1]=1;
    for(re int i=2;i<=n;++i)
    {
        ans1=1ll*ans1*i%mod;
        if(!vis[i]) primes[++cnt]=i,phi[i]=i-1;
        for(re int j=1;j<=cnt;++j)
        {
            if(primes[j]*i>n) break;
            vis[primes[j]*i]=1;
            if(i%primes[j]==0) {phi[i*primes[j]]=phi[i]*primes[j];break;}
            phi[i*primes[j]]=phi[primes[j]]*phi[i];
        }
    }
    for(re int i=1;i<=n;++i) phi[i]=phi[i]*2+phi[i-1]%(mod-1);

    //COMPUTES FUNCTION
    ans1=quickpow(ans1,2*n);
    for(re int i=2;i<=n;++i) ans2=1ll*ans2*quickpow(i,phi[n/i]-1)%mod;
    printf("%d",(1ll*ans1*quickpow(1ll*ans2*ans2%mod,mod-2))%mod);
    return 0;
}
```

$$(n!)^{2n} * (\prod_{d=1}^{n} d^{(2*sum[\frac{n}{d}]-1)\%(mod-1)})^{-2}$$



USACO 2021 February Contest, Platinum
Problem 1. No Time to Dry

Bessie has recently received a painting set, and she wants to paint the long fence at one end of her pasture. The fence consists of N consecutive 1-meter segments ($1 \le N \le 2 \cdot 10^5$). Bessie has N different colors available, which she labels with the letters 1 through N in increasing order of darkness (1 is a very light color, and N is very dark). She can therefore describe the desired color she wants to paint each fence segment as an array of N integers.

Initially, all fence segments are uncolored. Bessie can color any contiguous range of segments with a single color in a single brush stroke as long as she never paints a lighter color over a darker color (she can only paint darker colors over lighter colors).

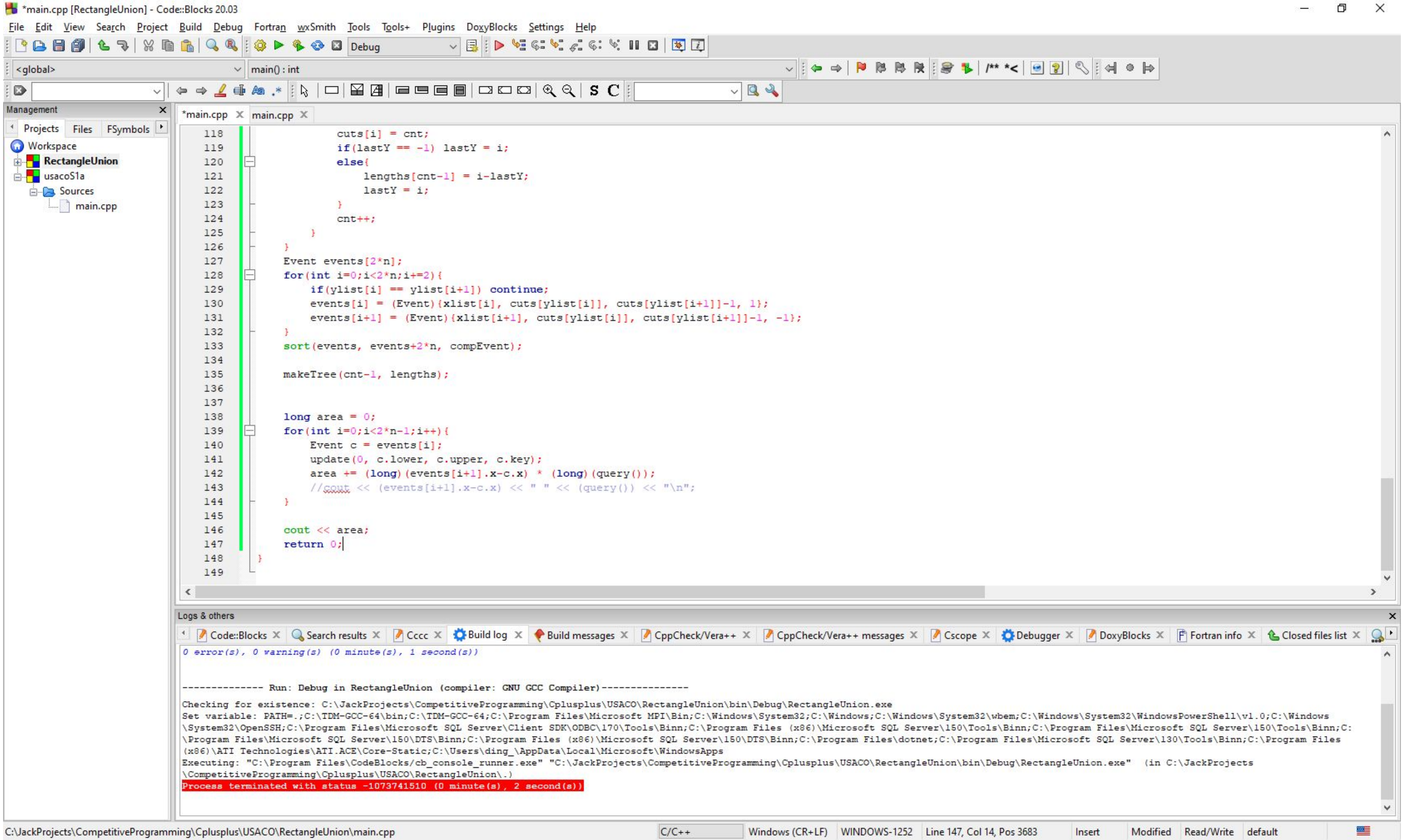
For example, an initially uncolored segment of length four can be colored as follows:

```
0000 -> 1110 -> 1122 -> 1332
```

Unfortunately, Bessie doesn't have time to waste watching paint dry. Thus, Bessie thinks she may need to leave some fence segments unpainted! Currently, she is considering Q candidate ranges ($1 \le Q \le 2 \cdot 10^5$), each described by two integers (a, b) with $1 \le a \le b \le N$ giving the indices of endpoints of the range $a \ldots b$ of segments to be painted.

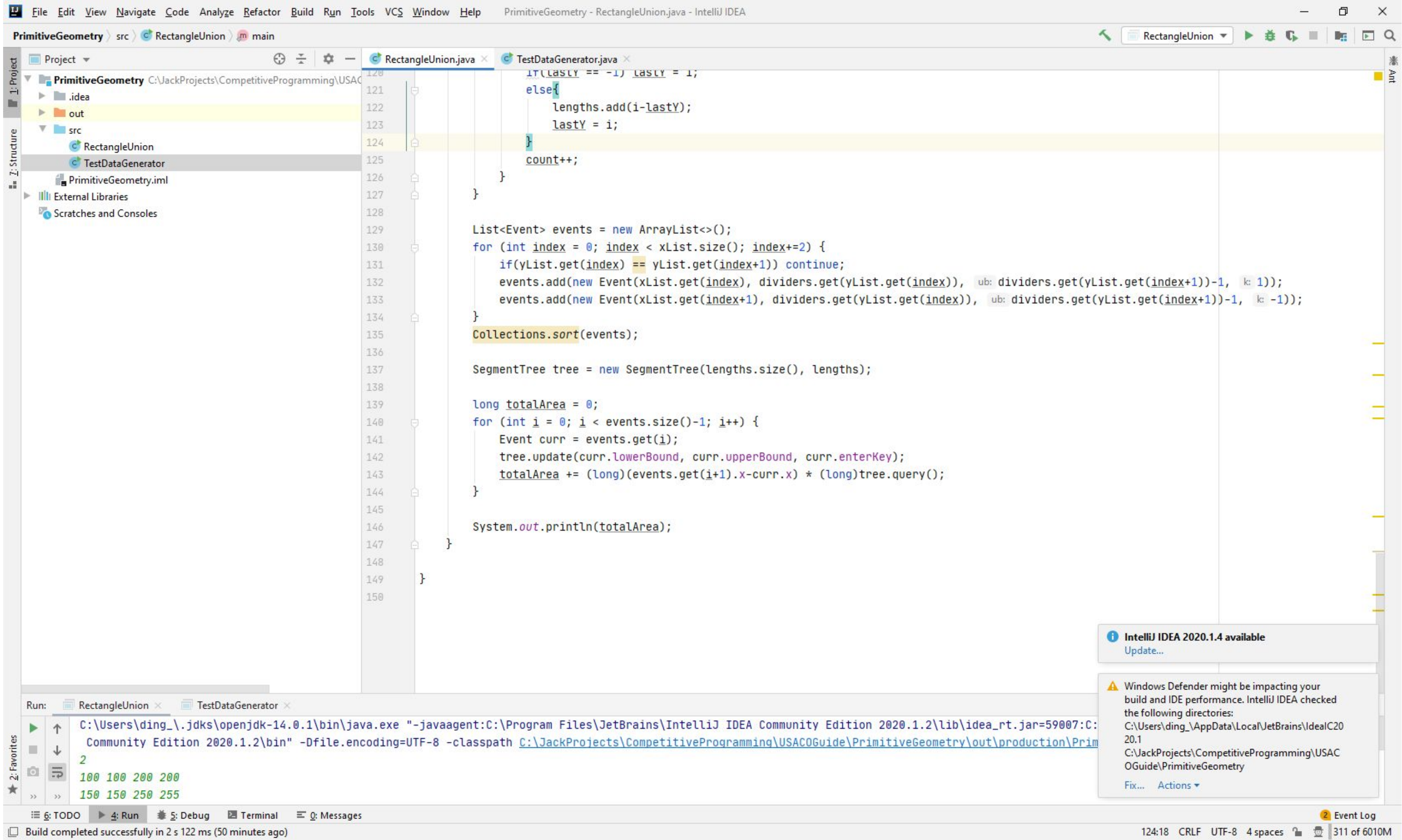
For each candidate range, what is the minimum number of strokes needed to paint every fence segment inside the range with its desired color while leaving all fence segments outside the range uncolored? Note that Bessie does not actually do any painting during this process, so the answers for each candidate range are independent.

```
#include<bits/stdc++.h>
using namespace std;
const int N=200099;
struct Pt{int c1,c2,acc;}P[N<<5];
int totalindex;
void recurs1(int a,int l,int r,int split){
    totalindex++;
    int curr = totalindex;
    if (l==r){
        P[curr].acc=P[a].acc+1;
        return;
    }
    P[curr].c1=P[a].c1; P[curr].c2=P[a].c2;
    int mid=(l+r)/2;
    if (split<=mid){
        P[curr].c1 = totalindex+1;
        recurs1(P[a].c1,l,mid,split);
    }
    else{
        P[curr].c2 = totalindex+1;
        recurs1(P[a].c2,mid+1,r,split);
    }
P[curr].acc=P[P[curr].c1].acc+P[P[curr].c2
].acc;
}
int rcrs2(int index,int intL,int intR,int l,int r){
    if (intL<=l&&r<=intR){
        return P[index].acc;
    }
    int sum=0;
    int mid=(l+r)/2;
    if (intL<=mid)
sum+=rcrs2(P[index].c1,intL,intR,l,mid);
    if (mid<intR)
sum+=rcrs2(P[index].c2,intL,intR,mid+1,r)
;
    return sum;
}
int top[N];
int main(){
    int n,j,q,Top[N],t[N];
    cin>>n>>q;
    for (int i=1;i<=n;i++){
        int a;
        cin>>a;
        while (j>0&&t[j]>a) j--;
        if (t[j]==a){
            top[i] = totalindex+1;
            recurs1(top[i-1],1,n,Top[j--]);
        }
        else top[i]=top[i-1];
        Top[++j]=i;
        t[j]=a;
    }
    for (int qu=0;qu<q;qu++){
        int l, r;
        cin>>l>>r;

cout<<r-l+1-rcrs2(top[r],l,r,1,n)+rcrs2(top[l-1],l,
r,1,n)<<"\n";
    }
    return 0;
}
```


*main.cpp [RectangleUnion] - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
main() : int
Management
Projects Files FSymbols
Workspace
RectangleUnion
usacoS1a
Sources
main.cpp
*main.cpp
main.cpp
Logs & others
Code::Blocks Search results Cccc Build log Build messages CppCheck/Vera++ CppCheck/Vera++ messages Cscope Debugger DoxyBlocks Fortran info Closed files list
C:\JackProjects\CompetitiveProgramming\Cplusplus\USACO\RectangleUnion\main.cpp
C/C++ Windows (CR+LF) WINDOWS-1252 Line 147, Col 14, Pos 3683 Insert Modified Read/Write default

```
            cuts[i] = cnt;
            if(lastY == -1) lastY = i;
            else{
                lengths[cnt-1] = i-lastY;
                lastY = i;
            }
            cnt++;
        }
    }
    Event events[2*n];
    for(int i=0;i<2*n;i+=2){
        if(ylist[i] == ylist[i+1]) continue;
        events[i] = (Event){xlist[i], cuts[ylist[i]], cuts[ylist[i+1]]-1, 1};
        events[i+1] = (Event){xlist[i+1], cuts[ylist[i]], cuts[ylist[i+1]]-1, -1};
    }
    sort(events, events+2*n, compEvent);

    makeTree(cnt-1, lengths);


    long area = 0;
    for(int i=0;i<2*n-1;i++){
        Event c = events[i];
        update(0, c.lower, c.upper, c.key);
        area += (long)(events[i+1].x-c.x) * (long)(query());
        //cout << (events[i+1].x-c.x) << " " << (query()) << "\n";
    }

    cout << area;
    return 0;
}

```

```
0 error(s), 0 warning(s) (0 minute(s), 1 second(s))


-------------- Run: Debug in RectangleUnion (compiler: GNU GCC Compiler)---------------

Checking for existence: C:\JackProjects\CompetitiveProgramming\Cplusplus\USACO\RectangleUnion\bin\Debug\RectangleUnion.exe
Set variable: PATH=.;C:\TDM-GCC-64\bin;C:\TDM-GCC-64;C:\Program Files\Microsoft MPI\Bin;C:\Windows\System32;C:\Windows;C:\Windows\System32\wbem;C:\Windows\System32\WindowsPowerShell\v1.0;C:\Windows
\System32\OpenSSH;C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\Tools\Binn;C:\Program Files (x86)\Microsoft SQL Server\150\Tools\Binn;C:\Program Files\Microsoft SQL Server\150\Tools\Binn;C:
\Program Files\Microsoft SQL Server\150\DTS\Binn;C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn;C:\Program Files\dotnet;C:\Program Files\Microsoft SQL Server\130\Tools\Binn;C:\Program Files
(x86)\ATI Technologies\ATI.ACE\Core-Static;C:\Users\ding_\AppData\Local\Microsoft\WindowsApps
Executing: "C:\Program Files\CodeBlocks/cb_console_runner.exe" "C:\JackProjects\CompetitiveProgramming\Cplusplus\USACO\RectangleUnion\bin\Debug\RectangleUnion.exe"  (in C:\JackProjects
\CompetitiveProgramming\Cplusplus\USACO\RectangleUnion\.)
Process terminated with status -1073741510 (0 minute(s), 2 second(s))
```



```
                if(lastY == -1) lastY = i;
                else{
                    lengths.add(i-lastY);
                    lastY = i;
                }
                count++;
            }
        }

        List<Event> events = new ArrayList<>();
        for (int index = 0; index < xList.size(); index+=2) {
            if(yList.get(index) == yList.get(index+1)) continue;
            events.add(new Event(xList.get(index), dividers.get(yList.get(index)),  ub: dividers.get(yList.get(index+1))-1,  k: 1));
            events.add(new Event(xList.get(index+1), dividers.get(yList.get(index)),  ub: dividers.get(yList.get(index+1))-1,  k: -1));
        }
        Collections.sort(events);

        SegmentTree tree = new SegmentTree(lengths.size(), lengths);

        long totalArea = 0;
        for (int i = 0; i < events.size()-1; i++) {
            Event curr = events.get(i);
            tree.update(curr.lowerBound, curr.upperBound, curr.enterKey);
            totalArea += (long)(events.get(i+1).x-curr.x) * (long)tree.query();
        }

        System.out.println(totalArea);
    }

}
```

```
C:\Users\ding_\.jdks\openjdk-14.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2020.1.2\lib\idea_rt.jar=59007:C:
 Community Edition 2020.1.2\bin" -Dfile.encoding=UTF-8 -classpath C:\JackProjects\CompetitiveProgramming\USACOGuide\PrimitiveGeometry\out\production\Prim
2
100 100 200 200
150 150 250 255
```

IntelliJ IDEA 2020.1.4 available
Update...

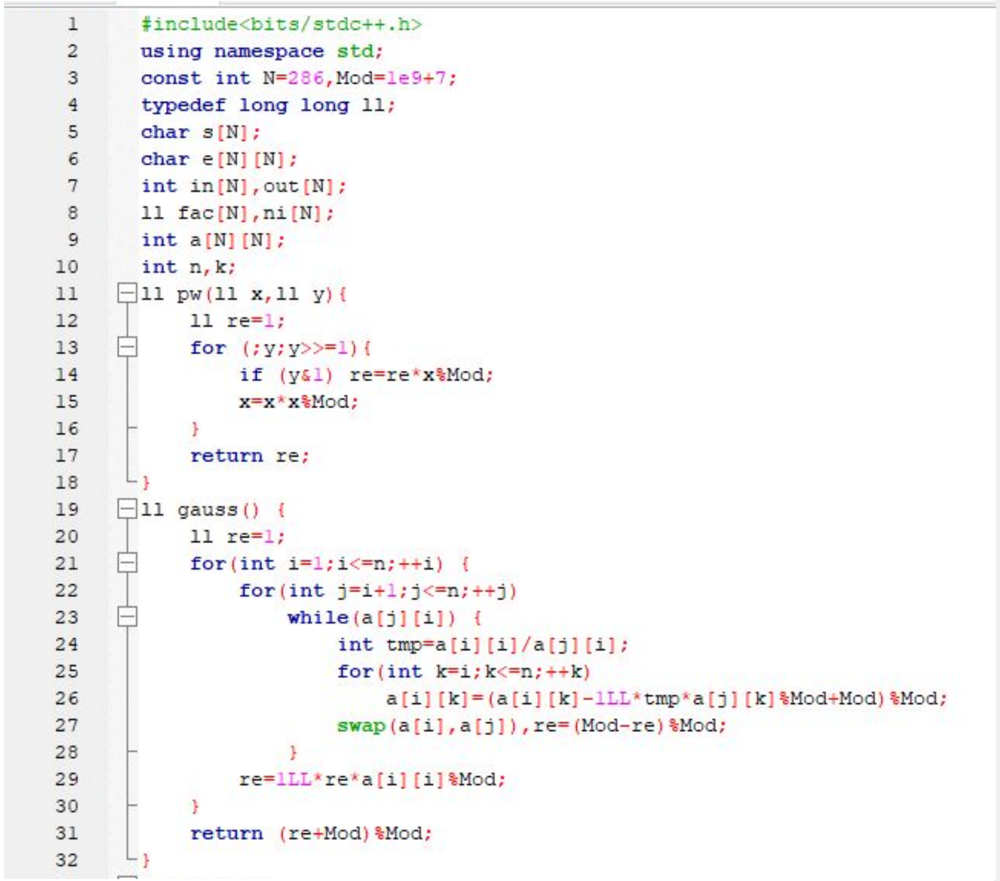
Windows Defender might be impacting your build and IDE performance. IntelliJ IDEA checked the following directories:
C:\Users\ding_\AppData\Local\JetBrains\IdeaIC2020.1
C:\JackProjects\CompetitiveProgramming\USACOGuide\PrimitiveGeometry

Fix... Actions ▾

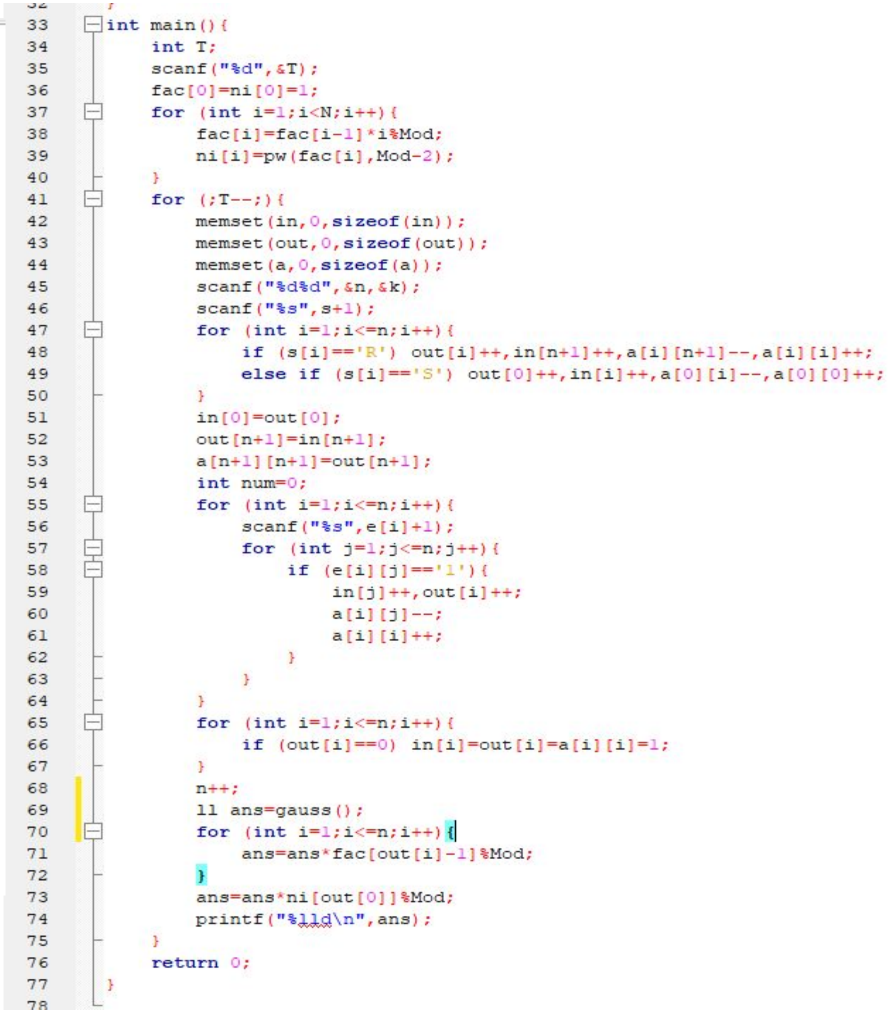
Build completed successfully in 2 s 122 ms (50 minutes ago)

124:18 CRLF UTF-8 4 spaces 311 of 6010M

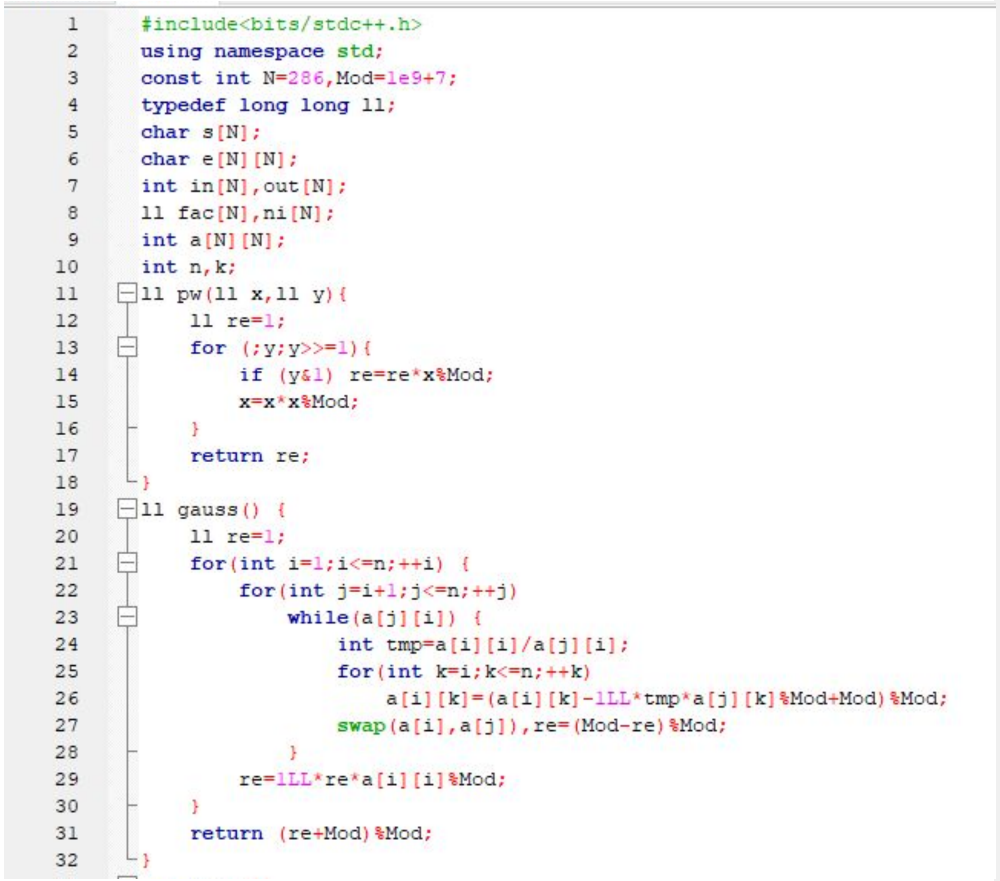
Demo



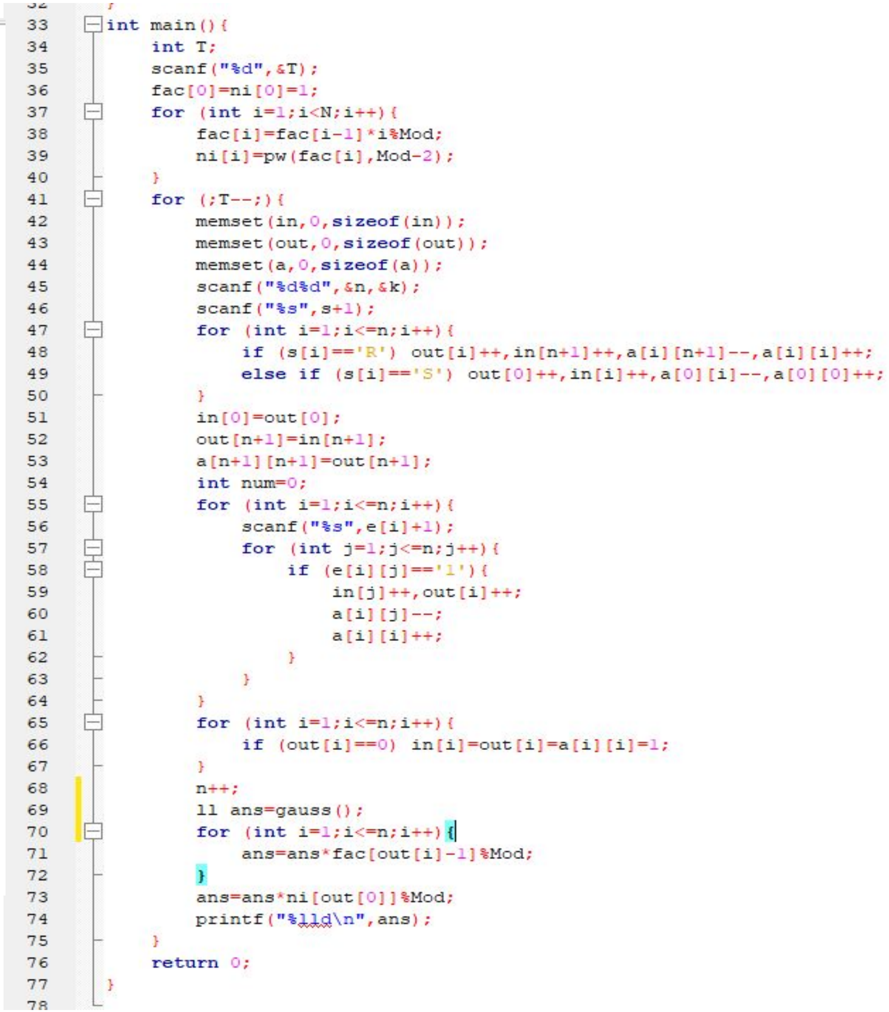
```cpp
#include<bits/stdc++.h>
using namespace std;
const int N=286,Mod=1e9+7;
typedef long long ll;
char s[N];
char e[N][N];
int in[N],out[N];
ll fac[N],ni[N];
int a[N][N];
int n,k;
ll pw(ll x,ll y){
    ll re=1;
    for (;y;y>>=1){
        if (y&1) re=re*x%Mod;
        x=x*x%Mod;
    }
    return re;
}
ll gauss() {
    ll re=1;
    for(int i=1;i<=n;++i) {
        for(int j=i+1;j<=n;++j)
            while(a[j][i]) {
                int tmp=a[i][i]/a[j][i];
                for(int k=i;k<=n;++k)
                    a[i][k]=(a[i][k]-1LL*tmp*a[j][k]%Mod+Mod)%Mod;
                swap(a[i],a[j]),re=(Mod-re)%Mod;
            }
        re=1LL*re*a[i][i]%Mod;
    }
    return (re+Mod)%Mod;
}
```



```cpp
int main(){
    int T;
    scanf("%d",&T);
    fac[0]=ni[0]=1;
    for (int i=1;i<N;i++){
        fac[i]=fac[i-1]*i%Mod;
        ni[i]=pw(fac[i],Mod-2);
    }
    for (;T--;){
        memset(in,0,sizeof(in));
        memset(out,0,sizeof(out));
        memset(a,0,sizeof(a));
        scanf("%d%d",&n,&k);
        scanf("%s",s+1);
        for (int i=1;i<=n;i++){
            if (s[i]=='R') out[i]++,in[n+1]++,a[i][n+1]--,a[i][i]++;
            else if (s[i]=='S') out[0]++,in[i]++,a[0][i]--,a[0][0]++;
        }
        in[0]=out[0];
        out[n+1]=in[n+1];
        a[n+1][n+1]=out[n+1];
        int num=0;
        for (int i=1;i<=n;i++){
            scanf("%s",e[i]+1);
            for (int j=1;j<=n;j++){
                if (e[i][j]=='1'){
                    in[j]++,out[i]++;
                    a[i][j]--;
                    a[i][i]++;
                }
            }
        }
        for (int i=1;i<=n;i++){
            if (out[i]==0) in[i]=out[i]=a[i][i]=1;
        }
        n++;
        ll ans=gauss();
        for (int i=1;i<=n;i++){
            ans=ans*fac[out[i]-1]%Mod;
        }
        ans=ans*ni[out[0]]%Mod;
        printf("%lld\n",ans);
    }
    return 0;
}
```



```cpp
#include<bits/stdc++.h>
using namespace std;
const int N=286,Mod=1e9+7;
typedef long long ll;
char s[N];
char e[N][N];
int in[N],out[N];
ll fac[N],ni[N];
int a[N][N];
int n,k;
ll pw(ll x,ll y){
    ll re=1;
    for (;y;y>>=1){
        if (y&1) re=re*x%Mod;
        x=x*x%Mod;
    }
    return re;
}
ll gauss() {
    ll re=1;
    for(int i=1;i<=n;++i) {
        for(int j=i+1;j<=n;++j)
            while(a[j][i]) {
                int tmp=a[i][i]/a[j][i];
                for(int k=i;k<=n;++k)
                    a[i][k]=(a[i][k]-1LL*tmp*a[j][k]%Mod+Mod)%Mod;
                swap(a[i],a[j]),re=(Mod-re)%Mod;
            }
        re=1LL*re*a[i][i]%Mod;
    }
    return (re+Mod)%Mod;
}
```



```cpp
int main(){
    int T;
    scanf("%d",&T);
    fac[0]=ni[0]=1;
    for (int i=1;i<N;i++){
        fac[i]=fac[i-1]*i%Mod;
        ni[i]=pw(fac[i],Mod-2);
    }
    for (;T--;){
        memset(in,0,sizeof(in));
        memset(out,0,sizeof(out));
        memset(a,0,sizeof(a));
        scanf("%d%d",&n,&k);
        scanf("%s",s+1);
        for (int i=1;i<=n;i++){
            if (s[i]=='R') out[i]++,in[n+1]++,a[i][n+1]--,a[i][i]++;
            else if (s[i]=='S') out[0]++,in[i]++,a[0][i]--,a[0][0]++;
        }
        in[0]=out[0];
        out[n+1]=in[n+1];
        a[n+1][n+1]=out[n+1];
        int num=0;
        for (int i=1;i<=n;i++){
            scanf("%s",e[i]+1);
            for (int j=1;j<=n;j++){
                if (e[i][j]=='1'){
                    in[j]++,out[i]++;
                    a[i][j]--;
                    a[i][i]++;
                }
            }
        }
        for (int i=1;i<=n;i++){
            if (out[i]==0) in[i]=out[i]=a[i][i]=1;
        }
        n++;
        ll ans=gauss();
        for (int i=1;i<=n;i++){
            ans=ans*fac[out[i]-1]%Mod;
        }
        ans=ans*ni[out[0]]%Mod;
        printf("%lld\n",ans);
    }
    return 0;
}
```