গুহা থেকে বেরিয়ে আজ আমি আমার অন্যতম প্রিয় বিষয়বস্তু কম্পিউটার প্রোগ্রামিং নিয়ে লিখা শুরু করে দিবো। টপিকটা হচ্ছে, একটা সংখ্যাকে এর n-তম ঘাতে তুলে ফেলা! খুবই সহজ কাজ! আমরা এই সহজ কাজটাকে প্রথমে আমাদের জন্য সহজ করে লিখবো,তারপর কম্পিউটারের জন্যে সহজ করে লিখবো।

১) আমাদের জন্য সহজ করেঃ

তোমার কাছে একটি সংখ্যা দেয়া আছে মনে করো “a”। তোমাকে ” a^n ” বের করতে হবে। সবচেয়ে সহজ উপায় কি হতে পারে?

উত্তরঃmath.h এ যে pow() ফাংশন আছে, তার ব্যবহার!

pow(a,n) লিখে ফেললেই কাজ শেষ। সমস্যা বাঁধে গিয়ে “Actual” মানের সাথে এই pow() ফাংশন থেকে জেনারেটেড রেজাল্ট এর Error নিয়ে। ধরো,তুমি 5^2 এই ফাংশনের সাহায্যে বের করলে। তুমি যদি এখন রেজাল্টটাকে কয়েক দশমিক ঘর পর্যন্ত প্রিন্ট করে দেখো,তাহলে মেশিনভেদে আউটপুট পাবে অনেকটা এমনঃ

25.000000000…..000001

যারা জানো না,pow() ফাংশনের রিটার্ন টাইপ কিন্তু ফ্লোট (float) !

তাই আমরা নিজেদের মত করে একটা ফাংশন বানিয়ে নিবো,যার রিটার্ন টাইপ হবে ইন্টিজার (integer).

ফাংশনটার প্রোটোটাইপঃ

১) ইনপুট হিসেবে নিবে **a** এবং **n**

২) **a** কে **n** বার নিজেদের মধ্যে একটা লুপ ঘুড়িয়ে গুণ করে দিবে।

৩) গুণফলটাকে রিটার্ন করে দিবে।

টাইম কমপ্লেক্সিটিঃ

যেহেতু লুপটা 1 থেকে n পর্যন্ত ঘুরছে,আর প্রতিটা গুণ ধরে নেয়া যায় কন্সট্যান্ট টাইমে হচ্ছে,তাই এই গুণ করার লুপটা **O(n)** সময় নিবে।

ফাংশন কোড (সি++) ঃ

```
int power(int a,int n)
{
    int res=1;
    while(n--)
    {
        res=res*a;
    }
    return res;
}
```

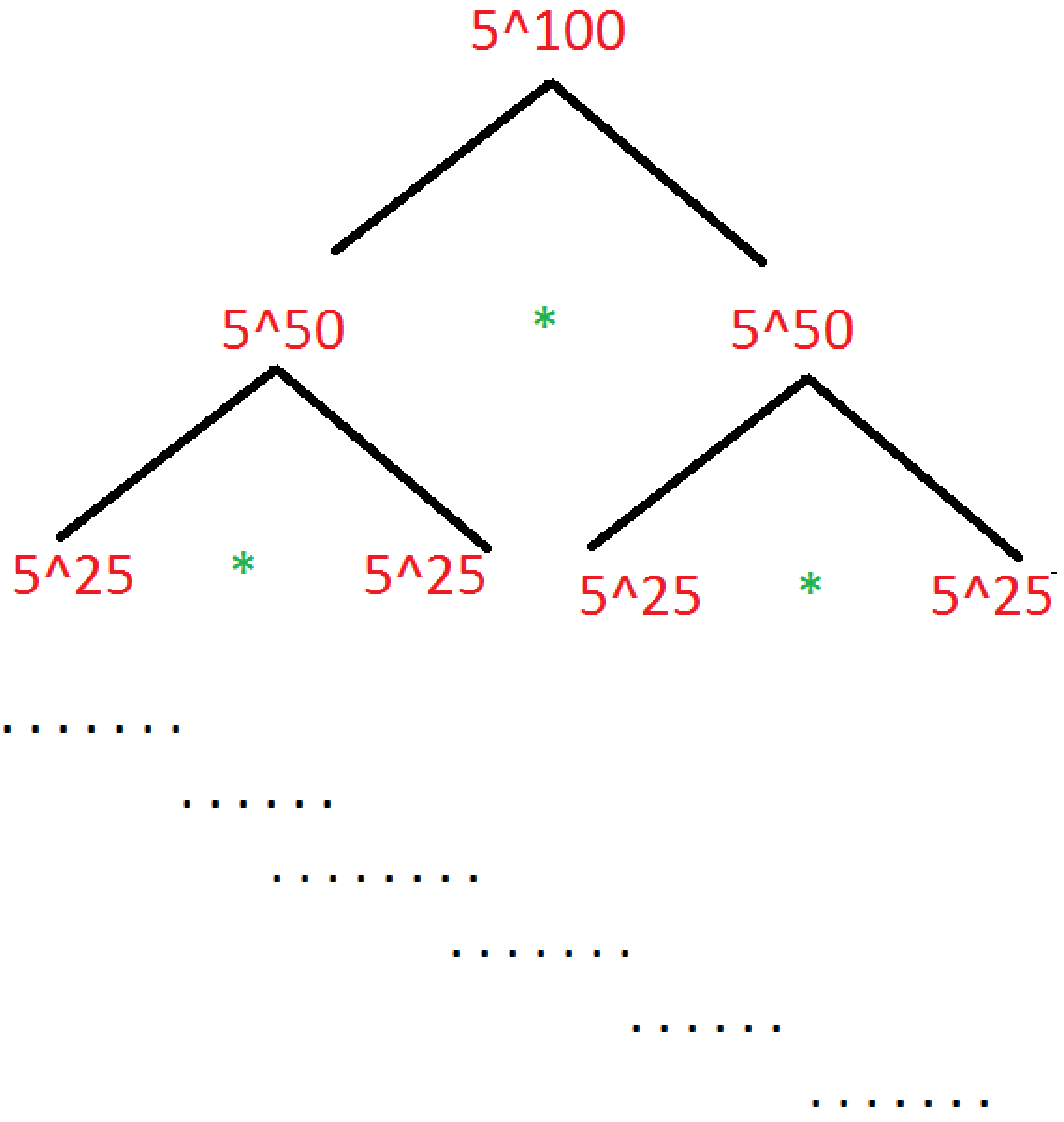
খুব সহজ কোড। চমৎকার ভাবে কাজও করবে।

২) কম্পিউটারের জন্য সহজ করেঃ

আমরা উপরের অংশে দেখলাম,আমাদের জন্য লিখতে সহজ একটা পাওয়ার ফাংশন,যার কার্যকারিতা যেমনই হোক না কেন,লিখাটা বেশ সোজা! তবে “লিখতে সোজা” কোড দিয়ে কি আর অপেক্ষাকৃত কঠিন বা সময়সাপেক্ষ কাজ করাটা বুদ্ধিমানের কাজ? অবশ্যই না! তাই আমরা এখন আমাদের জন্য “কিছুটা” কঠিন,কিন্তু Execute করতে কম্পিউটারের জন্য বেশ সহজ,সময় সাশ্রয়ী একটি পদ্ধতি দেখে আসবো,যার নাম এক্সপনেনসিয়েশন!

ধরে নাও,তোমাকে আমি 5^20 বের করতে বললাম। উপরের ফাংশনে 2০টা লুপ ঘুরে রেজাল্ট জেনারেট করে দিবে। কিন্তু 2০টা লুপ আমার কনটেস্টের জন্য অনেক কস্টলি(!)। কিভাবে এটাকে আরো দ্রুততম সময়ে করা যায়?

একটা ছোট্ট গ্রাফ/রিকারশন দেখে আসিঃ



ব্যাপারটা আসলে কঠিন কিছু না। আমাকে যদি বলা হয় 5^100 = ? আমি বলবো 5^100 = 5^50 * 5^50

আমাকে যদি বলা হয় “**5^50 এর মান তো x , তাহলে 5^100 এর মান কত?**”

আমি বলবো,”**আরে,এ তো সোজা! x * x ই হচ্ছে 5^100 “**

তার মানে দেখো,তুমি যদি 5^50 এর মান আগে থেকেই জানো,তোমার কি আর বেশি কিছু দরকার হচ্ছে? তুমি কি তোমার জানা মানটাই নিজেদের মাঝে একবার গুণ করে উত্তর বলে দিতে পারছো না?

এবার 5^50 = ?

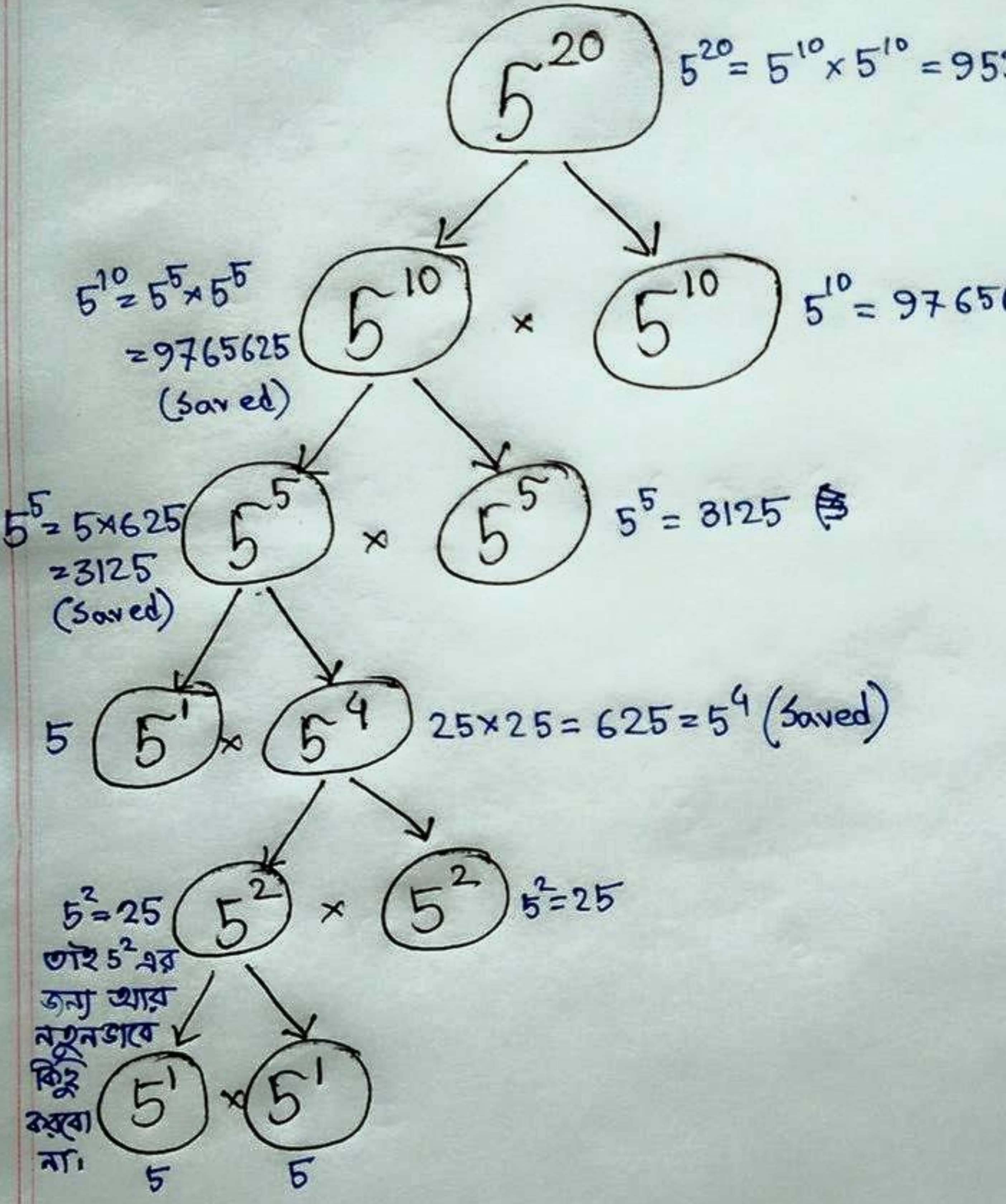
সোজা, 5^50 = 5^25 * 5^25

5^25 = ?

5^25 = 5 * (5^24) (কেননা,তুমি ২৫ কে সমান দুইভাগে ভাগ করতে পারবে না)

এভাবে যেতে যেতে একদম 5^1 পর্যন্ত চলে যাও,যার মান 5।

5^20 এর জন্যে আমাদের এই এলগরিদমটা কিভাবে রেজাল্ট জেনারেট করে,তা দেখে আসিঃ



$5^{20} = 5^{10} \times 5^{10} = 95$

$5^{10} = 5^{5} \times 5^{5}$
$= 9765625$
(Saved)

$5^{10} = 97\,65$

$5^{5} = 5 \times 625$
$= 3125$
(Saved)

$5^{5} = 3125$

5

$25 \times 25 = 625 = 5^{4}$ (Saved)

$5^{2} = 25$
তাই 5^{2} এর
জন্য আর
নতুনভাবে
কিছু
করবো
না।

$5^{2} = 25$

5

5

খেয়াল করে দেখো,এখানে হিসাবটা আগের চাইতে অনেক অনেক কমে গিয়েছে! একেবারে নিচে দেখো,5^1 হচ্ছে তোমার বেইজ কেইস! ওখান থেকে **একবার** বের করলে 5^2,সেখান থেকে **একবার** ক্যালকুলেট করা হলো 5^4 । সেখান থেকে 5^5 ক্যালকুলেট করা হলো **একবার ,** সেখান থেকে 5^10 ক্যালকুলেট করা হলো **একবার** । সেখান থেকে,5^20 ক্যালকুলেট করা হলো, **একবার।**
দেখো তো, “**একবার**” লিখাটা কয়বার আছে? ঠিকঠাকমত গুণে দেখলে উত্তরটা আসবে ” ৫ বার ”
আগের ফাংশনে এই কাজটা করতে কাজ করতে হত ২০ বার,আর এখানে ৫ বার। ২ ভিত্তিক log(20) = 4.32 বা প্রায় ৫
Fast enough,isn’t it?

ফাংশনটার প্রোটোটাইপঃ

১) ইনপুট নিবে **a** এবং **n**
২) **n** যদি জোড় হয়,তাহলে a^(n/2) ক্যালকুলেট করে,সেটাকে একটা ভ্যারিয়বলে স্টোর করবে। তারপর রিটার্ন করবে ভ্যারিয়বল*ভ্যারিয়বল।
৩) বিজোড় হলে রিটার্ন করবে a*a^(n-1)
৪) n==1 হলে , রিটার্ন করবে a.

টাইম কমপ্লেক্সিটিঃ

যেহেতু প্রতিটা ফাংশন থেকে দু’টো করে কাজ তৈরি হয়,কিন্তু এক্সিকিউট করতে হয় কেবল “একটা” , তাই প্রতি দু’টো কাজ থেকে একটি কাজ বাদ পরে যাচ্ছে। তাই টাইম কমপ্লেক্সিটি দাঁড়াচ্ছে O(log(n)). [এখানে লগের ভিত্তি ২]

ফাংশন কোড (সি++) ঃ

```
int power(int a,int n)
{
    if(n==1) return a;

    if(n%2==1)
    return a*power(a,n-1);

    int temp;
    temp=power(a,n/2);
    return temp*temp;
}
```

৩) কনটেস্ট্যান্টদের জন্য (ফাস্ট এক্সপনেনসিয়েশন):

রিকার্শন এক অদ্ভূত যাদুর নাম। কিভাবে কিভাবে যেন একটা ফাংশন লিখে ফেললে , ভিতর থেকে ঘুরিয়ে পেঁচিয়ে কাজ-টাজ করে আউটপুট দিয়ে দেয়! তবে এর একটা ড্র-ব্যাক হচ্ছে,এটা স্লো। হ্যা,লুপের চাইতে রিকার্শন বেশ স্লো। কারণ এটি কম্পিউটারের মেমরি থেকে একটা অংশ নিয়ে স্ট্যাক বানায়,সেটার ভেতর একটা একটা করে ফাংশন কল স্টোর করে বেইজ কেস পর্যন্ত এগিয়ে যায়,তারপর নিচ থেকে ফাংশনগুলো এক্সিকিউট করতে করতে উপরে উঠে আসে। এই যে অতিরিক্ত মেমরী টা লেগে যাচ্ছে,আমরা এই অংশে সেটাকে বাদ দিয়ে বোটম-আপ এপ্রোচে লুপ ঘুড়িয়ে a^n বের করবো। টাইম কমপ্লেক্সিটি থিওরিটিক্যালি সেইম,কিন্তু প্র্যাকটিক্যালি ডিফরেন্ট। মেমরী কমপ্লেক্সিটি সম্পর্কে আর নতুন করে বলে লাভ নেই!
দেখো, 5^20 কে আমরা আরেকটু অন্যভাবে চাইলেই লিখতে পারতামঃ

$$5^{20} = 5^{16} * 5^{4}$$

এখানে পার্থক্য আগেরটা থেকে একটাই যে,৫ এর উপর ঘাতগুলো সব **দুই এর ঘাত।**
২০ কে বাইনারী তে লিখলে হয়ঃ ১০১০০ (১৬+৪)
প্রথমে আমি ধরে নিই,আমাকে ৫ দেয়া আছে। এবং সাথে দেয়া আছে ঘাত হিসেবে ২০। আমাকে 5^20 বের করতে হবে। ২০ কে আবার বাইনারীতেও রূপান্তর করে ফেললাম।

Execution:

বাইনারী সংখ্যাটার বাম দিক থেকে প্রথম অংকটা ১ , তাহলে আমরা বের করবো (5^0)*(5^0)=1 এবং এর সাথে গুণ করে দিবো 5 । তাহলে আমরা এই স্টেপে পাচ্ছি = 5^1 [ধরে নিচ্ছি,5^0 আমরা শুরু থেকে কোথাও স্টোর করে রেখেছি]*
বাইনারী সংখ্যাটার বাম দিক থেকে দ্বিতীয় অংকটা ০, তাহলে আমরা বের করবো (5^1)(5^1)= 5^2
বাইনারী সংখ্যাটার বাম দিক থেকে তৃতীয় অংকটা ১, তাহলে আমরা বের করবো (5^2)*(5^2)=5^4 এবং এর সাথে গুণ করে দিবো 5। তাহলে আমরা এই স্টেপে পাচ্ছি = 5^5
বাইনারী সংখ্যাটার বাম দিক থেকে চতুর্থ অংকটা ০, তাহলে আমরা বের করবো (5^5)*(5^5)= 5^10
বাইনারী সংখ্যাটার বাম দিক থেকে পঞ্চম অংকটা ০, তাহলে আমরা বের করবো (5^10)*(5^10)= 5^20

কাজ শেষ!

মূল সুবিধাটা এখানেই,অতিরিক্ত কোনো মেমরী খরচ হচ্ছেনা। একটি log(n) সাইজের লুপ দিয়েই কাজটা করে ফেলা যাচ্ছে। কনটেস্টে এটা বেশ কাজে দেয়,যখন বেশ বড় ভ্যালুর জন্য তোমাকে a^n বের করতে হয়। তুমি mod করে করে এগিয়ে গিয়েও হয়তো সবসময় পার পাবেনা,মেমরী কমপ্লেক্সিটির প্যাচে ফেলে তোমার নিশ্চিত Accepted কোডটিকেও Memory Limit Exceeded এ পরিণত করে দিতে পারে।

ফাংশন কোডঃ

```
int power(int a,int n)
{
    string bin="";
    int temp;
    while(n>0)
    {
        temp=n%2;
        if(temp==0) bin+="0";
        else bin+="1";
        n/=2;
    }
    reverse(bin.begin(),bin.end());
    // bin নামক স্ট্রিং এ এখন আমাদের n এর বাইনারী ফর্মটি সেইভড!

    int res=1; // এই সেই 5^0,যা আমি উদাহরণে ব্যবহার করেছিলাম

    for(int i=0; i<bin.size(); i++)
```

```
    {
        res=res*res;
        if(bin[i]=='1') res=a*res;
    }
    return res;
}
```

শুধুমাত্র পদ্ধতি ৩ অবলম্বন করে আমি একটা প্রব্লেমে 0.020 সেকেন্ড রানটাইম থেকে,0.000 সেকেন্ডে নামিয়ে নিয়ে এসেছিলাম। তাই এর গুরুত্ব নেহায়েত কম নয়!
তো,জানাবে কেমন লাগলো। সবশেষে ধন্যবাদ জানাতে চাই আমার ডিপার্টমেন্ট এর শ্রদ্ধেয় লেকচারার আন্না ফারিহা ম্যাডামকে। উনার একটি লিখা পড়েই আমি অনুপ্রাণিত হয়ে এই পোস্টটি লিখেছি। লিখায় ভুল থাকলে কিংবা বুঝতে অসুবিধা হলে,কমেন্ট বক্সে জানাতে ভুলো না/ভুলবেন না। আজকের মতো এটুকুই।

আগামী পর্বে থাকছে অ্যারে মাল্টিপ্লিকেশন উইদ মডুলার এক্সপনেন্সিয়েশন,রিকারেন্স ইকুয়েশন সল্‌ভ ইন O(log(n)) টাইম!