অ্যালগরিদম - Algorithms সংখ্যাতত্ত্ব - Number theory

সংখ্যাতত্ত্ব: মৌলিক সংখ্যা- সিভ অফ এরাটোস্থেনিস

Prime number algorithm Bangla tutorial/ Sieve of Eratosthenes Bangla Tutorial

Sharif Hasan • September 13, 2021 সর্বশেষ আপডেট September 13, 2021 1 8,595

পড়তে 4 মিনিট লাগতে পারে

মৌলিক সংখ্যা বা Prime Number আসলে কি ? মৌলিক সংখ্যা হলো সেসব সংখ্যা যারা ১ থেকে বড় পূর্ণসংখ্যা এবং ১ ও ঐ সংখ্যা বাদে অন্যকোন সংখ্যাদ্বারা নিঃশেষে বিভাজ্য নয়। এখন যদি বলা হয় N যেকোনো একটি সংখ্যা। আপনাকে বলা হলো সংখাটি মৌলিক কিনা তা বের করে দিতে।

এই লিখার পুরনো ভার্সন এখানে।

এই সমস্যার সবথেকে সহজ সমাধান হলো ২ থেকে $n-1$ পর্যন্ত একটি লুপ চালাবো। প্রতিবার লুপ কন্ট্রোল ভেরিয়েবল i দিয়ে N কে ভাগ দিতে থাকবো। যদি কখনো i দিয়ে n কে ভাগ করা যায় তবে বলতে পারি n মৌলিক সংখ্যা নয়। যদি ভাগ না যায় তবে লুপের শেষে বলতে পারবো n মৌলিক সংখ্যা। নিচের কোডের মত।

প্রাইম ফ্যাক্টরাইজেশন নিয়ে জানতে এটা দেখুনঃ সংখ্যাতত্ত্ব: মৌলিক সংখ্যা (prime number) ও তার অ্যালগরিদম

```
bool is_prime(int n){
 if(n<2){
 return 0;
 }
        if(n==2){
                return 1;
        }
 for(int i=2;i<n;i++){
 if(n%i==0){
 return 0;
 }
 }
 return 1;
}
```

সহজ কোড। এই কোডের কমপ্লেক্সিটি কত? এখানে দেখা যাচ্ছে এখানে লুপ ঘুরবে 2 থেকে $n-1$ পর্যন্ত। এখানে কমপ্লেক্সিটি দাঁড়াচ্ছে $O(n)$। তারমানে $n = 9999999967$ হলে আমাদের লুপ এই কাজটি করতে $9999999967 - 2$ বার ঘুরবে। বুঝতেই পারছি ১/২ সেকেন্ডে এই কোড এত বড় সংখ্যা মৌলিক না যৌগিক তা বের করতে পারবে না।

$O(\sqrt{n})$ এ একটি সংখ্যা মৌলিক কি না তা বের করা

উপরের $O(n)$ সমাধানটিকে আমরা খুব সহজেই $O(\sqrt{n})$ এর রূপান্তর করে দিতে পারি। এর জন্য আমাদের একটি বিষয় নিয়ে পরিস্কার ধারনা অর্জন করতে হবে।

যদি কোন সংখ্যা n এর একটি গুণনীয়ক d হয়, তাহলে $\frac{n}{d}$ এর আরেকটি গুণনীয়ক। $d, \frac{n}{d}$ এদের মধ্যে ছোটটি অবশ্যই $\sqrt{n}$ এর সমান বা ছোট।

এখন আমরা দেখতে পাচ্ছি, d এর $\frac{n}{d}$ এর গুনফল n। এবার ধরে নেই $a = d$ এবং $b = \frac{n}{d}$।

যদি আমরা উপরের সিদ্ধান্তটি না মানি এবং ধরে নিই $a > \sqrt{n}$ এবং $b > \sqrt{n}$। তবে, $a \times b > n$ হবে। কারণ আমরা জানি $\sqrt{n} \times \sqrt{n} = n$। যেহেতু a, b কে $\sqrt{n}$ এর থেকে বড় ধরে নিয়েছি, তাই $a \times b = n$ হতে পারে না। কিন্তু $a \times b = d \times \frac{n}{d} = n$ হবার কথা। সুতরাং আমরা বলতে পারি এদের মধ্যে ছোটটির অবশ্যই $\sqrt{n}$ এর সমান বা ছোট হতে হবে।

উদাহরন সরূপ $n = 64$ হলে, গুণনীয়ক জোড় $(d, \frac{n}{d}) = (1, 64), (2, 32), (4, 16), (8, 8)$

সুতরাং আমরা দেখতে পাচ্ছি আমরা অতদূর n পর্যন্ত লুপ না চালিয়ে $\sqrt{n}$ পর্যন্ত লুপ চালালেই আমরা ১ বাদে সর্বোনিম্ন একটা হলেও গুণনীয়ক পাবো যদি n একটি যৌগিক সংখ্যা হয়। উপরের কোডের লুপের শর্তে একটু পরিবর্তন করলেই হবে।

```
bool is_prime(long long n){
 if(n==1||n==0) return 0;
 for(int i=2;i<=sqrt(n);i++){
 if(n%i==0){
 return 0;
 }
 }
 return 1;
}
```

সিভ অফ এরাটোস্থেনিস/ sieve of Eratosthenes Bangla tutorial

এখন ধরা যাক আমাদের সমস্যা আরেকটু পরিবর্তন করা হলো। এখন আপনাকে ১ থেকে n পর্যন্ত সকল মৌলিক সংখ্যা খুজতে বলা হয়েছে।

আমরা যদি নিচের কোডের মত করে করার চেষ্টা করি তাহলে কেমন হয়?

```
void sieve(int n){
 for(int i=1;i<=n;i++){
 if(is_prime(i)){
 printf("%d is prime\n",i);
 }
 }
}
```

আমরা i=1 থেকে i=n পর্যন্ত লুপ চালিয়েছি এবং i মৌলিক সংখ্যা হলে আমরা i কে মৌলিক হিসেবে প্রিন্ট করেছি। এখানে is_prime() এর কমপ্লিক্সিটি উপর থেকে দেখেছি $O(\sqrt{n})$। তো এই কোডটি যথেষ্ট দ্রুত নয়। আমরা সিভ অফ এরাটোস্থোনিস এর মাধ্যমে এর কম্প্লক্সিটি $O(nlog(log(n)))$ এ নামাতে পারি।

এখন আমরা যা করবো তা হলো একটি সংখ্যা i নিবো এবং তার যত গুনিতক আছে তাদেরকে যৌগিক সংখ্যা হিসেবে চিহ্নিত করবো। চিহ্নিত করার জন্য আমরা একটি অ্যারের সহায়তা নিবো, যেখানে ১ থেকে n পর্যন্ত সংখাগুলো থাকবে এবং শুরুতে সবাই মৌলিক হিসেবে বিবেচ্য হবে। পরে লুপ ঘুরানোর সময় যৌগিক গুলো আলাদা করা হবে।

```
void sieve(int n){
 bool mark[n+1];
 for(int i=0;i<n+1;i++){
 mark[i]=1; //mark এর ভেতর সকল সংখ্যাকে মৌলিক বলে দিলাম।
 }
 for(int i=1;i<=n;i++){
 /*
               @ i যদি মৌলিক হয় তবে
 @ i এর যত গুণনীয়ক আছে n এর সমান বা ছোট সবাইকে যৌগিক সংখ্যা হিসেবে
 @ চিহ্নিত করে দিতে হবে।
 */
 }
}
```

এখানে শুরুতে আমরা mark নামের একটি অ্যারে নিয়েছি। যেখানে প্রতিটি ইলিমেন্টের মান 1 দিয়েছি যা দিয়ে বুঝিয়েছি ঐ ইন্ডেক্সটি একটি মৌলিক সংখ্যা। কিন্তু আমরা জানি এটা সত্য নয়। তাই আমরা পরবর্তীতে আরেকটি লুপ চালিয়েছি। এখানে i যদি মৌলিক হয় তবে i এর প্রতিটি গুণিতককে (j) যৌগিক হিসেবে চিহ্নিত করবো। অর্থাৎ $mark[j] = 0$ করে দিবো।

```
void sieve(int n){
 bool mark[n+1];
 for(int i=0;i<n+1;i++){
 mark[i]=1; //mark এর ভেতর সকল সংখ্যাকে মৌলিক বলে দিলাম।
 }
 for(int i=2;i<=n;i++){
 if(mark[i]==1){ // তারমানে i মৌলিক সংখ্যা
 for(int j=2*i;j<=n;j+=i){ // কারণ i এর পরের i এর গুণিতক 2i, তারপর 3i বা 2i
 mark[j]=0; // i  এর সব গুণিতক যৌগিক সংখ্যা
 }
```

```
  }
  }
}
```

এখানে mark[i]==1 তখনি সত্য যখন i একটি মৌলিক সংখ্যা। পরের লাইনে আমরা i এর পরবর্তী গুণিতক $j = 2i$ থেকে শুরু করেছি। এভাবে $j = 3i, j = 4i$ করে বাড়তে থাকবে $(2i = i + i, 3i = 2i + i, 4i = 3i + i)$। এই $2i, 3i, 4i$ কেউই মৌলিক সংখ্যা নয়। তাই আমরা mark[j]=0 করে দিয়েছি।

$n = 10$ এর জন্য একটি সিমুলেশন করি।

i=2	1	2	3	4	5	6	7	8	9	10
	0	1	1	0	1	0	1	0	1	0

যখন $i = 2$ তখন mark অ্যারেতে $j = 2 * i = 4$,
$j = 4 + 2 = 6, 6 + 2 = 8, 8 + 2 = 10$ এদের মার্ক করা হয়ে যাবে (লাল রঙ দেয়া)। দেখতে পাচ্ছি mark অ্যারে এর 3 নং ইনডেক্স মার্ক করা হয় নি। অর্থাৎ আমরা বলতে পারি ৩ এর ১, ৩ বাদে কোন গুণনীয়ক নেই। তাই ৩ মার্ক হয় নি, অর্থাৎ ৩ একটি যৌগিক সংখ্যা।

i=3	1	2	3	4	5	6	7	8	9	10
	0	1	1	0	1	0	1	0	0	0

যখন $i = 3$ তখন ৩ এর জেসব গুণিতক রয়েছে 6,9 এরা কাটা যাবে (নীল রঙ করা)। কিন্তু দেখতে পারছি ৬ ইতোমধ্যে কাটা গিয়েছে ২ দ্বারা কারণ 3×3 বা 9 এর আগে ৩ এর যত গুণিতক আছে তারা অবশ্যই ৩ এর থেকে ছোট কোন সংখ্যারও গুণিতক (এটা খেয়াল রাখতে হবে, এই কনসেপ্ট এ আমরা কোড অপ্টিমাইজ করবো)। উদাহরণ, ২০ কিন্তু ৫ এর গুণিতক। আবার $4 \times 5 = 20$ এখানে $20 < 5 \times 5 => 20 < 25$ তাই 20 এর একটি গুণনীয়ক ৫ এর থেকে ছোট।

নিচের চিত্রটি প্রসেসটিকে বর্ণনা করছে।

সাধারণ ভাবে বলতে পারি $i \times i$ এর আগে i এর যত গুণিতক রয়েছে তা অবশ্যই i এর থেকে ছোট (অবশ্যই ১ হিসেবের বাইরে) কোন একটি সংখ্যারও গুনিতক। তাহলে বলা যাচ্ছে i এর এধরনের গুণিতক আগেই কেটে যাবে i এর থেকে কোন সংখ্যা দ্বারা।

তাই আমরা $j = i.i$ থেকে শুরু করলেই পারি। সুতরাং আমাদের কোড নিচের মত হবে।

```
void sieve(int n){
 bool mark[n+1];
 for(int i=0;i<n+1;i++){
 mark[i]=1; //mark এর ভেতর সকল সংখ্যাকে মৌলিক বলে দিলাম।
 }
 for(int i=2;i<=n;i++){
 if(mark[i]==1){ // তারমানে i মৌলিক সংখ্যা
 for(int j=i*i;j<=n;j+=i){
 mark[j]=0; // i  এর সব গুণিতক যৌগিক সংখ্যা
 }
 }
 }
}
```

আরেকটু উন্নয়ন করবো আমরা। উপরে আমরা দেখে এসেছি একটি সংখ্যা n এর জন্য $\sqrt{n}$ পর্যন্ত লুপ চালালেই আমরা পরীক্ষা করতে পারি সংখ্যাটি মৌলিক কিনা। এই কোডে প্রথম লুপে আমরা যদি $\sqrt{n}$ পর্যন্ত লুপ চালাই তাহলেই কিন্তু হয়। কারণ $2 - \sqrt{n}$ এর মধ্যে i এর যেই মান গুলো পাবো এগুলো অবশ্যই 1 থেকে n পর্যন্ত সংখাগুলোর গুণনীয়ক হবে যদি সংখ্যাটি যৌগিক হয়।

```
void sieve(int n){
 bool mark[n+1];
 for(int i=0;i<n+1;i++){
 mark[i]=1; //mark এর ভেতর সকল সংখ্যাকে মৌলিক বলে দিলাম।
 }
 for(int i=2;i*i<=n;i++){ // i<=sqrt(n) হলে i*i<=n [বর্গমূল পর্যন্ত লুপ চালিয়েছি]
 if(mark[i]==1){ // তারমানে i মৌলিক সংখ্যা
 for(int j=i*i;j<=n;j+=i){
 mark[j]=0; // i  এর সব গুণিতক যৌগিক সংখ্যা
 }
```

```
11  }
12  }
13         for(int i=0;i<=n;i++){
14  if(mark[i]){ // মৌলিক সংখ্যা i এর  ইনডেক্সটি 1 ইয়ে মার্ক করা।
15      cout<<i<<", ";
16  }
17  }
18 }
```

আমাদের কোড করা শেষ। এখন আমরা mark অ্যারের ভেতরে ১ থেকে n পর্যন্ত সমস্ত মৌলিক সংখ্যা কে পেয়ে গিয়েছি। ১৩ থেকে ১৭ নং লাইনে প্রিন্ট করেছি।

এই লিখাটির পুরনো ভার্সন রয়েছে। সেখানে কিছু ভুল থাকার কারণে এটি নতুন করে লিখা হলো।

লেখাটি কেমন লেগেছে আপনার?

রেটিং দিতে হার্টের উপর ক্লিক করুন।

গড় রেটিং 4.4 / 5. মোট ভোট: 34

#Number theory **#Prime number**

১টি মন্তব্য

gUeSS

September 28, 2021 at 11:02 PM

Awesome vaiya

Reply