সংখ্যাতত্ত্ব - Number theory

সংখ্যাতত্ত্ব: অয়লার টোশেন্ট ফাংশন/ ফাই ফাংশন

Euler's totient function Bangla tutorial/ Phi function Bangla tutorial/ ফাই ফাংশন টিউটোরিয়াল

Sharif Hasan • September 5, 2021 সর্বশেষ আপডেট April 20, 2022 0 714 পড়তে 6 মিনিট লাগতে পারে

অয়লার টোশেন্ট ফাংশন (Euler's Totient Function) যা ফাই ফাংশন (Phi function) (এর প্রতীক φ) হিসেবেও পরিচিত, একটি সংখ্যা n এর 1 থেকে n পর্যন্ত কত গুলো সহমৌলিক বা Co-Prime আছে তা গননা করতে ব্যবহার করা হয়। দুইটি সংখ্যাকে আমরা তখনি সহমৌলিক বলি যখন এই দুইটি সংখ্যার গসাগু ১ হয়। অন্য কথায় ১ ব্যাতিত সংখ্যাদুটির মধ্যে কোন সাধারণ গুনিতক না থাকে।

অয়লার টোশেন্ট ফাংশন: সমস্যার বিবরণ:

একটি সংখ্যা n দেয়া আছে। আমাদের বের করতে হবে 1 থেকে n পর্যন্ত কতগুলি সংখ্যা আছে যা n এর সাথে সহমৌলিক।

Euler's totient function/ phi function Bangla Tutorial

সমাধান পদ্ধতি ১

আমরা যদি খুব সহজে এটা সমাধান করতে চাই তবে আমরা এক কাজ করতে পারি। 1 থেকে n পর্যন্ত লুপ চালিয়ে দেখবো কোন কোন সংখ্যার সাথে n এর গসাগু 1 হয়। সেই সংখ্যাগুলো গননা করে আউটপুট দিবো। এর কোডটি নিচের মতো।

```
#include<stdio.h>
int gcd(int a,int b){
 if(b==0) return a;
 gcd(b,a%b);
}
int main()
{
 int n,cnt=0;
 scanf("%d",&n);
 for(int i=1;i<=n;i++){ //১ থেকে n পর্যন্ত লুপ
 if(gcd(i,n)==1) cnt++; // যদি i,n এর gcd ১ হয় তবে গননা করবো।
 }
 printf("%d",cnt);
 return 0;
}
```

এই পদ্ধতির টাইম কমপ্লেক্সিটি $O(n \log n)$। কারণ প্রথমে লুপ ঘুরবে n বার। লুপের প্রতিধাপে গসাগু হিসেব করা হয়েছে ইউক্লিডীয়ান ভাগের নিয়ম অনুসারে। এতে সময় লাগবে $\log n$। সুতরাং মোট টাইম কমপ্লিক্সিটি $O(n \log n)$

সমাধান পদ্ধতি ২

আমরা সমস্যাটি সমাধান করার জন্য অয়লার টোশেন্ট ফাংশন বা Phi ফাংশন এর সহায়তা নিবো। Phi ফাংশন হলো এমন একটি ফাংশন যাকে n ইনপুট দেয়া হলে ফাংশনটি $O(\sqrt{n})$ সময়ে 1 থেকে n পর্যন্ত মোট সহগুনকের সংখ্যা আউটপুট দিবে। এই ফাংশনের বেশ কয়েকটি প্রোপার্টি আছে। এইগুলো বুঝে নিলেই আমরা খুব সহজে অ্যালগরিদমটি বুঝতে পারবো।

অয়লার টোশেন্ট ফাংশন: প্রোপার্টি ১

যদি **p** একটি মৌলিক সংখ্যা হয় তবে $\varphi(p) = p - 1$। এটা বুঝার জন্য একটি সংখ্য ৭ ধরে নিই। যেহেতু ৭ একটি মৌলিক সংখ্যা, এবং মৌলিক সংখ্যাকে ১ এবং ঐ সংখ্যা ব্যাতিত অন্য কোন সংখ্যা দ্বারা ভাগ করা যায় না, তাই আমরা বলতে পারি ১ থেকে ৭ পর্যন্ত শুধু ১ এবং ৭ দ্বারা ৭ কে ভাগ যাবে।

এর মধ্যে শুধু ৭ এবং ৭ এর গসাগু ৭। বলতে পারি ৭ ব্যাতিত ১ থেকে ৭ পর্যন্ত বাকি সবার সাথে ৭ এর গসাগু ১। এমন সংখ্যা আছে মোট ৭-১=৬ টি। তাই $\varphi(7) = 6$।

অয়লার টোশেন্ট ফাংশন: প্রোপার্টি ২

যদি p একটি মৌলিক সংখ্যা হয় এবং a যেকোন একটি ধনাত্বক পূর্নসংখ্যা হয়, তবে $\varphi(p^a) = p^a.(1 - \frac{1}{p})$।

আমাদের বের করতে হবে 1 থেকে p^a পর্যন্ত কতগুলি সংখ্য আছে যাদের সাথে p^a এর গসাগু p অথবা এর থেকে বড় হবে। আমরা একটা বিষয় বুঝতে পারছি, p^a এর আগে যেসব সংখ্যার সথে এর গসাগু 1 এর বড়, সবার সাথেই p^a এর গসাগু অন্তত p হবে। হিসাব করলে

ধরা যাক p=3, a=2। এখন বের করা লগবে ১ থেকে ৯ পর্যন্ত কোন সংখ্যগুলোর সাথে ৯ এর গসাগু অন্তত p হবে।

$1, 2, 3, 4, 5, 6, 7, 8, 9$ এই ধারাটিকে আমরা নিচের মতো করে লিখতে পারি,

$1, 2, 3.1 + 0, 3.1 + 1, 3.1 + 2, 3.2 + 0, 3.2 + 1, 3.2 + 2, 3.3 + 0$
দেখা যাচ্ছে, $3.1 = 3, 3.2 = 6, 3.3 = 9$ এই ৩টি সংখ্যাই আছে যারা 3.3 বা 9

এর সহগুনক নয়। কারণ এদের সাথে 9 এর গসাগু সর্বোনিম্ন 3। এভাবে যেকোন p^a এর জন্য দেখানে যায় 1 থেকে p^a পর্যন্ত $\frac{p^a}{p}$ সংখ্যক সংখ্যা আছে যা p^a এর সহগুনক নয়। সুতরাং সহগুনক হলো, $p^a - \frac{p^a}{p}$ বা $p^a.(1-\frac{1}{p})$ টি সংখ্যা।

অয়লার টোশেন্ট ফাংশন: প্রোপার্টি ৩

Phi ফাংশন একটি মাল্টিপ্লিকেটিভ ফাংশন। যদি m এবং n দুটি সহমৌলিক সংখা হয়, তবে $\varphi(m.n) = \varphi(m).\varphi(n)$।

মূলত প্রোপার্টি ২ এবং ৩ ই আমাদের কাজে লাগবে। কারন প্রোপার্টি ১ এর $\varphi(p)$ কে লিখতে পারি, $\varphi(p^1)$ যেখানে $a = 1$, যা ২ নং প্রোপার্টিকে উপস্থাপন করে।

এখন আমরা জানি, সকল সংখ্যাকে আমরা কিছুসংখ্যক মৌলিক সংখ্যার গুনফল আকারে লিখতে পারি। যেমন, ১২ কে লিখতে পারবে $2 \times 2 \times 3 = 2^2 \times 3^1$

সুতরাং কোন সংখ্য n কে আমরা লিখতে পারবে, $n = p_1^{a_1} \times p_2^{a_2} \times \ldots \times p_k^{a_k}$

এই সমস্যা সমাধান করতে হলে আমাদের প্রথমে সংখ্যাটিকে মৌলিক গুনণীয়কে বিশ্লেষণ করে নিতে হবে। এবং ঐ মৌলিক সংখ্যা কতগুলি আছে তা গণনা করে রাখতে হবে।

অতঃপর $\varphi(n) = \varphi(p_1^{a_1} \times p_2^{a_2} \times \ldots \times p_k^{a_k})$ সমীকরণে প্রোপার্টি ৩ প্রোয়োগ করে সমাধান করে ফেলবো ।

$$\varphi(n) = \varphi(p_1^{a_1}.p_2^{a_2}\ldots p_k^{a_k})$$

বা, $\varphi(n) = \varphi(p_1^{a_1}).\varphi(p_2^{a_2})\ldots\varphi(p_k^{a_k})$ [প্রোপার্টি ৩ অনুসারে]

বা, $\varphi(n) = (p_1^{a_1} - \frac{p_1^{a_1}}{p_1}).(p_2^{a_2} - \frac{p_2^{a_2}}{p_2})\ldots.(p_k^{a_k} - \frac{p_k^{a_k}}{p_k})$ [প্রোপার্টি ২ অনুসারে]

বা, $\varphi(n) = p_1^{a_1}(1-\frac{1}{p_1}).p_2^{a_2}.(1-\frac{1}{p_2})\ldots.p_k^{a_k}.(1-\frac{1}{p_k})$

বা, $\varphi(n) = p_1^{a_1}.p_2^{a_2}.p_k^{a_k}.(1-\frac{1}{p_1}).(1-\frac{1}{p_2})\ldots.(1-\frac{1}{p_k})$

বা, $\varphi(n) = n.(1-\frac{1}{p_1}).(1-\frac{1}{p_2})\ldots.(1-\frac{1}{p_k})$ [$p_1^{a_1}.p_2^{a_2}.p_k^{a_k} = n$]

এক নজরে প্রাইম ফ্যাক্টরাইজেশন

আগেই বলেছি আমরা কোন সংখ্যা n কে তার মৌলিক গুনণীয়কের গুনফল আকারে প্রকাশ করতে পারি। $n = p_1^{a_1} \times p_2^{a_2} \times \ldots \times p_k^{a_k}$, যেমন $12 = 2^2.3^1$

আমরা যা করবো, i=2 থেকে i=sqrt(n) পর্যন্ত লুপ চালাবো, এবং i, n কে নিঃশেষে ভাগ করলে n=n/i করে প্রতিবার n কে আপডেট করবো। এমন করে ভাগ করলে আমরা n এর সকল মৌলিক গুনণীয়ক i এর মাধ্যমে পেয়ে যাবো। আরো জানতে এই লিখাটি পড়ুন সংখ্যাতত্ত্ব: মৌলিক সংখ্যা (prime number) ও তার অ্যালগরিদম।

প্রাইম ফ্যাক্টরাইজেশন এর মাধ্যমে অয়লার টোশেন্ট ফাংশন ইমপ্লিমেন্টেশন

$\varphi(n) = n.(1 - \frac{1}{p_1}).(1 - \frac{1}{p_2}) \ldots .(1 - \frac{1}{p_k})$ এই সুত্র আমরা উপর থেকে পেয়েছি। এখন আমাদের কাজ হলো এই সুত্রটি ইমপ্লিমেন্ট করা।

```
int phi(int n){
 float result=n;
 for(int i=2;i*i<=n;i++){ // i<=sqrt(n) বা   i*i<=n পর্যন্ত লুপ চলবে।
 if(n%i==0){ // যদি i, n এর একটি বিভাজক হয়।
 while(n%i==0){ // যতক্ষন পর্যন্ত n কে iদ্বারা ভাগ যাবে
```

```
6  n=n/i; // n থেকে i কেটে দিতে থাকবো।
7  }
8  result=result*(1-1.0/i); // সূত্র অনুযায়ী গুন করেছি।
9  }
10 }
11 // sqrt(n) এর থেকে বড় মৌলিক গুননীয়ক
12 //  একটি ই থাকা সম্ভব। যদি n>1 হয় তবে n এর বর্তমান মান ই তাকে উপস্থাপণ করে।
13 if(n>1){
14 result=result*(1-1.0/n); // সূত্র
15 }
16 return result;
17 }
```

এই ফাংশনকে একটি সংখ্যা n ইনপুট দিলে ফাংশনটি ১ থেকে n পর্যন্ত এর মোট সহগুনক এর সংখ্যা রিটার্ন করবে। এখানে সবার বাহিরের যেই লুপটি ৩ নং লাইনে, এই লুপটি $i.i = n => i^2 = n => i = \sqrt{n}$ বা $\sqrt{n}$ বার ঘুরবে। কারণ আমরা জানি কনো সংখ্যার কেবলমাত্র একটি মৌলিক গুণনীয়ক $\sqrt{n}$ এর বড় হতে পারে।

এর পর ৪ নং লাইনে, যদি i দ্বারা n কে নিঃশেষে ভাগ যায় তবে আমরা ৫ থেকে ৭ নং লাইনের while লুপ এ n থেকে সমস্ত i কেটে দিতে থাকবো। যতক্ষণ পর্যন্ত n এর মধ্যে একটিও i এর গুনিতক থাকবে ততক্ষন $n\%i == 0$ সত্য হবে। এভাবে কাটতে থাকলে এটা প্রমান করা সম্ভব যে if condition এর ভেতরে i সবসময় মৌলিক সংখ্যাই হবে।

এরপর ৮ নং লাইনে আমরা সুত্র প্রয়োগ করে result হিসেব করেছি।

১৩ নং লাইনে দেখেছি $n > 1$ কিনা। কারণ $n > 1$ হলে আমরা বলতে পারবো n এর একটি মৌলিক গুণনীয়ক রয়েছে যা $\sqrt{n}$ এর থেকে বড়। তাই আমরা এক্ষেত্রে আবার result আপডেট করেছি। তারপর রেজাল্ট রিটার্ন করেছি।

Euler Totient Function: 1 থেকে **n** পর্যন্ত সবগুলো সংখার **phi** ফাংশনের মান নির্ণয়

উপরে আমরা যেভাবে হিসেব করলাম সেই ফাংশনের কমপ্লেক্সিটি $O(\sqrt{n})$। এই ফাংশন শুধু একটি সংখার সহমৌলিক গননা করতে পারবে। এখন আমরা যদি 1 থেকে n পর্যন্ত প্রত্যেকটি সংখার সহমৌলিক গননা করতে চাই, তবে আমরা 1 থেকে n পর্যন্ত সংখাগুলোর জন্য আলাদা আলাদা করে বের করবো না। কারণ এটা ইফিসিয়েন্ট নয়।

এই কাজ করার দুইটা উপায় বর্ণনা করবো। প্রথমটি হলো সিভ অফ এরাটোস্থেনিস মৌলিক সংখ্যা (Prime number) এর অ্যালগরিদম গুলো। আমি এখানে সিভ নিয়ে বিস্তারিত আলোচনা করছি না।

আমরা সিভ দিয়ে মৌলিক সংখ্যা বের করার সময় কি করি? একটি সংখ্যা i নেই, তারপর পরীক্ষা করে দেখি i কি ইতোমধ্যে কোন সংখ্যার গুনিতক হিসেবে কাটা গিয়েছে কিনা।

যদি কাটা না যায় তারমানে i একটি মৌলিক সংখ্যা এবং i এর যত গুনিতক আছে এর পর সবাইকে আমরা $1 - \frac{1}{i}$ দিয়ে গুন করবো।

```
#define mx 15
float phi[mx+1];
void sievePhi(int n){
 for(int i=1;i<=n;i++){
 phi[i]=i;
 }
 for(int i=2;i<=n;i++){
 if(phi[i]==i){
 for(int j=i;j<=n;j+=i){
 phi[j]*=(1-1.0/i);
 }
 }
 }
}
```

এই ফাংশনের প্রথম লুপে আমরা phi[i]=i; করে দিয়েছি (এখানে phi অ্যারেটি ১ থেকে n পর্যন্ত সংখাগুলোর ফাই ফাংশনের মান ধারন করবে)। এতে করে আমাদের সুত্রের শুরুতে যেই n গুন আছে তা আসাইন করা হলো। এর পর লুপে আমরা ২ থেকে শুরু করেছি। কারণ ২ ছোট মৌলিক সংখ্যা। লক্ষ করি আমাদের লুপের শর্ত i*i<=n না হয়ে i<=n হয়েছে। কারণ হিসেবে ১৪ কে বিবেচনা করি। ১৪ এর মৌলিক গুণনীয়ক ২,৭। এখন i*i<=n শর্ত মোতাবেক লুপ ঘুরালে আমরা লুপের ভেতরে i এর সর্বোচ্চ মান পাবো ৩। কিন্তু ফাই ফাংশনের সুত্রে আমরা দেখছি সকল মৌলিক গুণনীয়ক দিয়ে কাজ করতে হয়। তাই আমাদেরকে ৭ ও বিবেচনা করতে হবে।

দ্বিতীয় লুপের মধ্যে if(phi[i]==i) এটা শুধু i যদি মৌলিক সংখ্যা হয় তখনি সত্য হবে এবং পরের লুপে মৌলিক সংখ্যা i এর সমস্ত গুনিতক j এর phi[j] কে (1-1.0/i) দ্বারা গুন করেছি। এতে করে আমরা $O(n \log(\log n))$ এ ১ থেকে n পর্যন্ত সংখ্যা গুলোর ফাই ফাংশনের মান বের করলাম।

Euler’s Totient Function: Divisor Sum property

এই প্রপার্টিটি দারুন। সংখ্যা n এর সমস্ত গুননিয়কের ফাই ফাংশনের যোগফল n এর সমান হবে। অর্থাৎ $\sum_{d|n} \varphi(d) = n$

অর্থাৎ $n = d_1 \times d_2 \times d_3 \times \ldots . \times d_n$ হলে
$n = \varphi(d_1) + \varphi(d_2) + \varphi(d_3) + \ldots + \varphi(d_n)$।

উপরের sievePhi ফাংশন দিয়ে আমরা d_1, d_2, d_n ইত্যাদি গুননিয়কের মান বের করে খুব সহজেই এটা ইমপ্লিমেন্ট করতে পারবো।

আমরা ব্লগের sieve of eratosthenes নিয়ে লিখাটি একটু আউটডেটেড। আপডেট করা হবে ইনশাল্লাহ।

আশা করি বুঝতে পেরেছেন। এই লিখায় অনেক গুলো সমীকরণ রয়েছে, ভুল হলে অনুগ্রহ করে আমাকে জানাবেন। শিগ্রই হাজির হবো আরেকটি লিখা নিয়ে। সেই পর্যন্ত বিদায়।

অনুশীলনের জন্য নিচের সমস্যা গুলো সমাধান করতে পারেন।

- SPOJ #4141 "Euler Totient Function" [Difficulty: CakeWalk]
- UVA #10179 "Irreducible Basic Fractions" [Difficulty: Easy]
- UVA #10299 "Relatives" [Difficulty: Easy]
- UVA #11327 "Enumerating Rational Numbers" [Difficulty: Medium]
- TIMUS #1673 "Admission to Exam" [Difficulty: High]
- UVA 10990 – Another New Function
- Codechef – Golu and Sweetness
- SPOJ – LCM Sum
- GYM – Simple Calculations (F)
- UVA 13132 – Laser Mirrors
- SPOJ – GCDEX
- UVA 12995 – Farey Sequence
- SPOJ – Totient in Permutation (easy)
- LOJ – Mathematically Hard
- SPOJ – Totient Extreme
- SPOJ – Playing with GCD
- SPOJ – G Force
- SPOJ – Smallest Inverse Euler Totient Function
- Codeforces – Power Tower

লেখাটি কেমন লেগেছে আপনার?

রেটিং দিতে হার্টের উপর ক্লিক করুন।

গড় রেটিং 4.4 / 5. মোট ভোট: 17

#সংখ্যাতত্ত্ব