সংখ্যাতত্ত্ব - Number theory

সংখ্যাতত্ত্ব: এক্সটেন্ডেড ইউক্লিডিয়ান অ্যালগরিদম

Extended Euclidean algorithm Bangla tutorial/ Calculating gcd in O(log n) time/ Extended Euclid Bangla/ এক্সটেন্ডেড ইউক্লিডীয় অ্যালগরিদম

Sharif Hasan • August 23, 2021 সর্বশেষ আপডেট May 28, 2022 0 503 পড়তে 3 মিনিট লাগতে পারে

আমরা এর আগের ইউক্লিডিয়ান অ্যালগোরিদম নিয়ে লিখায় দেখেছিলাম কিভাবে দুইটি সংখ্যা a, b এর গসাগু $O(\log n)$ এ বের করা যায়। ইউক্লিডিয়ান অ্যালগোরিদমের বর্ধিত ভার্সনের (এক্সটেন্ডেড ইউক্লিডিয়ান অ্যালগরিদম বা Extended Euclidean algorithm) মাধ্যমে আমরা গ.সা.গু বের করার পাশাপাশি একটি সমীকরণ $a.x + b.y = gcd(a, b)$ সমাধান করা যায় ।

এখানে a, b হলো প্রদত্ত দুটি পূর্ণসংখ্যা যাদের গসাগু বের করতে হবে এবং x, y হলো a, b এর সহগ এবং এরাও দুইটি পূর্ণ সংখ্যা। আমরা এক্সটেন্ডেড ইউক্লিডিয়ান অ্যালগোরিদমের এর মাধ্যমে x, y এর মান বের করতে পারবো। ‘

Extended Euclidean (এক্সটেন্ডেড ইউক্লিডিয়ান) algorithm Bangla tutorial

$$a.x + b.y = gcd(a, b) \ldots\ldots\ldots (1)$$

এখানে আমাদের জেনে রাখা জরুরি যে সবসময় x, y এর এমন পাওয়া সম্ভব যার জন্য $a.x + b.y = gcd(a, b)$ সত্য হবে। এবং $gcd(a, b)$ ই হলো সবথেকে ছোট ধনাত্মক সংখ্যা যাকে a, b এর এমন linear combination রূপে লিখা সম্ভব। প্রমান এখানে (Bézout’s Lemma)।

আমরা গত লিখায় যেই কোডটা করেছিলাম সেই কোডটাকে আবার একটু দেখি,

```
int gcd(int a,int b){
 if(b==0) return a; // যখন b=0 হবে, তখন a আমদের গ.সা.গু.।
 return gcd(b,a%b); //a=b এবং b=a%b করে দিলাম রিকার্সিভ ফাংশন কলের মাধ্যমে।
}
```

এখানে প্রতিবার আমরা $a = b$ এবং $b = a\%b$ এর মাধ্যমে রিকার্সিভ কল করেছি। সুতরাং আমরা লিখতে পারি,

$$a = b$$

$b = a\%b$ বা $a = a - \lfloor \frac{a}{b} \rfloor$ এখানে [ভাগশেষ (b)= ভাজ্য (a) – ভাগফল x ভাজক (b)] এবং $\lfloor x \rfloor$, $floor(x)$ এর সমতুল্য।

ধরা যাক ইনপুট $b, a\%b$ এর জন্য আমাদের সমাধান হলো x_1, y_1। তবে (1) নং সমীকরণ থেকে আমরা লিখতে পারি,

$b.x_1 + (a\%b).y_1 = gcd(a,b)$ … যেহেতু $a = b$ এবং $b = a\%b$

> *এখানে অনেকের কনফিউশন তৈরি হয়, এখানে $b.x_1$ এ b লিখেছি কারণ প্রতিবার রিকারসিভ কল করার সময় a এর মান হিসেবে b কে পাস করা হয় [a=b]। একই ভাবে $(a\%b).y_1$ লিখার কারণ হলো রিকার্সিভ কল করার সময় b এর মান হিসেবে $a\%b$ ইনপুট দেয়া হয় [b=a%b]।*

এখন সবকিছু মাথা থেকে ঝেরে ফেলি। ধরা যাক আমরা রিকার্সিভ কল করতে করতে বেস কেসে (Base case) এসে পৌছালাম। আমরা জানি বেস কেসে $b = 0$ এবং a এর মান গ.সা.গু.র সমান হয়। তাই $a = gcd(a,b)$ । আমরা যদি এসময় $a.x_1 + b.y_1 = gcd(a,b)$ এর সমাধান করার চেষ্টা করি তবে কিন্তু সহজেই বলতে পারি, $x_1 = 1, y_1 = 0$ হবে। কারণ,
$g.1 + 0.0 = gcd(a,b)$

বিশেষ দ্রষ্টব্য: y_1 এর মান আমরা যাই নিই, আমাদের সমীকরণ সত্য হবে।

সুতরাং আমরা বলতে পারি বেস কেসে আমরা x_1, y_1 এর সমাধান পেয়েছি, যেখানে $a = gcd(a,b)$ এবং $b = 0$। এই সমাধান কাজে লাগিয়ে আমরা প্রদত্ত a, b এর জন্য x, y এর মান বের করবো।

এখন x_1, y_1 যদি আমাদের একটি সমাধান হয় তবে

$$
\begin{aligned}
&a.x_1 + b.y_1 = gcd(a,b)\\
&b.x_1 + (a\%b).y_1 = gcd(a,b)\\
&b.x_1 + (a - \lfloor \frac{a}{b} \rfloor .b).y_1 = gcd(a,b)\\
&b.x_1 + a.y_1 - b.\lfloor \frac{a}{b} \rfloor .y_1 = gcd(a,b)\\
&a.y_1 + b.(x_1 - \lfloor \frac{a}{b} \rfloor .y_1) = gcd(a,b) \ldots\ldots\ldots (2)
\end{aligned}
$$

(1) এবং (2) নং সমিকরনের a, b এর সহগ সমীকৃত করে পাই,

$x = y_1$

$$y = x_1 - \lfloor \frac{a}{b} \rfloor . y_1$$

হয়ে গিয়েছে। আমরা সমস্যার সমাধান পেয়ে গিয়েছি। এখানে আমরা আগের স্টেট এর সমাধান কাজে লাগিয়ে বর্তমান স্টেটের a, b এর মানের জন্য সমাধান বের করতেছি। এভাবে রিকার্সিভ কল করে আমরা প্রদত্ত a, b এর জন্য সমাধান বের করবো।

আশা করি এই পর্যন্ত বুঝা গিয়েছে, কোড দেখলে পরিস্কার হয়ে যাবে ইনশাআল্লাহ।

Extended Euclidean (এক্সটেন্ডেড ইউক্লিডিয়ান) algorithm Bangla – Code implementation with C++

নিচে আমরা C++ ব্যবহার করে এক্সটেন্ডেড ইউক্লিডিয়ান অ্যালগরিদমের ইমপ্লিমেন্টেশন করেছি। এখানে extended_euclid() ফাংশনটা 4 টি প্যারামিটার নিবে। এর মধ্যে প্রথম দুইটি দুইটি পুর্ন সংখ্যা ইনপুট নিবে যাদের গসাগু আমরা বের করবো। পরের দুইটি প্যারামিটার প্রথমত কোন মান নিবে না, কিন্তু রিকার্সিভ ভাবে সমিকরনের সমাধান বহন করে আনবে। আর এই ফাংশন একটি পূর্ণসংখ্যা বা Integer রিটার্ন করবে যা a, b এর গ.সা.গু.।

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4
5  int extended_euclid(int a,int b,int &x,int &y){
6   if(b==0){
7   x=1;
8   y=0;
9   return a;
10  }
11  int x1,y1;
12  int gcd=extended_euclid(b,a%b,x1,y1);
13  x=y1;
14  y=x1-floor(a/b)*y1;
15  //cout<<x<<" "<<y<<endl;
16  return gcd;
17 }
18
19 int main(){
20  int x,y;
21  int gcd=extended_euclid(12,8,x,y);
22  cout<<gcd<<" "<<x<<" "<<y<<endl;
23  return 0;
24 }
```

এই অ্যালগরিদমের দুইটি অংশ আছে। প্রথম অংশ ফাংশনের শুরু থেকে লাইন ১১ পর্যন্ত এবং দ্বিতীয় অংশ ১২ থেকে ফাংশনের শেষ পর্যন্ত। প্রথম অংশে আমরা রিকার্সিভ কল করার মাধ্যমে বেস কেস পর্যন্ত আসি যেখানে ফাংশন কলের ধাপ শেষ হয় এবং রিটার্ন করে উপরে যাবার ধাপ শুরু হয়।

উপরের চিত্রটি দেখি। আমরা `int gcd=extended_euclid(b,a%b,x1,y1);` এই লাইনের মাধ্যমে রিকার্সিভ কল করতে থেকেছি, যতক্ষন পর্যন্ত বেস কেস b=0 তে না পৌছাই। প্রথম ধাপে a=12, b=8 এখানে a এবং b এর গসাগু বের করবো। এর পরের ধাপে a=8, b=4 এবং তার পরের ধাপে a=4, b=0।

যখনি b=0, তখন আমরা বেস কেসে চলে এসেছি। আমরা শুরুতে দেখেছিলাম, যদি $g = gcd(a, b)$ হয় তবে $a.x + b.y = g$ এ x এবং y এর মান যথাক্রমে 1,0 কারণ এখানে $a = g$।

পরের অংশে আমরা x এবং y এর মান হিসাব করার মাধ্যমে উপরে যেতে থাকি gcd এর মান তার কলার ফাংশন কে রিটার্ন করার মাধ্যমে।

উপরের ডায়াগ্রাম খেয়াল করুন। আমি রিকার্সিভ কলের কোন পর্যায়ে চলকের মান কত তাকে প্রকাশ করেছি। আশাকরি বুঝতে সুবিধা হবে।

যখন আমরা extended_euclid() ফাংশন কল করেছি তখন $x1, y1$ কে আমরা **Call by reference** এ প্রেরণ করেছি। এতে করে x1 এবং y1 এর মাধ্যমে আমরা প্রতিটি রিকার্সিভ কলের সমাধান পাবো। যা ব্যবহার করে আমরা x, y এর মান বের করতে পারবো।

কোডে x=y1 এবং y=x1-floor(a/b)*y1 এর মাধ্যমে আমরা x এবং y এর মান বের করে গিয়েছি। এখানে x এবং y কে আপডেট করার সময় এর পূর্ববর্তী রিকার্সিভ স্টেজের x1 এবং y1 এর মান পরিবর্তন হবে, যেহেতু x এবং y কে রেফারেন্সের মাধ্যমে ইনপুট দেয়া হয়েছে। এভাবে একসময় আমরা x, y এর সমাধান পাবো।

Practice Problems

- 10104 – Euclid Problem
- GYM – (J) Once Upon A Time
- UVA – 12775 – Gift Dilemma

আজকে এই পর্যন্তই। পরবর্তী লিখাতে সংখ্যাতত্ত্বের অন্য কোন বিষয়ে আলোচনা করবো। সেই পর্যন্ত #Happy_coding.

লেখাটি কেমন লেগেছে আপনার?

রেটিং দিতে হার্টের উপর ক্লিক করুন।

গড় রেটিং 4.7 / 5. মোট ভোট: 18

#সংখ্যাতত্ত্ব