৭ এবং ৮ – আমরা যদি এ দু'টি সংখ্যার গ.সা.গু (GCD) নিই, তাহলে পাব ১। অর্থাৎ এই দু'টি সংখ্যার মধ্যে ১ ছাড়া আর কোনো কমন ফ্যাক্টর নেই। এক্ষেত্রে আমরা বলে থাকি ৭ এবং ৮ হল সহমৌলিক (relatively prime)। আবার ৪ এবং ৮, এদের কমন ফ্যাক্টর রয়েছে ১, ২ এবং ৪। যেহেতু এদের মধ্যে ১ ছাড়া আরও কমন ফ্যাক্টর রয়েছে, তাই এরা রিলেটিভলি প্রাইম না!

এখন ধরা যাক, আমাদেরকে বলা হল ১ থেকে ৮ পর্যন্ত ৮ এর যতগুলো রিলেটিভ প্রাইম আছে তা বের করতে হবে, তাহলে আমরা প্রথমেই ১ থেকে ৮ পর্যন্ত সংখ্যা গুলো লিখে নিঃ

১ ২ ৩ ৪ ৫ ৬ ৭ ৮

এদের মধ্যে ৮-এর গুণনীয়ক আছে ২ এবং ৪, শুরুতেই এরা বাদ। এছাড়াও ৬ বাদ, কারণ ৮ এবং ৬ এর গসাগু ২।

তাহলে বাকি থাকলোঃ

১ ৩ ৫ ৭

তাহলে, আমাদের শর্ত অনুযায়ী রিলেটিভ প্রাইম পেয়েছি ৪ টি।

আমরা বলবো Φ(8) = 4

এখন প্রশ্ন হল এই Φ আসলো কোথা থেকে! এই Φ ফাংশনটি হল **Breakability of a number**। অর্থাৎ একটি সংখ্যা যতটি সংখ্যার সহমৌলিক, তাই হল সেই সংখ্যার Φ ফাংশনের মান। এই ফাংশনের আবিস্কারক Euler। অন্যভাবে একে বলা হয় **Totient Function**।

এবার আমরা আরেকটি সংখ্যার ফাই ফাংশনের মান বের করি – ১৩

gcd(1,13) = 1
gcd(2,13) = 1
gcd(3,13) = 1
gcd(4,13) = 1
gcd(5,13) = 1
gcd(6,13) = 1
gcd(7,13) = 1
gcd(8,13) = 1
gcd(9,13) = 1
gcd(10,13) = 1
gcd(11,13) = 1
gcd(12,13) = 1
gcd(13,13) = 13
অর্থাৎ, Φ(13) = 12 = 13-1

এবার তুমি যদি আরও কয়েকটি মৌলিক সংখ্যা নিয়ে তার Φ ফাংশনের মান বের কর, যেমন ৭,১১,১৯, তাহলে দেখবে প্রতি ক্ষেত্রেই

Φ(n) = n-1, when n is prime

তবে মৌলিক সংখ্যার ক্ষেত্রে কাজটা অনেক সহজ হলেও অন্যান্য ক্ষেত্রে কিছুটা জটিল। এজন্য আমাদের একটি Theorem জানতে হবে। সেটি হলঃ

Φ(m*n) = Φ(m) * Φ(n), when m and n is relatively prime

খেয়াল কর, এই theorem-এর জন্য m এবং n অবশ্যই রিলেটিভলি প্রাইম হতে হবে। এখন আমাদের কাছে দু'টি তত্ত্ব আছে।

প্রথমত, **Φ(n) = n-1, when n is prime**

দ্বিতীয়ত, **Φ(m*n) = Φ(m) * Φ(n), when m and n is relatively prime**

তাহলে আমরা যেকোনো সংখ্যার Φ ফাংশনের মান বের করতে চাইলে সে সংখ্যাটিকে ততক্ষণ দু'টি সহমৌলিক সংখ্যার গুণফল হিসেবে লিখবো, যতক্ষণ না প্রত্যেকটিই মৌলিক সংখ্যা হয়। একটা উদাহরণ দিয়ে ব্যাপারটা বুঝার চেষ্টা করা যাক। ধরি, আমাদের সংখ্যাটি হল ২১০।

তাহলে,

$$\begin{aligned}\Phi(210) &= \Phi(2) * \Phi(105)\\ &= \Phi(2) * \Phi(3) * \Phi(35)\\ &= \Phi(2) * \Phi(3) * \Phi(5) * \Phi(7)\end{aligned}$$

এখন, এখানে ২, ৩, ৫ এবং ৭ মৌলিক সংখ্যা হওয়ায় এদের ф ফাংশনের মান হবে n-1। তাহলে,

$\Phi(210) = (2-1) * (3-1) * (5-1) * (7-1) = 1*2*4*6 = 48$

এখন ধরা যাক, আমরা 4-এর ф ফাংশনের মান বের করতে চাই। আমরা এখন পর্যন্ত যা জানি, তা অনুযায়ী,
$\Phi(4) = \Phi(2)*\Phi(2) = 1*1 = 1$

কিন্তু আমরা যদি ম্যানুয়ালি 4-এর ф ফাংশনের মান বের করি তাহলে দেখবোঃ

gcd(1,4) = 1, রিলেটিভলি প্রাইম
gcd(2,4) = 2
gcd(3,4) = 1, রিলেটিভলি প্রাইম
gcd(4,4) = 4

অর্থাৎ, $\Phi(4) = 2$

যা আমাদের থিওরেম-এর ক্ষেত্রে মিলে নি। আমাদের থিওরেম শুধু তখন মিলবে যখন প্রতিটি প্রাইমের ঘাত (power) 1 হয়।

কিন্তু এক্ষেত্রে $\Phi(4) = \Phi(2^2)$, অর্থাৎ, ২-এর ঘাত ১ ছিল না, তাই থিওরেমটিও কাজ করে নি! এক্ষেত্রে আমাদের একটি অনুসিদ্ধান্ত (postulate) ব্যবহার করতে হবে। অনুসিদ্ধান্তটি হলঃ

$\Phi(n^k) = n^k - n^{k-1}$, when n is prime

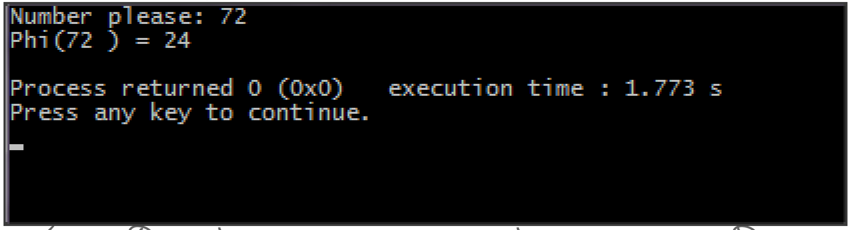
তাহলে আমরা 4-এর রিলেটিভ প্রাইম বের করবো এভাবেঃ

$\Phi(4) = \Phi(2^2) = 2^2 - 2^{2-1} = 2$

অর্থাৎ, কাজ শেষ! এবার আমরা ৭২-এর ф ফাংশনের মান বের করি।

$$\begin{aligned}\Phi(72) &= \Phi(8) * \Phi(9)\\ &= \Phi(2^3) * \Phi(3^2)\\ &= (2^3 - 2^{3-1}) * (3^2 - 3^{2-1})\\ &= (8-4) * (9-3)\\ &= 24\end{aligned}$$

আমরা এবার একটি সিম্পল সি কোড লিখে দেখে নিই আমাদের বের করা (৭২)-এর মান ঠিক আছে কি না। কোডটির আউটপুটঃ



```
Number please: 72
Phi(72 ) = 24

Process returned 0 (0x0)   execution time : 1.773 s
Press any key to continue.
```

অর্থাৎ সব ঠিকমতই কাজ করছে। তোমরা চাইলে আরও কয়েকটি সংখ্যার জন্য কাজটি করে দেখতে পার। যেমনঃ

- 425
- 625

- 512
- 995328
- 124179

উত্তর মিলিয়ে দেখতে পার কোডটা রান করে। কোডটা আছে এখানে।
তবে একটা সমস্যা। এ কোড দিয়ে Φ(124179) বের করতে গিয়ে দেখবা অনেক সময় লেগে যাচ্ছে। কারণ আমরা আমাদের থিওরেমগুলো ব্যবহার না করে কামলা খাটানো নিয়মে (Brute Force) করে ফেলছি। তাহলে এবার আমরা আরেকটা কোড লেখার চেষ্টা করবো, যেটা আরও বেশি এফিসিয়েন্ট হবে।
এজন্য আমরা আবার আমাদের Φ(72)-এর কাছে ফিরে যাব।
72 = (2 x 2 x 2) x (3 x 3)
Φ(৭২)-এর সর্বোচ্চ মান হতে পারে ৭২। তাই আমরা প্রথমেই একটা ভ্যারিয়েবল নিব ans এবং এর মান ইনিশিয়ালাইজ করে দিব ৭২।
এর মৌলিক গুণনীয়ক আছে দুইটি – ২ এবং ৩।
১ থেকে ৭২-এর মধ্যে ২-এর গুণনীয়ক কয়টি আছে বল তো? সহজ উত্তরঃ ৭২/২ = ৩৬ টি।
তাহলে, আমরা লিখবো ans = ans-36 = 36
এবার আমরা ৭২-কে যতক্ষন দুই দিয়ে ভাগ করা যায় (৩ বার করা যাবে), ভাগ করে যাব।
72/2 = 36
36/2 = 18
18/2 = 9
এবার, 9 = 3*3
১ থেকে ৩৬ এর মধ্যে ৩-এর গুণিতক আছে ৭২/৯ টি
তাহলে ans = ans – 72/9= 36-8 = 24
এরপর ৯ কে ২ বার ৩ দিয়ে ভাগ করলে পেয়ে যাব ১। তাই আর কিছু করা লাগবে না!। তার মানে আমাদের উত্তর পেলাম ২৪, যা আগের সাথে মিলে যায়! এক্ষেত্রে আমরা শুধু ১ বিয়োগ করে দিয়েই কাজ শেষ করে দিতাম! 😀
এবার বল তো, যদি মৌলিক সংখ্যাকে এভাবে করতে চাই, তাহলে কী করবো? কোনো সংখ্যা দিয়েই তো ভাগ যাবে না। তাহলে কী ans = n রিটার্ন করে দিব? না। যদি প্রাইম হত, তাহলে শেষে অবশ্যই ১ আসবে। যদি ১ না আসে, সেক্ষেত্রে ans=ans-1 করে দিব। কারন, প্রাইমের ক্ষেত্রে, Φ(n) = n-1। কথা না বাড়িয়ে কোডটাই লিখে দি। আসা করি বুঝতে পারবে।

```
#include <stdio.h>
int totient (int i) {
    int ans; /* Result */
    int j;

    if (i==1) return 1;
        ans=i;
        /* Check for divisibility by every prime number below the square root. */
        /* Start with 2. */
        if (i%2==0) {
        ans -= ans/2;
        while (i%2==0) {
            i/=2;
        }
    }
```

```

        /* Since this doesn't use a list of primes, check every odd number. Ideally, skip past composite numbers.*/
    for (j=3; j*j<=i; j+=2) {
        if (i%j==0) {
            ans-=ans/j;
            while (i%j==0) {
                i/=j;
            }
        }
    }

    /* If i>1, then it's the last factor at this point. */
    if (i>1) ans -= ans/i;

    /* Return the result. */
    return ans;
}
int main() {
    int in;
    printf("A number please: ");
    scanf("%d", &in);
    printf("Phi(%d) = %d\n", in, totient(in));
    return 0;
}
```

কিন্তু যখন ডাটা ইনপুট সেট অনেক বড় হবে অর্থাৎ, পরপর কয়েকটি ইন্টিজারের জন্য ফাই ফাংশনের মান জানতে চাওয়া হয়, সেক্ষেত্রে এই এলগোতেও টিএলই খেতে হবে। এজন্য আমরা প্রাইম নাম্বারের মত এক্ষেত্রেও সিভ ব্যবহার করবো। নিচে কোডটা দেওয়া হলঃ

```
#include <stdio.h>
#define MAX 2000100
int phi[MAX] ;
void sievePHI(){
    long long i,j;
    for( i=2;i<MAX;i++){
        if( phi[i]==0){
            phi[i] = i-1;
            for( j = i*2; j<MAX; j+=i){
                if( phi[j] == 0 )phi[j] = j;
                phi[j] /= i ;
                phi[j] *= (i-1) ;
            }
        }
    }
}
int main() {
    int T;
    printf("Koi ta number er Phi function er value jante chan?\n");
    scanf("%d", &T);
    sievePHI();
    while (T--) {
        int in;
```

```
        printf("A number please: ");
        scanf("%d", &in);
        printf("Phi(%d) = %d\n", in, phi[in]);
    }
    return 0;
}
```

এই কোডের সাহায্যে আমরা প্রথম ২০০০০০০ ইন্টিজারের ফাই ফাংশনের মান যথেষ্ট দ্রুত বের করতে পারবো!
আর অতিরিক্ত হিসেবে কিছু দিয়ে যাচ্ছি আজ। ২০ ডিজিটের নিচে যেকোনো নাম্বারের ফাই ফাংশনের ভ্যালু অত্যন্ত দ্রুততার সাথে যদি বের করতে চাও,নিচের কোডটি তাহলে তোমাদের দারুণ কাজে লাগবে। কারণ নিচের কোডটি কোনোপ্রকার অ্যারে এর সাহায্য ছাড়াই পুরো কাজটি করে ফেলে। ☺

```
#include <bits/stdc++.h>
using namespace std;
#define ll long long

ll po(ll a,ll b)
{
    ll ans=1;
    while(b--) ans*=a;
    return ans;
}
ll prime(ll a)
{
    for(int i=1; i*i<=a; i++)
    {
        if(a%i==0) return 1;
    }
    return 0;
}

void phi(long long int n)
{
    ll i,mul=1,holder,fre=0;
    if(prime(n)==0) mul=n-1;
    else
    {
        for(i=2; i*i<=n; i++)
        {
            if(n%i==0)
            {
                while(n%i==0)
                {
                    n=n/i;
                    holder=i;
                    fre++;

                }
                mul*=(po(holder,fre-1)*(holder-1));
                fre=0;
```

```
            }
        }
        if(n!=1) {

                mul*=(n-1);
        }

    }
    cout << mul << endl;
}

int main()
{
    long long int n;
    scanf("%lld",&n);
    phi(n);
    return 0;
}
```