函 数

sqrt 函数与性能入门

```
#include <bits/stdc++.h>//1-1023 javacn
using namespace std;
int main()
{
	int n;		// 定义整数变量 n
	cin>>n;		// 输入 n
	int i;
	bool flag = true;
	for(i=2;i<=sqrt(n);i++)
	{
		if(n%i==0)
		{
			flag = false;
			break;
		}
	}
	if(n<=1)
	{
		flag = false;
	}

	if(flag)	cout<<"T";
	else		cout<<"F";

	return 0;
}
```

```cpp
#include<bits/stdc++.h>//2-1084  javacn
using namespace std;
int main()
{
	int n,i,s=0;//s 作为和，初始值为 0
	cin>>n;
	for(i=2;i<=sqrt(n);i++)
	{
		if(n%i==0) //判断此时的 i 是不是就是 n 的平方根
		{
			if(i==sqrt(n))
			{
				s = s + i;
			}
			else
			{
				s = s + i + n/i;
			}
		}
	}
	cout<<s;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//3-1060  javacn  推荐学习
using namespace std;
int main()
{
	// 定义整数变量 n ，s 作为和，初始值为 0
	int n,s=0;
	// 输入 n
	cin>>n;
	if(n%2==0)
	{
		// 如果 n 是偶数，则求 n 所有的约数之和
		// 包括 1 和本身
		for(int i=1;i<=n;i++)
		{
			if(n%i==0)
			{
				s+=i;
			}
		}
	}
	else
	{
		// 如果 n 是奇数，则求 1-n 之间所有的偶数之和
		for(int i=1;i<n;i++)
		{
			if(i%2==0)
			{
				s+=i;
			}
		}
	}
	cout<<s;	// 输出 s
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//4-1089   wintereta
using namespace std;
int zs(int x){
	for(int j=2;j<=sqrt(x);j++)
	{
		if(x%j==0)
		{
			return 1;
		}
	}
	if(x==1)
	{
		return 1;
	}
	else
	{
		return 0;
	}
}
int main(){
	for(int i=100;i<=999;i++)
	{
		int g=i/1%10;
		int s=i/10%10;
		int b=i/100%10;
		if(g!=s&&s!=b&&g!=b)
		{
			if(s>(g+b)&&zs(s+b))
			{
				cout<<i<<endl;
			}
		}
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//5-1061  javacn
using namespace std;
int main()
{
	// 定义整数变量 n 作为输入的数 ，i 用作循环 ，y 记录最小的约数
	int n,i,y;
	// 输入 n
	cin>>n;
	// 默认标识是质数
	bool f = true;
	for(i=2;i<=sqrt(n);i++)
	{
		// 如果 n 能被 i 整除，说明 n 不是质数
		if(n%i==0)
		{
			f = false;
			y = i;
			break;
		}
	}
	if(n<=1)
	{
		f = false;
	}

	if(f)
	{
		cout<<"Yes";
	}
	else
	{
		cout<<y;
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//6-1253-1  hongds  推荐学习
using namespace std;
int main(){
	int i,q,b,s,g,x;
	for(i=1000;i<=9999;i++)
	{
		q=i/1000;
		b=i/100%10;
		s=i/10%10;
		g=i%10;
		x=sqrt(i);
		if(q==b&&s==g&&b!=s&&i==x*x)
		{
			cout<<i;
		}
	}
	return 0;
}
#include <bits/stdc++.h>//6-1253-2  javacn  推荐学习
using namespace std;// 思路：遍历 i 范围为 4 位整数  i 满足 3 个条件 分别判断
int main(){
	int i;
	for(i=1000;i<=9999;i++) // 循环范围在 4 位数以内
	{
		// 前两位数字是相同的，后两位数字是相同的
		if(i/1000==i/100%10&&i%100/10==i%10)
		{
			if(sqrt(i)==int(sqrt(i)))// 四位的车号刚好是一个整数的平方
			{
				cout<<i<<endl;
			}
		}
	}
	return 0;
}
```

更为严谨的解法，因为题目要求前后两位数不能是同一个数，并且车的号码可能是 00 开头的

```
#include<bits/stdc++.h>//6-1253-3  vixenly
using namespace std;
int main(){

	// 四位号 n，n 为完全平方数，前两位数相同，后两位数相同，
	/*n 的范围为 0000~9999，	0011,0022,0033.....0099
	1100,	1122,1133,1144.....1199
	2200,2211,	2233,2244.....2299
	...
	9911,9922,9933,9944.....9988,
	一共 100 个数，且前两位和后两位能被 11 整除
	*/
	int n,i,j;
	double r;
	for(i=0;i<=9;i++)
	{
		for(j=0;j<=9;j++)
		{
			if(i!=j)
			{
				n=i*11*100+j*11;
				r=sqrt(n);
				if(r==(int)r)
				{
					cout<<n;
					break;
				}
			}

		}
	}
	return 0;
}
```

思路：穷举遍历查找符合条件的数就跳出循环

```
#include <bits/stdc++.h>//7-1267  javacn
using namespace std;
int main()
{
	int n=0;
	// 穷举查找符合条件的 n
	while(1)
	{
		// 判断 n+100 是一个完全平方数
		if(sqrt(n+100)==int(sqrt(n+100)))
		{
			// 判断 n+100+168 是一个完全平方数
			if(sqrt(n+100+168)==int(sqrt(n+100+168)))
			{
				// 两个条件都符合则输出 n 并跳出循环
				cout<<n;
				break;
			}
		}
		n++;
	}
	return 0;
}
```

思路：1. 若一个数能表示成某个整数的平方的形式，则称这个数为完全平方数，判断 i 的平方根是否是整数

2. 前两位相同且后两位相同

```
#include <bits/stdc++.h>//8-1092  javacn
using namespace std;

int main()
{
	int x,g,s,b,q;
	// 循环范围内的数
	for(int i = 1000;i <= 9999;i++)
	{
		x = sqrt(i);
		q = i / 1000;
		b = i / 100 % 10;
		s = i / 10 % 10;
		g = i % 10;

		// 判断平方数
		if(x * x == i && q == b && s == g)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//9-1063  javacn  仅作参考
using namespace std;
int main()
{
	int m,n,c,s=0,i,j;//s:素数的个数
	cin>>m>>n;

	for(i=m;i<=n;i++)		//m~n 范围内寻找
	{
		// 对每次循环的 i 进行判断是不是素数
		// 每次判断 i 时，c=0
		c = 0;

		for(j=2;j<=i-1;j++)//j:循环其实是 i 可能的因子
		{
			if(i%j==0)
			{
				c++;
				break;
			}
		}

		if(c==0 && i>=2)    // 如果条件成立，那么当前的 i 就是素数
		{
			s++;
		}
	}
	cout<<s;
	return 0;
}
```

解法一：嵌套循环求出每个数因子的数量，注意因子个数计数器需要在 i 循环进入之后才能清零。

```
#include <bits/stdc++.h>//10-1495-1  javacn  推荐学习
using namespace std;
/* 思路：第一步：循环 1~n 的每个数
第二步：嵌套循环求出每个数有几个因子 */
int main()
{
	int i,j,n,c;
	cin>>n;
	for(i = 1;i <= n;i++)
	{
		// 计数器清零，求每个数 i 有几个因子，都要从 0 开始求
		c = 0;
		// 循环 i 可能的因子范围
		// 写法一：循环 2~i-1，逐个判断是否是因子
//		for(j = 2;j <= i - 1;j++)
//		{
//			if(i % j == 0){   c++;  }
//		}
		// 写法二：循环 2~sqrt(i)，成对求解 i 的因子
		for(j = 2;j <= sqrt(j);j++)
		{
			if(i % j == 0)
			{
				// 两个因子相等
				if(j == i / j)    { r = r + 1;}// 只能算 1 个
				else              { r = r + 2;}// 两个因子不等，算 2 个
			}
		}
		cout<<c<<endl;
	}
	return 0;
}
```

解法二：定义函数求因子数量，在 main 中调用函数求每个数的因子数量

```
#include <bits/stdc++.h>//10-1495-2  javacn  推荐学习
using namespace std;
/*
思路：
第一步：定义函数，求一个整数 n 有几个因子
第二步：在 main 中循环 1~n 求出每个数有几个因子
*/
int fun(int n){
	int r = 0;
	int i;
	/* 不成对求，循环到 n-1 */
	/*
	for(i = 2;i <= n - 1;i++){
		if(n % i == 0){   r++;  }
	}
	*/
	// 成对求，循环到 sqrt(n)
	for(i = 2;i <= sqrt(n);i++)
	{
		if(n % i == 0)
		{	// 判断 2 个因子是否相等，不等算 2 个，相等算 1 个
			if(i != n / i) r = r + 2;
			else r = r + 1;
		}
	}
	return r;
}
int main()
{
	int i,n;
	cin>>n;
	for(i = 1;i <= n;i++)   {	cout<<fun(i)<<endl;	}
	return 0;
}
```

函数入门 - 函数修改简单问题

思路：

1. 定义变量 A，B
2. 输入 A,B
3. 判断 A 和 B 哪个大，输出较大的数

```
#include <iostream>//1-1034  javacn
using namespace std;

int main()
{
	int A,B;	// 定义变量 A，B
	cin>>A>>B;	// 输入 A,B

	if(A>B)		// 判断 A 和 B 哪个大，输出较大的数
	{
		cout<<A;
	}
	else
	{
		cout<<B;
	}
	return 0;
}
```

思路 1：1. 定义 3 个变量 a,b,c

2. 输入 a,b,c

3. 当一个数比另两个数大时它就是最大的，利用多分支解决判断

```
#include <iostream>//2-1039-1  javacn
using namespace std;
int main()
{
	int a,b,c;  // 定义变量 a,b,c
	cin>>a>>b>>c;	// 输入这 3 个整数
	// 找出最大的数
	if(a > b && a > c)	{	cout<<a;	}
	else if(b > c)		{	cout<<b;	}
		else		{	cout<<c;	}
	return 0;
}
```

思路 2：1. 定义 3 个变量 a,b,c

2. 输入 a,b,c

3. 利用”打擂台”的思想找出最大的数，定义变量 max，比较两数后存放较大的数

```
#include <iostream>//2-1039-2  javacn
using namespace std;
int main()
{
	int a,b,c;  // 定义变量 a,b,c
	cin>>a>>b>>c;	// 输入这 3 个整数
	// 定义变量 max 存放比较中较大的值
	//max 初始值等于 a
	int max = a;

	if(b > max){	max = b;	}	// 如果 b 比 max 大，把 b 赋值给 max

	if(c > max){	max = c;	}	// 如果 c 比 max 大，把 c 赋值给 max

	cout<<max;  // 输出 max
	return 0;
}
```

解法 2：自定义函数

```
#include <bits/stdc++.h>//3-1258-1  javacn  推荐学习
using namespace std;
int jc(int x)
{
	int r=1;
	int i;
	for(i=1;i<=x;i++)
	{
		r=r*i;
	}
	return r;
}
int jcsum(int x)
{
	int sum=0;
	while(x/10!=0||x%10!=0)
	{
		sum=sum+jc(x%10);
		x=x/10;
	}
	return sum;
}
int main()
{
	int i;
	for(i=100;i<=999;i++)
	{
		if(jcsum(i)==i)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//3-1258-2  javacn  推荐学习
using namespace std;
int main()
{
	int i,j,bw,sw,gw,x,y,z;
    // 因为不知道三位数是哪个     // 所以要寻找
	for(i=100;i<=999;i++)
	{

		bw = i/100;          // 对当前 i 拆位
		sw = i/10%10;
		gw = i%10;
		// 分别计算 bw、sw、gw 的阶乘
		x = 1;
		y = 1;
		z = 1;
		for(j=1;j<=bw;j++)
		{
			x = x * j;
		}
		for(j=1;j<=sw;j++)
		{
			y = y * j;
		}
		for(j=1;j<=gw;j++)
		{
			z = z * j;
		}
		if(x + y + z == i)  // 如果它们阶乘之和等于 i，则 i 就是我们要的数
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

```
#include <bits/stdc++.h>//4-1511  hyb
using namespace std;
int main()
{
	int n,s=0,c=0;
	cin>>n;
	for(int i=1; i<=n; i++)
	{
		s=0;
		int t = i;
		while(t!=0)
		{
			s = s + t%10;
			t = t/10;
		}
		if(s==13)	{	c++;	}
	}
	cout<<c;
	return 0;
}
```

解法 1：嵌套循环

```
#include <bits/stdc++.h>//5-1019-1 javacn
using namespace std;
int main(){
	int n,i,j,s=0,c;
	cin>>n;
	for(i=1;i<=n;i++)
	{
		c=1;
		for(j=1;j<=i;j++)		{	c = c*j;	}
		s+=c;
	}
	cout<<s;
	return 0;
}
```

解法 2：自定义函数

```
#include <bits/stdc++.h>//5-1019-2  javacn  推荐学习
using namespace std;
int jc(int x){
	int r=1;
	int i;
	for(i=1;i<=x;i++)	{	r=r*i;	}
	return r;
}
int main()
{
	int n;
	cin>>n;
	int i,sum=0;
	for(i=1;i<=n;i++)
	{
		sum=sum+jc(i);
	}
	cout<<sum;
	return 0;
}
```

考解法 - 单循环实现（非官方）

```
#include<bits/stdc++.h>//5-1019-3 hasome 效率最高的解法
using namespace std;
int s,n,i,t=1;
int main(){
	cin>>n;
	for(i=1;i<=n;i++)// 计算过 (i-1)！后，只要将结果再乘 i 即可
	{
		t=t*i;
		s=s+t;
	}
	cout<<s<<endl;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//6-1058-1 求出 100 至 999 范围内的所有水仙花数 javacn
using namespace std;
/*
	函数名 lfsum(int x)
	参数 int x
	返回值 int r
	说明  求各位数字立方之和
*/
int lfsum(int x)
{
	int r=0;// 用于存放和，初始值为 0
	int a;
	// 用短除法循环求各个位的立方和
	while(x/10!=0||x%10!=0)
	{
		a=x%10;
		r=r+a*a*a;
		x=x/10;
	}
	return r;
}
int main()
{
	int i;
	// 循环范围为 100 到 999 的整数
	for(i=100;i<=999;i++)
	{
		// 如果 i 的各位数字立方之和 = i 就输出 i
		if(lfsum(i)==i)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

```cpp
#include<iostream>//6-1058-2  水仙花数 jiangyf70   仅供参考
using namespace std;

void sxhs(int n)
{
	if(n <= 999)
	{
		int  a = n / 100;
		int  b = n / 10 % 10;
		int  c = n % 10;
		if(a * a * a + b * b * b + c * c * c == n)
		{
			cout << n << endl;
		}
		sxhs(n + 1);
	}
}

int main()
{
	sxhs(100);
	return 0;
}
```

思路：拆分位数，如位数与 x 相同，则 count++

```
#include<bits/stdc++.h>//7-1445  javacn
using namespace std;

int main()
{
	int x,m,n,ci,count=0;
	cin>>x>>m>>n;

	for(int i=m; i<=n; i++)
	{
		ci=i;
		while(ci!=0)
		{
			if(ci%10==x)
			{
				count++;
				break;
			}
			ci=ci/10;
		}
	}
	cout<<count;
	return 0;
}
```

自定义函数

```
#include<bits/stdc++.h>//1-1137  13202828973
using namespace std;
int a[1010];
bool sushu(int n)//判断素数函数
{
	bool f = true;
	for(int i = 2;i <= sqrt(n);i++)
	{
		if(n % i == 0)
		{
			f = false;
			break;
		}
	}
	//判断特殊情况
	if(n <= 1)		f = false;
	return f;
}

int main()
{
	int k=0;
	for(int i=1000;i<=3000;i++)
	{
		if(sushu(i)==true&&sushu(i%1000)==true&&
sushu(i%100)==true&&sushu(i%10)==true)
		{
			cout<<i<<" ";
		}
	}
	return 0;
}
```

注意判断 ?+2<=?   i+2<=n;

```
#include <bits/stdc++.h>//2-1139  javacn
using namespace std;
// 素数判断
bool sushu(int n)
{
	bool r = true;// 假设是素数
	for(int i = 2;i <= sqrt(n);i++)	// 找 n 因子范围
	{
		if(n % i == 0)
		{
			r = false;
			break;
		}
	}
	if(n <= 1) r = false;
	return r;
}

int main()
{
	int i,n;
	bool f1,f2;
	cin>>n;
	for(i = 2;i <= n;i++)
	{
		f1 = sushu(i);
		f2 = sushu(i+2);
		if(f1 == true && f2 == true && i + 2 <= n)
		{
			cout<<i<<" "<<i+2<<endl;
		}
	}
	return 0;
}
```

定义函数，求一个数各个位的和（含 1，不含 n），循环 1~n 逐个判断是否是完全数。

```
#include <bits/stdc++.h>//3-1150  javacn
using namespace std;
int sum(int n){          /*求 n 的因子和（包括 1，不含 n）*/
      int r = 0;         //存放因子和
      for(int i = 1;i <= sqrt(n);i++)
      {
            if(n % i == 0)
            {
                  if(i == n / i)//如果因子相等
                  {
                        r = r + i;
                  }
                  else
                  {
                        r = r + i + n / i;
                  }
            }
      }
      r = r - n;//去掉 n
      return r;
}
int n,c = 0;
int main(){
      cin>>n;
      for(int i = 1;i <= n;i++)
      {
            if(sum(i) == i)
            {                 //cout<<i<<endl;
                  c++;
            }
      }
      cout<<c;
      return 0;
}
```

```cpp
#include <bits/stdc++.h>//4-1513-1  13202828973
using namespace std;
int a[1010];
bool sushu(int n)
{ //判断素数函数
	bool f = true;
	for(int i = 2;i <= sqrt(n);i++)
	{
		if(n % i == 0)
		{
			f = false;
			break;
		}
	}
	//判断特殊情况
	if(n <= 1) f = false;
	return f;
}

int main()
{

	for(int i=10;i<=99;i++)
	{
		if(sushu(i)==true&&sushu(i%10*10+i/10)==true)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

思路：循环所有的 2 位数，逐个判断是否是绝对素数。

由于本题要求：两位数及其个位和十位调换位置后都是素数，因此我们定义函数判断一个数是否是素数。注意变量命名尽可能规范，函数名要见名知意。

```
#include <bits/stdc++.h>//4-1513-2  javacn
using namespace std;
int sushu(int n) // 定义函数，判断素数
{
	bool f = true;	// 用来标记 n 是否是素数，假设是素数
	int i;
	for(i=2; i<=sqrt(n); i++)
	{
		if(n%i==0)
		{
			f = false;
			break;
		}
	}
	if(n <= 1) f = false;
	return f;
}

int main()
{
	int i,c=0,x;
	for(i=10; i<=99; i++)	// 循环所有的 2 位数，逐个判断是否是绝对素数
	{
		x = i % 10 * 10 + i / 10;		// 求 i 倒过来的数
		//cout<<x<<endl;// 可以打印测试一下
		if(sushu(i)==true && sushu(x)==true)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//5-1514 javacn
using namespace std;
// 定义函数  求一个数的各个位上的和
int qiuhe(int n)
{
	int s=0;//s 变量存放数的每位的和
	// 通过短除法求
	while(n!=0)
	{
		s=s+n%10;
		n=n/10;
	}
	return s;
}
int main()
{
	int n,m;
	cin>>n;
	//n 变量如果是一位数，树根就是自己  否则要不停的求和直到是个位数
	while(n>=10)
	{
		m = qiuhe(n);
		n = m;
	}
	cout<<n<<endl;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//6-1512 javacn
using namespace std;
bool sushu(int n){      // 定义函数：判断一个整数是否是素数
      bool r = true;    // 思路：假设是素数，循环找 n 的因子，找到因子就不是素数
      int i;
      for(i = 2;i <= sqrt(n);i++)
      {
            if(n % i == 0)          // 如果找到 n 的因子
            {
                  r = false;
                  break;
            }
      }
      if(n <= 1)  {   r =false;   }     // 素数是大于 1 的自然数如果没有因子
      return r;
}
int main(){
      int i,j,m,g,s;//i 代表乙的年龄   j 代表甲的年龄
    // 因为年龄和是两位数，且甲比乙大 13   所有乙的范围是 1-86
      for(i=1;i<=86;i++)
      {
            j = i + 13;
            m = i + j;
            if(m>=10&&m<=99&&sushu(m)==true)
            {
                  g = m % 10;
                  s = m / 10;
                  if(g+s==13)
                  {
                        cout<<j<<" "<<i<<endl;
                  }
            }
      }
    return 0;
}
```

```cpp
#include<bits/stdc++.h>//7-1151  javacn
using namespace std;
bool sushu(int n)        //定义函数：判断一个整数是否是素数
{
	bool r = true;     //思路：假设是素数，循环找 n 的因子，找到因子就不是素数
	int i;
	for(i = 2;i <= sqrt(n);i++)
	{
		if(n % i == 0)          //如果找到 n 的因子
		{
			r = false;
			break;
		}
	}
	if(n <= 1)  {       r =false;   }     //素数是大于 1 的自然数如果没有因子
	return r;
}
int main(){
	int n;
	cin>>n;
	bool f = false;//假设不是桐桐数
	for(int i=2;i<=sqrt(n);i++)    //循环找 n 的因子 从 2-sqrt(n)
	{
		if(n%i==0&&sushu(i)==true&&sushu(n/i)==true)
		{
			f = true;
			break;
		}
	}
	if(f == true)
		cout<<"It’ s a Tongtong number."<<endl;
	else
		cout<<"It’ s not a Tongtong number."<<endl;
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//8-1143  javacn
using namespace std;
// 定义函数：判断一个数是否是合数
bool heshu(int n)
{
	bool f = false;// 假设不是合数
	for(int i=2; i<=sqrt(n); i++)
	{
		if(n % i ==0)
		{
			f = true;
			break;
		}
	}
	//1 既不是素数也不是合数  我们假设不是合数
	// 所有 1 2 3 这些不进循环的都是满足的  不需要单独讨论
	return f;
}
int main()
{
	int i,j,k;
	for(i=100;i<1000;i++)
	{
		j = i / 10;
		k = i / 100;
		if(heshu(i) == true && heshu(j) == true && heshu(k) == true)
		{
			cout<<i<<endl;
		}
	}
	return 0;
}
```

思路一：遍历 1 到 n 的数，逐个调用函数判断回文

```
#include <bits/stdc++.h>//9-1149-1  javacn  推荐学习
using namespace std;

// 用于反读数字，返回倒序后的 x
int fun(int x)
{
	int r=0;
	while(x!=0)
	{
		r = x%10+r*10;
		x = x/10;
	}
	return r;
}

int main()
{
	int n,i,num=0;//num 计数回文数的个数
	cin>>n;
	for(i=1;i<=n;i++)
	{
		// 如果正读和反读一样就是回文数
		if(i==fun(i))
		{
			num++;
		}
	}
	cout<<num;
	return 0;
}
```

思路二：嵌套循环，循环每个数，逐个求每个数倒过来的数，判断是否是回文。

```
#include <bits/stdc++.h>//9-1149-2 javacn
using namespace std;
int n,c = 0,s;//s 代表倒过来的数
int t;// 用来过渡 i 的值，不能直接拆 i
//1~n 中的回文数的个数
int main()
{
	cin>>n;
	for(int i = 1;i <= n;i++)
	{
		t = i;
		// 反复使用的求和变量，每次清零
		s = 0;

		// 求 i 倒过来的数
		while(t != 0)
		{
			s = s * 10 + t % 10;
			t = t / 10;
		}

		// 如果是回文
		if(i == s)
		{
			//cout<<i<<endl;
			c++;
		}
	}

	cout<<c;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//9-1149-3  w2016010182
using namespace std;
char a[10010];
bool pd(int i)
{

	int l=0;
	while(i!=0)
	{
		a[l]=char(i%10+'0');
		i=i/10;
		l++;
	}
	string st=" ",ts=" ";
	bool flag=false;

	for(int j=0;j<l;j++)
	{
		st=st+a[j];
		ts=a[j]+ts;
	}
	if(st==ts)  {     flag=true;        }
	return flag;
}
int main()
{
	int n,sum=0;
	cin>>n;
	for(int i=1;i<=n;i++)
	{
		if(pd(i)==true)    {     sum++;}//cout<<i<<endl;
	}
	cout<<sum;
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//10-1063  javacn
using namespace std;
int main()
{
	int m,n,c,s=0,i,j;//s: 素数的个数
	cin>>m>>n;
	//m~n 范围内寻找
	for(i=m;i<=n;i++)
	{
		// 对每次循环的 i 进行判断是不是素数
		// 每次判断 i 时，c=0
		c = 0; // 这句作用相当于 bool 变量，标记是否是质数

		for(j=2;j<=i-1;j++)//j: 循环其实是 i 可能的因子
		{
			if(i%j==0)
			{
				c++;
				break;
			}
		}
		// 如果条件成立，那么当前的 i 就是素数
		if(c==0 && i>=2)
		{
			s++;
		}
	}
	cout<<s;
	return 0;
}
```

求出 N 以内的全部素数，并按每行五个数显示

```
#include<bits/stdc++.h>//11-1064  javacn
using namespace std;
int main()
{
	int n,c=0,f,i,j;// 素数的个数
	cin>>n;
	for(i=1;i<=n;i++)
	{
		f = 0;// 初始因子数
		for(j=2;j<=i-1;j++)
		{
			if(i%j==0)
			{
				f ++;
				break;// 只要找到一个因子就跳出
			}
		}
		if(f==0 && i>1)// 如果 i 是一个质数
		{
			cout<<i<<" ";
			c++;
			if(c%5==0)  // 如果总个数是 5 的倍数，多输出一个换行符
			{
				cout<<endl;
			}
		}
	}
	return 0;
}
```

完数判断

思路 :1 求输入数 n 的因子和

2 如果 n 等于 n 的因子和，输出 yes

```
#include<bits/stdc++.h>//12-1859   javacn
using namespace std;

int main()
{
	int n,sum=0;
	cin>>n;
	// 求 n 的因子和
	for(int i=1;i<=n/2;i++)
	{
		if(n%i==0)
		{
			sum+=i;
		}
	}
	// 如果 n 等于 n 的因子和，输出 yes
	if(sum==n)
	{
		cout<<"yes";
	}
	else
	{
		cout<<"no";
	}
	return 0;
}
```

```cpp
#include<bits/stdc++.h>//13-1862  友好数 javacn
using namespace std;
int yinzihe(int n)// 定义函数：求一个整数的因子和（不包括本事）
{
	int r = 1;// 因子 1 是都有的 直接加进来了
	for(int i=2;i<=sqrt(n);i++)    // 所有从 2 开始找
	{
		if(n%i==0&&i!=n/i)
		{
			r = r + i + n/i;
		}
		else if(n%i==0&&i==n/i)
		{
			r = r + i;
		}
	}
	return r;
}
int main()
{
	int n,m,x,y;
	cin>>n>>m;
	x = yinzihe(n);
	y = yinzihe(m);
	if(x == m && y == n)
	{
		cout<<"yes"<<endl;
	}
	else
	{
		cout<<"no"<<endl;
	}
	return 0;
}
```

递归函数

统计每个月兔子的总数

```
#include<bits/stdc++.h>//1-1238    w2016010182
using namespace std;
long long b[50]={0};
long long tz(int  n)
{

	if(1<=n&&n<=2)
	{
		b[n]=1;
	}
	if(b[n]!=0)
	{
		return b[n];
	}
	if(b[n]==0)
	{
		return b[n]=(tz(n-1)+tz(n-2));
	}
}
int main()
{
	int n;
	long long x;
	cin>>n;
	x=tz(n);
	cout<<x;
	return 0;
}
```

```
#include<bits/stdc++.h>//2-1146          2 求 s 的值
using namespace std;
long long ss[50];
long long fun(int n)//fun 意思是函数
{
      if(n==0)           ss[n]=1;
      if(ss[n]!=0)       return ss[n];
      if(ss[n]==0)       return ss[n]=(fun(n-1)+n);
}
int main()
{
      long long s=0;
      int i=0;
      while(s<=5000)
      {
            s=s+fun(i);
            i++;
      }
      cout<<s;
      return 0;
}
```

```cpp
#include<bits/stdc++.h>//3-1147            3. 前 n 项的和
using namespace std;
long long tu[50];
long long fun(int  n)
{
      if(n==1||n==2)     tu[n]=1;
      if(tu[n]!=0)       return tu[n];
      if(tu[n]==0)       return tu[n]=(fun(n-1)+fun(n-2));
}
int main()
{
      double s=0;// 总和
      int n;
      cin>>n;
      for(int i=1;i<=n;i++)    {      s=s+fun(i)*1.0/fun(i+1);       }
      printf("%.3lf",s);//.3 保留三位数    lf 是 double 数据类型
      return 0;
}
```

汉诺塔的移动次数

通过递推发现规律： n 个金片要移动次数 = (n-1) 个金片要移动的次数 * 2 + 1

用 A(n) 表示 n 个金片移动的步数： A(n) = A(n-1) * 2 + 1

因为：

需要先将 (n-1) 个顶部的金片从 A，借助于 C，移动到 B（A(n-1) 步）

将最底层的最大金片移动到 C（1 步）

需要先将 (n-1) 个顶部的金片从 B，借助于 A，移动到 C（A(n-1) 步）

因此得出结论：

A(n) = A(n-1) * 2 + 1

```
#include <bits/stdc++.h>//4-1223 javacn
using namespace std;

// 求 n 个金片的移动次数
int fun(int n)
{
	if(n == 1)
		return 1;//1 个金片不需要递归
	else
		return fun(n-1)*2+1;
}

int main()
{
	int n;
	cin>>n;
	cout<<fun(n);
	return 0;
}
```

```cpp
#include <bits/stdc++.h>//5-1148 1989444607
using namespace std;
int num(int n)
{
	if(n==1)
		return 1;
	else
		return num(n-1)+n;
}
int main()
{
	int n;
	cin>>n;
	int s = 0;
	for(int i=1;i<=n;i++)
		s+=num(i);
	cout<<s;
	return 0;
}
```

思路：先找出每层方块的次数，然后找出规律，可以计算每层的次数。

1

1+2

1+2+3

1+2+3+4

1+2+3+4+5

一共 35 个

可以直接找到规律：x(x+1)(x+2)/6

```cpp
#include<bits/stdc++.h>//6-1145  w2016010182
using namespace std;
long long a[1005];
long long qh(int i)
{
	if(i==0)
		a[0]=1;
	if(a[i]!=0)
		return a[i];
	else
		return a[i]= qh(i-1)+i;
}

int main()
{
	int  n,i;
	long long sum=0;
	cin>>n;
	for(i=0;i<n;i++)// 所有项数和
		sum=sum+qh(i);
//	for(i=1;i<=n;i++)// 所有项数
//	{			cout<<a[i]<<endl;
//	}
	cout<<sum;
}
```

土地分割

欧几里得算法求解

把一块 M*N 米的土地分割成同样大的正方形，要求 1：没有土地剩余，2：分割出的正方形土地最大。 没有剩余就是说 M、N 都能被分割出来的正方形边长整除； 分割出来的土地最大，也就是说在可以整除 M、N 的边长中找最大的那个数 ； 结论：求 M、N 的最大公约数。 利用欧几里得算法轻松实现：（注意数据范围）

```
#include <iostream>//7-1335  anselxu
using namespace std;

long long gcd(long long x,long long y)// 辗转相除法求最大公约数
{

	if(y==0)// 除数为 0，输出 x
		return x;

	return gcd(y,x%y);// 否则，除数为被除数，余数为除数继续调用 gcd
}
int main()
{
	long long n,m;
	cin>>n>>m;
	cout<<gcd(n,m);
	return 0;
}
```