1. 质数：在大于1的整数中，如果只包含1和本身这两个约数，就被称为质数，或者叫素数

（1）质数的判定——试除法

```
//1. 试除法—时间复杂度 O(sqrt(n))

bool isprime(int n)
{
     if(n < 2)    return fasle;
     for(int i = 2; i <= n / i; i++){
         if(n % i == 0)    return false;
     }
     return true;
}


```

（2）分解质因数——试除法

```
void divide(int n)
{
    for(int i = 2; i <= n / i; i++){
        if(n % i == 0){
            int cnt = 0;
            while(n % i == 0){
                cnt++;    //以i为底的质数的个数
                n /= i;
            }
        }
    }

    if(n > 1)    cout << n << 1 << endl;
    puts("");
}
```

```

```

（3）埃式筛质数

```
const int N = 1e6 + 10;
int primes[N], cnt;
bool vis[N];

//筛出1~n之间所有的质数，并存储在primes数组中
void get_primes(int n)
{
    for(int i = 2; i <= n; i++){
        if(!vis[i]) //如果i没有被筛过就进行计算
        {
            primes[cnt++] = i;  //primes存储质数的数组
            //把i的所有倍数给筛掉
            for(int j = i + i; j <= n; j += i)  vis[j] = true;
        }
    }
}

```

（4）线性筛质数

```
const int N = 1e6 + 10;
int primes[N], cnt;
bool vis[N];

//筛出1~n之间所有的质数，并存储在primes数组中
void get_primes(int n)
{
    for(int i = 2; i <= n; i++){
        if(!vis[i]) primes[cnt++] = i;    //如果i没被筛过，就加入到primes数组中表示i是质数
        for(int j = 0; primes[j] <= n / i; j++){
            vis[primes[j] * i] = true;    //把当前质数的倍数筛掉
```

```
            if(i % primes[j] == 0)    break;    //primes[j]一定是i的最小质因子
        }
    }
}

```