# basic

dev c++ 调试时候发生软件崩溃解决办法

安装好dev cpp，准备调试的时候发现软件崩溃，这种情况很好解决。只要在工具菜单中点开编译选项，找到代码生成/优化一栏，将链接器的"产生调试信息"选项改为yes，即可

## 素数筛

只需对sqrt以下过筛即可

统计个数时应当避免重复，或者算完求前缀和

## 二分

查找一个数，有则返回地址，没有返回-1

lower_bound 返回最早能插入的位置 0 1 | 2 2 3 ->2 upper_bound 同理

传参是x,x+n,tar,cmp,返回值是指针，需要与首地址相减

二分查找bsrarch 传参是*目标，源，数目，大小，cmp

```
int binarySearch(int[] nums, int target) {
    int left = 0;
    int right = nums.length - 1;

    while(left <= right) {
        int mid = (right + left) / 2;
        if(nums[mid] == target)
            return mid;
        else if (nums[mid] < target)
            left = mid + 1;
        else if (nums[mid] > target)
            right = mid - 1;
        }
    return -1;
}
```

查找左边界，返回值也等效于小于tar的元素有几个

```
int left_bound(int[] nums, int target) {
    if (nums.length == 0) return -1;
    int left = 0;
    int right = nums.length;

    while (left < right) {
        int mid = (left + right) / 2;
```

```
        if (nums[mid] == target) {
            right = mid;
        } else if (nums[mid] < target) {
            left = mid + 1;
        } else if (nums[mid] > target) {
            right = mid;
        }
    }
    return left;
}
```

查找右边界

```
int right_bound(int[] nums, int target) {
    if (nums.length == 0) return -1;
    int left = 0, right = nums.length;

    while (left < right) {
        int mid = (left + right) / 2;
        if (nums[mid] == target) {
            left = mid + 1;
        } else if (nums[mid] < target) {
            left = mid + 1;
        } else if (nums[mid] > target) {
            right = mid;
        }
    }
    return left - 1;
```

# 二分求max的min

求最小的最大只需反转转移即可

```
while(l<=r)
{
    if(mid)
        ans=mid
        l=mid+1
    else
        r=mid-1
}
```

# 运算符重载

```
double operator * (const vec& a)const   { return x*a.x + y*a.y; }
```

# 矩阵乘法

A矩阵n行m列，B矩阵m行p列，乘完以后得到m*p大小矩阵

矩阵乘法乘一次的开销是m·n·p

```
for(int i=1;i<=m;i++)
    for(int j=1;j<=p;j++)
        for(int k=1;k<=m;k++)
            c[i][j]+=a[i][k]*b[k][j];
```

# 排列组合

```
long long fact(long long A)
{
    long long a = A;
    for (int i = a - 1; i > 0; i--)
        a *= i;

    return a;
}
//从m里面选n个
long long A(long long m, long long n)
{
    if (m < n)  return 0;
    long long ans = 1;
    for (int i = 0; i < n; i++)
        ans *= (m-i);

    return ans;
}

long long C(long long m, long long n)
{
    if (m == n) return 1;
    if (m < n)  return 0;
    long long a = A(m, n);
    long long b = A(n, n);

    return a / b;
}
```

# 质因数分解

```
int x = n;//n最多有一个质因子大于sqrt (n)
for(int i = 2;i * i <= n && x > 1;i++){//严谨性：所有非质数因子已经被分解。可以预先算
```

```
出质数表优化
while(x % i == 0){
x /= i;
printf("%d ", i);
}
}
if(x > 1) printf("%d ", x);
```

## GCD

```
int gcd(int a,int b){
    return b?gcd(b,a%b):a;
}
```

exgcd求出的x，y满足ax+by=gcd(a,b)

```
void exgcd(int &x,int &y,int a,int b)
{
    if(!b)
    {
        x=1;
        y=0;
        return;
    }
    exgcd(x,y,b,a%b);
    int t=x;
    x=y;
    y=t-a/b*y;
}
```

## 秦九昭算法

快速求多项式的值

n为最高次数，x为带入多项式的值

从低位到高位是多项式高次到低次的系数

```
double sum;
    sum = arr[0];
    for (int i = 0; i <n-1; i++)
    {
        sum = sum * x + arr[i+1];
    }
    return sum;
```