

## E U L E R

- 3的倍数和5的倍数 如果我们列出10以内所有3或5的倍数，我们将得到3、5、6和9，这些数的和是23。求1000以内所有3或5的倍数的和。

```
#include <stdio.h>
int main() {
    int sum3 = (3 + 999) * 333 / 2;
    int sum5 = (5 + 995) * 199 / 2;
    int sum15 = (15 + (1000 / 15) * 15) * (1000 / 15) / 2;
    printf("%d\n", sum3 + sum5 - sum15);
    return 0;
}
```

- 偶斐波那契数 斐波那契数列中的每一项都是前两项的和。由1和2开始生成的斐波那契数列前10项为：

1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ... 考虑该斐波那契数列中不超过四百万的项，求其中为偶数的项之和。

```
#include <cmath>
using namespace std;
#define MAX_M 4000000
int main() {
    long long sum = 0;
    int a = 0, b = 1;
    while (a + b < MAX_M) {
        b = a + b;
        a = b - a;
        if (!(b & 1)) sum += b;
    }
    printf("%lld\n", sum);
    return 0;
}

#include <cmath>
using namespace std;
#define MAX_M 4000000

int main() {
    long long sum = 0;
    int f[3] = {0, 1};
    int n = 1;
    while (f[n % 3] + f[(n - 1) % 3] < MAX_M) {
        n += 1;
        f[n % 3] = f[(n - 1) % 3] + f[(n - 2) % 3];
        if (!(f[n % 3] & 1)) sum += f[n % 3];
    }
    printf("%lld\n", sum);
    return 0;
}
```

- 最大质因数 13195的所有质因数为5、7、13和29。600851475143最大的质因数是多少？

```
using namespace std;
#define N 600851475143LL

int main() {
    long long num = N, ans, i = 2;
    while (i * i <= num) {
        if (num % i == 0) ans = i; // ans 记录最大的素因子
        while (num % i == 0) num /= i;
        i++;
    }
}
```

```

if (num != 1) ans = num; // num 为什么一定是素数
printf("%lld\n", ans);
return 0;
}

```

- 最大回文乘积 回文数就是从前往后和从后往前读都一样的数。由两个2位数相乘得到的最大回文乘积是  $9009 = 91 \times 99$ 。

找出由两个3位数相乘得到的最大回文乘积。

```

using namespace std;

int is_valid(int x, int base) {
    int raw = x, temp = 0;
    while (x) {
        // 推导这段代码的正确性
        temp = temp * base + x % base;
        x /= base;
    }
    return temp == raw;
}

int main() {
    int ans = 0;
    for (int i = 100; i < 1000; i++) {
        for (int j = i; j < 1000; j++) {
            if (is_valid(i * j, 10) && i * j > ans) ans = i * j;
        }
    }
    printf("%d\n", ans);
    return 0;
}

```

- 最小倍数 2520是最小的能够被1到10整除的数。

最小的能够被1到20整除的正数是多少？

```

#include <stdio.h>
#include <stdlib.h>
#include <inttypes.h>
int64_t gcd(int64_t a, int64_t b) {
    if (!b) return a;
    return gcd(b, a % b);
}
int main() {
    int64_t ans = 1;
    for (int64_t i = 1; i <= 20; ++i) {
        ans *= i / gcd(i, ans);
    }
    printf("%"PRIId64"\n", ans);
    return 0;
}

```

- 平方的和与和的平方之差 前十个自然数的平方的和是

$1^2 + 2^2 + \dots + 10^2 = 385$  前十个自然数的和的平方是

$(1 + 2 + \dots + 10)^2 = 55^2 = 3025$  因此前十个自然数的平方的和与和的平方之差是  $3025 - 385 = 2640$ 。

求前一百个自然数的平方的和与和的平方之差。

```
#include <cmath>
using namespace std;
int main() {
    int sum1 = 0, sum2 = 0;
    sum1 = 5050;
    sum2 = (2 * (100 * 100 * 100) + 3 * (100 * 100) + 100) / 6;
    printf("%d\n", sum1 * sum1 - sum2);
    return 0;
}
```

- 第10001个素数 列出前6个素数，它们分别是2、3、5、7、11和13。我们可以看出，第6个素数是13。

第10,001个素数是多少？

```
#include <cmath> //BL
using namespace std;

inline bool is_prime(int x) {
    if (x <= 1) return false;
    for (int i = 2; i * i <= x; i++) {
        if (x % i == 0) return false;
    }
    return true;
}

int main() {
    int cnt = 0, i = 1;
    while (cnt < 10001) {
        i++;
        if (is_prime(i)) cnt += 1;
    }
    cout << i << endl;
    return 0;
}

#include <cmath> //SSS
using namespace std;
#define MAX_N 200000
int is_prime[MAX_N + 5];

int main() {
    for (int i = 2; i * i <= MAX_N; i += 1 + (i % 2)) {
        if (is_prime[i]) continue;
        is_prime[i] = i;
        for (int j = i * i; j <= MAX_N; j += i) {
            if (is_prime[j] == 0) is_prime[j] = i;
        }
    }
    for (int i = 2; i <= MAX_N; i++) {
        if (is_prime[i]) continue;
        is_prime[+is_prime[0]] = i;
    }
    cout << is_prime[10001] << endl;
    return 0;
}

#include <cmath> //XXS
using namespace std;
#define MAX_N 100
int is_prime[MAX_N + 5];
int prime[MAX_N + 5];
```

```

int main() {
    for (int M = 2; M <= MAX_N; M++) {
        if (is_prime[M] == 0) {
            prime[++prime[0]] = M;
        }
        for (int j = 1; j <= prime[0]; j++) {
            if (prime[j] * M > MAX_N) break;
            is_prime[prime[j] * M] = 1;
            printf("%d = %d * %d\n", prime[j] * M, prime[j], M);
            if (M % prime[j] == 0) break;
        }
    }
    return 0;
}

```

- 连续数字最大乘积 在下面这个1000位正整数中，连续4个数字的最大乘积是  $9 \times 9 \times 8 \times 9 = 5832$ 。找出这个1000位正整数中乘积最大的连续13个数字。它们的乘积是多少？

```

#include <cmath>
using namespace std;
#define MAX_N 1000
char num[MAX_N + 5];

int main() {
    int len = 0;
    while (~scanf("%s", num + len)) len += strlen(num + len);
    long long p = 1, zero_cnt = 0, ans = 0;
    for (int i = 0; num[i]; i++) {
        num[i] -= '0';
        if (num[i]) p *= num[i];
        else zero_cnt += 1;
        if (i - 13 < 0) continue;
        if (num[i - 13]) p /= num[i - 13];
        else zero_cnt -= 1;
        if (zero_cnt == 0 && p > ans) ans = p;
    }
    printf("%lld\n", ans);
    return 0;
}

```

- 特殊毕达哥拉斯三元组 毕达哥拉斯三元组是三个自然数  $a < b < c$  组成的集合，并满足

$a^2 + b^2 = c^2$  例如， $3^2 + 4^2 = 9 + 16 = 25 = 5^2$ 。

有且只有一个毕达哥拉斯三元组满足  $a + b + c = 1000$ 。求这个三元组的乘积  $abc$ 。

```

#include<iostream>
#include<cmath>
using namespace std;
int gcd(int a,int b){
    return (b? gcd(b,a%b):a);
}
int main(){
    int ans=0;
    for(int n=1;n<=33;n++){
        for(int m=n+1;m*m+n*n<=1000 && !ans;m++){
            if(gcd(n,m)!=1)continue;
            int a=2*m*n;
            int b=m*m-n*n;
            int c=m*m+n*n;
            if(1000%(a+b+c)==0){
                int k=1000/(a+b+c);
                cout<<k<<endl;
                ans=a*b*c*(int)pow(k,3);
            }
        }
    }
}

```

```

        if(ans) break;
    }
    if(ans)break;
}

cout<<ans<<endl;

return 0;
}

```

- 素数的和 所有小于10的素数的和是 $2 + 3 + 5 + 7 = 17$ 。

求所有小于两百万的素数的和。

```

#include <inttypes.h>
#define MAX_RANGE 2000000
int32_t prime[MAX_RANGE + 5] = {0};
int main() {
    int64_t sum = 0;
    for (int32_t i = 2; i <= MAX_RANGE; ++i) {
        if (!prime[i]) { prime[++prime[0]] = i; sum += i; }
        for (int32_t j = 1; j <= prime[0] && prime[j] * i <= MAX_RANGE; ++j) {
            prime[i * prime[j]] = 1;
            if (i % prime[j] == 0) break;
        }
    }
    printf("%"PRIid64"\n", sum);
    return 0;
}

```

- 这四个数的乘积是 $26 \times 63 \times 78 \times 14 = 1788696$ 。

在这个 $20 \times 20$ 方阵中，四个在同一方向（从下至上、从上至下、从右至左、从左至右或者对角线）上相邻的数的乘积最大是多少？（方向数组）

```

#include <cmath>
using namespace std;
#define MAX_N 20

int dir[4][2] = {1, 0, 0, 1, 1, -1, 1, 1};
int grid[MAX_N + 5][MAX_N + 5];

int calc(int x, int y) {
    int ans = 0;
    for (int k = 0; k < 4; k++) {
        int p = 1;
        for (int step = 0; step < 4; step++) {
            int dx = x + step * dir[k][0];
            int dy = y + step * dir[k][1];
            if (dx < 0 || dx >= MAX_N) break;
            if (dy < 0 || dy >= MAX_N) break;
            p *= grid[dx][dy];
        }
        if (p > ans) ans = p;
    }
    return ans;
}

int main() {
    for (int i = 0; i < MAX_N; i++) {
        for (int j = 0; j < MAX_N; j++) {
            cin >> grid[i][j];
        }
    }
    int ans = 0;
    for (int i = 0; i < MAX_N; i++) {

```

```

        for (int j = 0; j < MAX_N; j++) {
            int p = calc(i, j);
            if (p > ans) ans = p;
        }
    }
    cout << ans << endl;
    return 0;
}

```

- 最长考拉兹序列 在正整数集上定义如下的迭代序列：

$n \rightarrow n/2$  (若n为偶数)  $n \rightarrow 3n + 1$  (若n为奇数)

从13开始应用上述规则，我们可以生成如下的序列：

13 → 40 → 20 → 10 → 5 → 16 → 8 → 4 → 2 → 1 可以看出这个序列（从13开始到1结束）共有10项。尽管还没有被证明，但我们普遍认为，从任何数开始最终都能迭代至1（“考拉兹猜想”）。

从小于一百万的哪个数开始，能够生成最长的序列呢？

注：序列开始生成后允许其中的项超过一百万。（记忆化）

```

#include <cmath>
using namespace std;
#define MAX_KEEP_SIZE 10000000

int f[MAX_KEEP_SIZE + 5];

int get_chain_length(long long x) {
    if (x == 1) return 1;
    if (x <= MAX_KEEP_SIZE && f[x] != 0) return f[x];
    int ret;
    if (x & 1) ret = get_chain_length(3 * x + 1) + 1;
    else ret = get_chain_length(x / 2) + 1;
    if (x <= MAX_KEEP_SIZE) f[x] = ret;
    return ret;
}

int main() {
    int max_len = 0, num = 0;
    for (int i = 1; i < 1000000; i++) {
        int l = get_chain_length(i);
        if (l > max_len) max_len = l, num = i;
    }
    cout << num << " " << max_len << endl;
    return 0;
}

```

- 计算出以下一百个50位数的和的前十位数字(大整数加法)

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <inttypes.h>
int main() {
    char w[55];
    int32_t ans[100] = {0}, len;
    for (int t = 0; t < 100; t++) {
        scanf("%s", w);
        len = strlen(w);
        if (len > ans[0]) ans[0] = len;
        for (int i = len - 1; i >= 0; --i){
            ans[len - i] += w[i] - '0';
        }
    }
}

```

```

}
for (int i = 1; i <= ans[0]; ++i) {
    if (ans[i] >= 10) {
        ans[i + 1] += ans[i] / 10;
        ans[i] %= 10;
        if (i + 1 > ans[0]) ans[0] = i + 1;
    }
}
for (int i = ans[0]; i > ans[0] - 10; --i) {
    printf("%d", ans[i]);
}
printf("\n");
return 0;
}

```

- 幂的数字和  $2^{15} = 32768$ , 而32768的各位数字之和是  $3 + 2 + 7 + 6 + 8 = 26$ 。

21000的各位数字之和是多少? (大整数乘法)

```

#include <cmath>
using namespace std;
#define MAX_N 400
int num[MAX_N + 5];

int main() {
    num[0] = num[1] = 1;
    for (int i = 0; i < 100; i++) {
        for (int j = 1; j <= num[0]; j++) num[j] *= 1024;
        for (int j = 1; j <= num[0]; j++) {
            if (num[j] < 10) continue;
            num[j + 1] += num[j] / 10;
            num[j] %= 10;
            num[0] += (j == num[0]);
        }
    }
    int sum = 0;
    for (int i = num[0]; i >= 1; i--) {
        sum += num[i];
    }
    printf("%d\n", sum);
    return 0;
}

```

- 最大路径和 I 从下面展示的三角形的顶端出发, 不断移动到在下一行与其相邻的元素, 能够得到的最大路径和是23。 3

```

7 4
2 4 6
8 5 9 3

```

如上图, 最大路径和为  $3 + 7 + 4 + 9 = 23$ 。 (动态规划)

```

#include <cmath>
using namespace std;
#define MAX_N 100
int val[MAX_N + 5][MAX_N + 5];

int main() {
    for (int i = 0; i < MAX_N; i++) {
        for (int j = 0; j <= i; j++) {
            cin >> val[i][j];
        }
    }
    for (int i = MAX_N - 2; i >= 0; --i) {
        for (int j = 0; j <= i; j++) {
            val[i][j] += max(val[i + 1][j], val[i + 1][j + 1]);
        }
    }
}

```

```

    }
    cout << val[0][0] << endl;
    return 0;
}

```

- 字典序排列 排列指的是将一组物体进行有顺序的放置。例如，3124是数字1、2、3、4的一个排列。如果把所有排列按照数字大小或字母先后进行排序，我们称之为字典序排列。0、1、2的字典序排列是：

012 021 102 120 201 210 数字0、1、2、3、4、5、6、7、8、9的字典序排列中第一百万位的排列是什么？(全排列状态数)

```

#include <cmath>
using namespace std;
#define MAX_N 10
int fact[MAX_N + 5];
int num[MAX_N + 5];

void init(int n) {
    fact[0] = 1;
    num[0] = 1;
    for (int i = 1; i <= n; i++) {
        fact[i] = fact[i - 1] * i;
        num[i] = 1;
    }
    return ;
}

int get_num(int k, int val, int n, int &x) {
    int step = k / val;
    x = 0;
    for (int t = 0; t <= step; x += (t <= step)) {
        t += num[x];
    }
    k %= val;
    num[x] = 0;
    return k;
}

int main() {
    init(MAX_N);
    int k = 1000000 - 1;
    for (int i = MAX_N; i >= 1; i--) {
        int x;
        k = get_num(k, fact[i - 1], MAX_N, x);
        cout << x;
    }
    cout << endl;
    return 0;
}

```