

# МАТЕМАТИКА

+  
o

Школа::Кода  
Олимпиадное  
программирование

2020-2021 Таганрог

# Модульная арифметика

Вычисления по модулю:

- Сложение:  $c = (a + b) \% \text{MOD}$ ;
- Вычитание:  $c = (a + \text{MOD} - b) \% \text{MOD}$
- Умножение:  $c = (a * b) \% \text{MOD}$
- Деление (умножение на элемент обратный в кольце по модулю):  $c = (a * b^{\text{MOD}-2}) \% \text{MOD}$

Полезная формула:

- Деление с округлением вверх:  $c = (a + b - 1) / b$

# Бинарное возведение в степень

Рекурсивная реализация

```
int Pow(int x, int y)
{
    if (y == 0)
        return 1;
    int p = Pow(x, y / 2);
    p *= p;
    if (y % 2)
        p *= x;
    return p;
}
```

Вычисление по модулю

```
const int64 MOD = 1e9 + 7;

int64 Pow(int64 x, int64 y)
{
    if (y == 0)
        return 1;
    int64 p = Pow(x, y / 2);
    p = (p * p) % MOD;
    if (y % 2)
        p = (p * x) % MOD;
    return p;
}
```

# НОД и НОК

- НОД (GCD) – наибольший общий делитель.  
Пример:  $\text{НОД}(18, 12) = 6$
- НОК (LCM) – наименьшее общее кратное.  
Пример:  $\text{НОК}(18, 12) = 36$



```
int GCD(int a, int b)
{
    if (b == 0)
        return a;
    return GCD(b, a % b);
}
```

```
int LCM(int a, int b)
{
    return a * b / GCD(a, b);
}
```

# Алгоритм Евклида

Алгоритм Евклида работает за  $O(\log \min(a, b))$

# Простые числа

- Простые числа – это натуральные числа, имеющие ровно два различных натуральных делителя  
Например, 2, 3, 5, 7, 11 и т.д.
- На олимпиадах часто встречаются задачи, для решения которых нужно определить, является ли заданное число простым.

# Наивный метод

```
bool isPrime(int n)
{
    if (n == 1)
    {
        return false;
    }
    for (int i = 2; i < n; i++)
    {
        if (n % i == 0)
        {
            return false;
        }
    }
    return true;
}
```

# Реальный метод

```
bool isPrime(int n)
{
    if (n == 1)
    {
        return false;
    }
    for (int i = 2; i * i <= n; i++)
    {
        if (n % i == 0)
        {
            return false;
        }
    }
    return true;
}
```



# Решето Эратосфена

```
vector<bool> isPrime;
vector<int> primes;

void Eratosphen(int n)
{
    isPrime.assign(n + 1, 1);
    isPrime[0] = isPrime[1] = 0;
    for (int i = 2; i < isPrime.size(); i++)
    {
        if (!isPrime[i])
            continue;
        primes.push_back(i);
        for (int j = i + i; j < isPrime.size(); j += i)
        {
            isPrime[j] = false;
        }
    }
}
```

## Нахождение всех различных делителей числа

```
vector<int> d;  
for (int i = 1; i * i <= x; ++i)  
{  
    if (x % i == 0)  
    {  
        d.push_back(i);  
        if (i * i != x)  
        {  
            d.push_back(x / i);  
        }  
    }  
}
```

## Разложение числа на простые множители

```
map<int, int> d;  
for (int i = 2; i * i <= x; ++i)  
{  
    while (x % i == 0)  
    {  
        d[i]++;  
    }  
}
```