Лабораторная работа №8

Математические основы защиты информации и информационной безопасности

Леонтьева К. А., НПМмд-02-23

7 ноября 2023

Российский университет дружбы народов

Москва, Россия

Цель лабораторной работы

1) Реализовать на языке программирования алгоритмы для выполнения арифметических операций с большими целыми числами

Считаем, что число записано в b-ичной системе счисления, b - натуральное число, $b \geq 2$. Натуральное n-разрядное число будем записывать в виде

$$u = u_1 u_2 ... u_n.$$

При работе с большими целыми числами знак такого числа удобно хранить в отдельной переменной. Например, при умножении двух чисел, знак произведения вычисляется отдельно. Квадратные скобки обозначают, что берется целая часть числа.

Ход выполнения лабораторной работы

- Реализуем алгоритм сложения неотрицательных целых чисел

```
def remove_zeros(w):
        z = 0
        while w[z] == 0:
            z = z + 1
        return(w[z:])
```

```
#Алгоритм 1
def a1(uu,vv,b):

    u = [int(i) for i in str(uu)]
    v = [int(i) for i in str(vv)]
    l = len(u)
    w = []
    n = len(u) - 1

    j = n
    k = 0
    while j != -1:
        w.append((u[j] + v[j] + k) % b)
        k = (u[j] + v[j] + k) // b
        j = j - 1
    if k != 0:
        w.append(k)
    w.reverse()
    return print(''.join(str(i) for i in remove_zeros(w)))
a1(23, 11, 10)
```

```
34
```

Figure 1: Алгоритм 1

Ход выполнения лабораторной работы

- Реализуем алгоритм вычитания неотрицательных целых чисел

```python
#Алгоритм 2
def a2(uu,vv,b):
    u = [int(i) for i in str(uu)]
    v = [int(i) for i in str(vv)]
    l = len(u)
    w = []
    n = len(u) - 1

    j = n
    k = 0
    while j != -1:
        w.append((u[j] - v[j] + k) % b)
        k = (u[j] - v[j] + k) // b
        j = j - 1
    w.reverse()
    return print(''.join(str(i) for i in remove_zeros(w)))

a2(2035, 2000, 10)
```

```
35
```

Figure 2: Алгоритм 2

Ход выполнения лабораторной работы

- Реализуем алгоритм умножения неотрицательных целых чисел столбиком

```
#Алгоритм 3
def a3(uu,vv,b):
    u = [int(i) for i in str(uu)]
    v = [int(i) for i in str(vv)]
    n = len(u) - 1
    m = len(v) - 1
    j = m
    w = [0] * (len(u) + len(v))
    while j >= 0:
        if v[j] == 0:
            w[j] == 0
            j = j - 1
        else:
            i = n
            k = 0
            while i >= 0:
                t = u[i] * v[j] + w[i + j + 1] + k
                w[i + j + 1] = t % b
                k = t // b
                i = i - 1
            w[j] = k
            j = j - 1
        z = 0
        while w[z] == 0:
            z = z + 1
    return print(''.join(str(i) for i in remove_zeros(w)))
a3(5497, 296, 10)
```

```
1627112
```

Figure 3: Алгоритм 3

Ход выполнения лабораторной работы

- Реализуем алгоритм умножения быстрым столбиком

```
#Алгоритм 4
def a4(uu,vv,b):
    u = [int(i) for i in str(uu)]
    v = [int(i) for i in str(vv)]
    n = len(u) - 1
    m = len(v) - 1
    w = [0] * (len(u) + len(v))

    t = 0
    for s in range(m + n + 2):
        for i in range(s + 1):
            if (n - i < 0) or (m - s + i < 0):
                t = t
            else:
                t = t + u[n - i] * v[m - s + i]
        w[m + n - s + 1] = t % b
        t = t // b
    return print(''.join(str(i) for i in remove_zeros(w)))

a4(5497, 296, 10)
```

```
1627112
```

Figure 4: Алгоритм 4

Ход выполнения лабораторной работы

- Реализуем алгоритм деления многоразрядных целых чисел

```
#Алгоритм 5
def a5(uu,vv,b):
    u = uu
    v = vv

    n = len([int(i) for i in str(uu)]) - 1
    t = len([int(i) for i in str(vv)]) - 1
    q = [0] * (n - t + 1)
    r = [0] * (t + 1)


    while u >= v * b ** (n - t):
        q[n-t] = q[n-t] + 1
        u = u - v * b ** (n - t)

    n = len([int(i) for i in str(u)]) - 1
    t = len([int(i) for i in str(v)]) - 1

    for i in range(n, t, -1):
        u_ = [int(i) for i in str(u)]
        u_.reverse()
        v_ = [int(i) for i in str(v)]
        v_.reverse()

        if u_[i] >= v_[t]:
            q[i-t-1] = b - 1
        else:
            q[i-t-1] = (u_[i] * b + u_[i-1]) // v_[t]
```

Figure 5: Алгоритм 5

Ход выполнения лабораторной работы

```
        while q[i-t-1] * (v_[t] * b + v_[t-1]) > u_[i] * b ** 2 + u_[i-1] * b + u_[i-2]:
            q[i-t-1] = q[i-t-1] - 1
        u = u - q[i-t-1] * b ** (i - t - 1) * v

        if u < 0:
            u = u + v * b ** (i-t-1)
            q[i-t-1] = q[i-t-1] - 1

    q.reverse()
    return print('Частное =', ''.join(str(i) for i in remove_zeros(q)), 'Остаток =', u)

a5(389725851, 79116, 10)
```

```
Частное = 4926 Остаток = 435
```

Figure 6: Алгоритм 5

Вывод

- В ходе выполнения данной лабораторной работы были реализованы алгоритмы для выполнения арифметических операций с большими целыми числами