РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

Отчёт по лабораторной работе №8

Дисциплина: *Математические основы защиты информации и информационной безопасности*

Студент: Майорова О.А., НФИмд-02-21
Преподаватель: д.ф.-м.н. Кулябов Д.С.

Москва 2021

Содержание

List of Figures

List of Tables

1 Цель работы

Цель: Ознакомиться с целочисленной арифметикой многократной точности.

2 Задание

Программно реализовать алгоритмы: сложения неотрицательных целых чисел, вычитания неотрицательных целых чисел, умножения неотрицательных целых чисел, быстрый столбик и деления многоразрядных целых чисел.

3 Теоретическое введение

Длинная арифметика — выполняемые с помощью вычислительной машины арифметические операции (сложение, вычитание, умножение, деление, возведение в степень, элементарные функции) над числами, разрядность которых превышает длину машинного слова данной вычислительной машины. Эти операции реализуются не аппаратно, а программно, с использованием базовых аппаратных средств работы с числами меньших порядков.

Длинная арифметика применяется в следующих областях:

- составление кода для процессоров (микроконтроллеров) низкой разрядности. Например, микроконтроллеры серии AVR имеют АЦП с разрядностью 10 бит и регистры с разрядностью 8 бит. Этого недостаточно для обработки информации с АЦП; без длинной арифметики не обойтись;
- криптография. Большинство систем подписывания и шифрования данных используют целочисленную арифметику по модулю m, где m — очень большое натуральное число, не обязательно простое. Например, при реализации метода шифрования RSA, криптосистемы Рабина или схемы Эль-Гамаля требуется обеспечить точность результатов умножения и возведения в степень порядка 10309;
- математическое. Результат вычисления на бумаге должен совпадать с результатом работы компьютера с точностью до последнего разряда. В частности, калькулятор Windows (начиная с Windows 95) проводит четыре арифметических действия с намного большей точностью, чем позволяет процессор x86. Для научных и инженерных расчётов длинная арифметика

применяется редко, так как ошибки во входных данных обычно намного больше, чем ошибки округления [1].

Будем считать, что число в b-ичной системе счисления, b - натуральное число, $b \geq 2$. Натуральное n-разрядное число число будем записывать в виде: $u = u_1 u_2 ... u_n$. При работе с большими целыми числами знак такого числа удобно хранит в отдельной переменной. Например, при умножении двух чисел, знак произведения вычисляется отдельно [2].

4 Выполнение лабораторной работы

Для выполнения лабораторной работы был выбран язык Python. Далее реализуем представленные алгоритмы в виде функции в соответствии с описанием из задания к лабораторной работе.

Сначала реализуем алгоритм сложения неотрицательных целых чисел:

```
def alg1(u, v, n, b):
    u = [int(x) for x in str(u)]
    v = [int(x) for x in str(v)]
    w = []
    j = n
    k = 0

    while j > 0:
        w.append((u[j-1] + v[j-1] + k) % b)
        k = (u[j-1] + v[j-1] + k) // b
        j -= 1

    w.append(k)
    w.reverse()

    return int(''.join(str(x) for x in w))
```

Результатом запуска функции будет рис. 4.1.

```
alg1(4773, 4237, 4, 10), 4773+4237
```

```
(9010, 9010)
```

```
alg1(4321, 1234, 4, 10), 4321+1234
```

```
(5555, 5555)
```

Figure 4.1: Проверка функции

Далее реализуем алгоритм вычитания неотрицательных целых чисел:

```
def alg2(u, v, n, b):
    u = [int(x) for x in str(u)]
    v = [int(x) for x in str(v)]
    w = []
    j = n
    k = 0

    while j > 0:
        w.append((u[j-1] - v[j-1] + k) % b)
        k = (u[j-1] - v[j-1] + k) // b
        j -= 1

    w.reverse()

    return int(''.join(str(x) for x in w))
```

Результатом запуска функции будет рис. 4.2.

```
alg2(4773, 4237, 4, 10), 4773-4237
```

```
(536, 536)
```

```
alg2(4321, 1234, 4, 10), 4321-1234
```

```
(3087, 3087)
```

Figure 4.2: Проверка функции

Далее реализуем алгоритм умножения неотрицательных целых чисел:

```
def alg3(u, v, b):
    u = [int(x) for x in str(u)]
    v = [int(x) for x in str(v)]
    w = [0] * len(u + v)
    j = len(v)
    k = 0

    while j > 0:
        if v[j-1] == 0:
            w[j-1] = 0
        else:
            i = len(u)
            k = 0
            while i > 0:
                t = u[i-1]*v[j-1] + w[i+j-1] + k
                w[i+j-1] = t % b
                k = t // b
                i -= 1
            w[j-1] = k
        j -= 1
```

```
    return int(''.join(str(x) for x in w))
```

Результатом запуска функции будет рис. 4.3.

```
alg3(4773, 4237, 10), 4773*4237
```

```
(20223201, 20223201)
```

```
alg3(4321, 1234, 10), 4321*1234
```

```
(5332114, 5332114)
```

Figure 4.3: Проверка функции

Далее реализуем алгоритм быстрый столбик:

```
def alg4(u, v, b):
    u = [int(x) for x in str(u)]
    v = [int(x) for x in str(v)]
    w = [0] * len(u + v)
    t = 0
    n = len(u)
    m = len(v)

    j = -n + 1
    for s in range(m + n):
        if s >= (m + n) // 2:
            r1 = j
            r2 = s+1-j
        else:
            r1 = 0
            r2 = s+1

        for i in range(r1, r2):
```

```
            t += u[n-i-1] * v[m-s+i-1]

        j += 1
        w[m+n-s-1] = t % b
        t = t // b

    return int(''.join(str(x) for x in w))
```

Результатом запуска функции будет рис. 4.4.

```
alg4(4773, 4237, 10), 4773*4237
```

```
(20223201, 20223201)
```

```
alg4(4321, 1234, 10), 4321*1234
```

```
(5332114, 5332114)
```

Figure 4.4: Проверка функции

Наконец, реализуем алгоритм деления многоразрядных целых чисел:

```
def alg5(u, v, b):
    n = len(str(u))
    t = len(str(v))
    q = [0] * (n - t + 1)

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

    ul = [int(x) for x in str(u)]
    vl = [int(x) for x in str(v)]
```

```
    i = n
    while i > t:
        if ul[i] >= vl[t]:
            q[i-t-1] = b-1
        else:
            q[i-t-1] = (ul[i]*b + ul[i-1])/vl[t]

        while q[i-t-1]*(vl[t]*b + vl[t-1]) > ul[i] * b**2 + ul[i-1]*b + ul[i-2]:
            q[i-t-1] -= 1

        u = u - q[i-t-1] * b**(i-t-1) * v
        ul = [int(x) for x in str(u)]

        if u < 0:
            u = u + v * b**(i-t-1)
            ul = [int(x) for x in str(u)]

            q[i-t-1] -= 1

        i -= 1

    return int(''.join(str(x) for x in q)), u
```

Результатом запуска функции будет рис. 4.5.

```
alg5(4773, 4237, 10), 4773//4237, 4773%4237
```

```
((1, 536), 1, 536)
```

```
alg5(4321, 1234, 10), 4321//1234, 4321%1234
```

```
((3, 619), 3, 619)
```

Figure 4.5: Проверка функции

Можно видеть, что был получен верный результат для каждого алгоритма, и функции работают корректно.

5 Выводы

Таким образом, была достигнута цель, поставленная в начале лабораторной работы. Было осуществлено знакомство с целочисленной арифметикой многократной точности. Также была получена реализация на языке Python алгоритмов сложения неотрицательных целых чисел, вычитания неотрицательных целых чисел, умножения неотрицательных целых чисел, быстрый столбик и деления многоразрядных целых чисел.

Список литературы

1. Длинная арифметика [Электронный ресурс]. Википедия: Свободная энциклопедия, 2021. URL: https://ru.wikipedia.org/w/index.php?title=Длинная_арифметика&oldid=111839679.
2. Бубнов С.А. Лабораторный практикум по основам криптографии. 2012.