Лабораторная работа №8. Целочисленная арифметика многократной точности.

Предмет: Математические основы защиты информации и информационной безопасности

Александр Сергеевич Баклашов

Содержание

List of Figures

1 Цель работы

Рассмотреть и реализовать алгоритмы для выполнения арифметических операций с большими целыми числами.

2 Задание

Реализовать алгоритмы для выполнения арифметических операций с большими целыми числами.

3 Теоретическое введение

Данные алгоритмы позволяют складывать, вычитать, умножать и делить числа различных систем счисления.

4 Выполнение лабораторной работы

4.1 Алгоритм 1 (сложение неотр. целых чисел)

4.1.1 Задача

Реализовать алгоритм для сложения неотрицательных целых чисел.

4.1.1.1 Решение

Реализуем алгоритм для сложения неотрицательных целых чисел (рис. 4.1)

In [1]:
```
# Алгоритм 1 (сложение неотр. целых чисел)
```

In [2]:
```
n = 3
b = 2
u = [1, 0, 1]
v = [0, 1, 0]
j = n - 1
k = 0
w = []

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

# Вывод результата в двоичной системе
result = ''.join(map(str, w))
print(result)
```

```
111
```

Figure 4.1: Алгоритм 1

4.2 Алгоритм 2 (вычитание неотр. целых чисел)

4.2.1 Задача

Реализовать алгоритм для вычитания неотрицательных целых чисел.

4.2.1.1 Решение

Реализуем алгоритм для вычитания неотрицательных целых чисел (рис. 4.2)

In [3]:
```
# Алгоритм 2 (вычитание неотр. целых чисел)
```

In [4]:
```
n = 3
b = 2
u = [1, 0, 1]
v = [0, 1, 0]
j = n - 1
k = 0
w = []

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

# Вывод результата в двоичной системе
result = ''.join(map(str, w))
print(result)
```

```
011
```

Figure 4.2: Алгоритм 2

4.3 Алгоритм 3 (умножение неотр. целых чисел столбиком)

4.3.1 Задача

Реализовать алгоритм для умножения целых чисел столбиком.

4.3.1.1 Решение

Реализуем алгоритм для умножения целых чисел столбиком (рис. 4.3)

In [5]:
```
# Алгоритм 3 (умножение неотр. целых чисел столбиком)
```

In [6]:
```
b = 2
n = 3
u = [1, 0, 1]
m = 3
v = [0, 1, 0]
w = [0]*(m + n)

for j in range(m-1, -1, -1):
    if v[j] == 0:
        w[j] = 0
    else:
        k = 0
        for i in range(n-1, -1, -1):
            temp_sum = u[i] * v[j] + w[i + j + 1] + k
            w[i + j + 1] = temp_sum % b
            k = temp_sum // b
            w [j] = k

result = ''.join(map(str, w))
print(result)
```

```
001010
```

Figure 4.3: Алгоритм 3

4.4 Алгоритм 4 (быстрый столбик)

4.4.1 Задача

Реализовать алгоритм для умножения целых чисел быстрым столбиком.

4.4.1.1 Решение

Реализуем алгоритм для умножения целых чисел быстрым столбиком (рис. 4.4)

```
# Алгоритм 4 (быстрый столбик)
```

```
b = 10
n = 4
u = [2, 3, 5, 5, 5]
m = 2
v = [1,0,0]
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):
            flag = 1
        else:
            temp_sum = temp_sum + u[n - i] * v[m - s + i]
    w[m + n - s + 1] = temp_sum % b
    temp_sum = temp_sum // b

result = ''.join(map(str, w))
print(result)
```

```
02355500
```

Figure 4.4: Алгоритм 4

4.5 Алгоритм 5 (деление многоразрядных целых чисел)

4.5.1 Задача

Реализовать алгоритм для деления многоразрядных целых чисел

4.5.1.1 Решение

Реализуем алгоритм для деления многоразрядных целых чисел (рис. 4.5)

In [9]:
```
# Алгоритм 5 (деление многоразрядных целых чисел)
```

In [10]:
```
b = 10
u = 101
v = 10
n = 3
t = 2

q = [0] * (n - t + 1)
r = [0] * (t + 1)

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

for i in range(n, t+2, -1):
    if u[i] >= v[t]:
        q[i-t-1] = b - 1
    else:
        q[i-t-1] = (u[i] * b + u[i - 1]) // v[t]
    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] -= 1
    u -= q[i-t-1] * (b ** (i-t-1)) * v
    if u < 0:
        u += v * (b ** (i - t -1))
        q[i-t-1] -= 1
r = u
print("q =", q[::-1])
print("r =", r)
```

```
q = [1, 0]
r = 1
```

Figure 4.5: Алгоритм 5

5 Выводы

В ходе данной лабораторной работы я рассмотрел и реализовал алгоритмы для выполнения арифметических операций с большими целыми числами.

6 Библиография

1. Python documentation. [Электронный ресурс]. М. URL: Python documentation (Дата обращения: 28.09.2023).

2. Лабораторная работа №8. Целочисленная арифметика многократной точности. - 3 с. [Электронный ресурс]. М. URL: Лабораторная работа №8. Целочисленная арифметика многократной точности. (Дата обращения: 20.12.2023).