

Отчёт по лабораторной работе №8.

Целочисленная арифметика многократной точности

*Дисциплина: Математические основы защиты информации
и информационной безопасности*

Студент: Леонова Алина Дмитриевна, 1032212306

Группа: НФИмд-01-21

Преподаватель: д-р.ф.-м.н., проф. Кулябов Дмитрий Сергеевич

25 декабря, 2021, Москва

Цель и задание работы

Цель работы

Целью данной работы является ознакомление с 5 алгоритмами для выполнения арифметических операций с большими целыми числами и их реализация.

Задание

Реализовать программно алгоритмы:

- сложение неотрицательных целых чисел
- вычитание неотрицательных целых чисел
- умножение неотрицательных целых чисел столбиком
- быстрый столбик
- деление многоразрядных целых чисел

Теоретическое введение

Рассмотрим алгоритмы для выполнения арифметических операций с большими целыми числами. Будем считать, что число записано в b -ичной системе счисления, b – натуральное число, $b \geq 2$.

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.

Алгоритм 1 (сложение неотрицательных целых чисел)

Алгоритм 1 (сложение неотрицательных целых чисел).

Вход. Два неотрицательных числа $u = u_1 u_2 \dots u_n$ и $v = v_1 v_2 \dots v_n$; разрядность чисел n ; основание системы счисления b .

Выход. Сумма $w = w_0 w_1 \dots w_n$, где w_0 – цифра переноса – всегда равная 0 либо 1.

1. Присвоить $j := n, k := 0$ (j идет по разрядам, k следит за переносом).
2. Присвоить $w_j = (u_j + v_j + k) \pmod{b}$, где w_j – наименьший неотрицательный вычет в данном классе вычетов; $k = \left\lfloor \frac{u_j + v_j + k}{b} \right\rfloor$.
3. Присвоить $j := j - 1$. Если $j > 0$, то возвращаемся на шаг 2; если $j = 0$, то присвоить $w_0 := k$ и результат: w .

Figure 1: Алгоритм 1

Алгоритм 5 (деление многоразрядных целых чисел).

Алгоритм 5 (деление многоразрядных целых чисел).

Вход. Числа $u = u_n \dots u_1 u_0$, $v = v_t \dots v_1 v_0$, $n \geq t \geq 1$, $v_t \neq 0$, разрядность чисел соответственно n и t .

Выход. Частное $q = q_{n-t} \dots q_0$, остаток $r = r_t \dots r_0$.

1. Для j от 0 до $n - t$ присвоить $q_j := 0$.
2. Пока $u \geq vb^{n-t}$, выполнять: $q_{n-t} := q_{n-t} + 1$, $u := u - vb^{n-t}$.
3. Для $i = n, n - 1, \dots, t + 1$ выполнять пункты 3.1 – 3.4:
 - 3.1 если $u_i \geq v_t$, то присвоить $q_{i-t-1} := b - 1$, иначе присвоить $q_{i-t-1} := \frac{u_i b + u_{i-1}}{v_t}$.
 - 3.2 пока $q_{i-t-1}(v_t b + v_{t-1}) > u_i b^2 + u_{i-1} b + u_{i-2}$ выполнять $q_{i-t-1} := q_{i-t-1} - 1$.
 - 3.3 присвоить $u := u - q_{i-t-1} b^{i-t-1} v$.
 - 3.4 если $u < 0$, то присвоить $u := u + v b^{i-t-1}$, $q_{i-t-1} := q_{i-t-1} - 1$.
4. $r := u$. Результат: q и r .

Figure 2: Алгоритм 5

Выполнение лабораторной работы

Реализация алгоритма 1 (сложение неотрицательных целых чисел)

```
1 # 1. Сложение неотрицательных целых чисел
2 # b - система счисления
3 def алгоритм1(u, v, b):
4     n = len(u)
5     print('* ', u, v, n, b)
6     if len(u) != len(v):
7         return print('Ошибка, числа разной разрядности')
8
9     u, v, w = list(u), list(v), []
10    k = 0
11
12    for j in range(n-1, -1, -1):
13        a = int(u[j]) + int(v[j]) + k
14        w.append(a % b)
15        k = a // b
16        #print(j, a, w, k)
17
18    w.append(k)
19    res = ''
20    for i in w[::-1]:
21        res += str(i)
22    return res
23
```

Figure 3: 1. Сложение неотрицательных целых чисел

Реализация алгоритма 2 (вычитание неотрицательных целых чисел)

```
24
25 # 2. Вычитание неотрицательных целых чисел
26 def алгоритм2(u, v, b):
27     n = len(u)
28     print('* ', u, v, n, b)
29     if int(u) < int(v):
30         return print('Ошибка, u < v')
31     if len(u) != len(v):
32         return print('Ошибка, числа разной разрядности')
33
34     u, v, w = list(u), list(v), []
35     k = 0
36
37     for j in range(n-1, -1, -1):
38         a = int(u[j]) - int(v[j]) + k
39         w.append(a % b)
40         k = a // b
41         #print(j, a, w, k)
42
43     w.append(k)
44     res = ''
45     for i in w[::-1]:
46         res += str(i)
47     return res
```

Figure 4: 2. Вычитание неотрицательных целых чисел

Реализация алгоритма 3 (умножение неотрицательных целых чисел столбиком)

```
49
50 # 3. Умножение неотрицательных целых чисел столбиком
51 def алгоритм3(u, v, b):
52     if u < v:
53         u, v = v, u
54     print('* ', u, v, b)
55
56     n = len(u)
57     m = len(v)
58     u, v = list(u), list(v)
59     w = [0] * (m + n)
60
61     for j in range(m-1, -1, -1):
62         if v[j] != 0:
63             k = 0
64             for i in range(n-1, -1, -1):
65                 t = int(u[i]) * int(v[j]) + w[i+j+1] + k
66                 w[i+j+1] = t % b
67                 k = t // b
68                 #print(j, i, t, w, k)
69
70             w[j] = k
71
72     res = ''
73     for i in w:
74         res += str(i)
75     return res
76
```

Figure 5: 3. Умножение неотрицательных целых чисел столбиком

Реализация алгоритма 4 (быстрый столбик)

```
77
78 # 4. Быстрый столбик
79 def алгоритм4(u, v, b):
80     if u < v:
81         u, v = v, u
82     print('* ', u, v, b)
83
84     t = 0
85     n = len(u)
86     m = len(v)
87     u, v = list(u), list(v)
88     w = [0] * (m + n)
89
90     for s in range(0, m + n):
91         for i in range(0, s + 1):
92             if 0 <= n-i-1 < n and 0 <= m-s+i-1 < m:
93                 t += int(u[n-i-1]) * int(v[m-s+i-1])
94
95             #print(s, i, t)
96             w[m+n-s-1] = t % b
97             t = t // b
98
99     res = ''
100     for i in w:
101         res += str(i)
102     return res
103
```

Figure 6: 4. Быстрый столбик

Промежуточные функции для 5 алгоритма

```
104
105 # словарь {символ: число}
106 str2num = {chr(i) : (i - ord('A') + 10) for i in range(ord('A'),ord('Z'))}
107 for i in '0123456789':
108     str2num[i] = int(i)
109 # словарь {число: символ}
110 num2str = {value : key for (key, value) in str2num.items()}
111
112
113 # перевод b-ичного числа в десятичное
114 def to_10(u_str, b, array = False):
115     # array = True, если число u передано в виде массива чисел
116     u_array = u_str if array else [str2num[letter] for letter in u_str]
117     u = 0
118     for i in range(len(u_array)):
119         u += (b ** i) * u_array[len(u_array) - i - 1]
120     return u
121
122
123 # перевод десятичного числа в b-ичное
124 def to_b(u, b):
125     q, r = u // b, u % b # частное q и остаток r
126     w = num2str[r]
127
128     while q >= b:
129         q, r = q // b, q % b
130         w += num2str[r]
131
132     if q != 0: w += num2str[q]
133
134     return w[::-1]
135
```

Figure 7: Промежуточные функции для 5 алгоритма

Реализация алгоритма 5 (деление многоразрядных целых чисел)

```
146
147 q = [0] * (n - t + 1)
148 # was 2
149 while u_10 >= v_10 * (b ** (n - t)):
150     q[n - t] += 1
151     u_10 -= v_10 * b ** (n - t)
152
153 u = to_b(u_10, b)
154
155 u = [str2num[letter] for letter in u]
156 v = [str2num[letter] for letter in v_str]
157
158 # was 3
159 for i in range(n, t, -1):
160     if u[n - i] >= v[0]:
161         q[i - t + 1] = b - 1
162     else:
163         q[i - t + 1] = (u[n - i] * b + u[n - i + 1]) // v[0]
164
165         while q[i - t + 1] * (v[0] * b + v[1]) > u[n - i] * (b ** 2) + u[n - i + 1] * b + u[n - i + 2]:
166             q[i - t + 1] -= 1
167
168     u_10 = to_10(u, b, True)
169     u_10 -= v_10 * q[i - t + 1] * (b ** (i - t + 1))
170
171     if u_10 < 0:
172         u_10 += v_10 * (b ** (i - t + 1))
173         q[i - t + 1] -= 1
174
175     u = to_b(u_10, b); u = [str2num[letter] for letter in u]
176
```

Figure 8: 5. Деление многоразрядных целых чисел

```
185
186 def check(f):
187     print('Результат: ',f('34','12', 10))
188     print('Результат: ',f('1234','567', 10))
189     print('Результат: ',f('67890','12345', 10))
190     print('Результат: ',f('11','10', 2))
191     print('Результат: ',f('1100101','1010101', 2))
192
193
194 print('1. Сложение неотрицательных целых чисел')
195 check(algoritm1)
196
197 print('\n2. Вычитание неотрицательных целых чисел')
198 check(algoritm2)
199
200 print('\n3. Умножение неотрицательных целых чисел столбиком')
201 check(algoritm3)
202
203 print('\n4. Быстрый столбик')
204 check(algoritm4)
205
206 print('\n5. Деление многозначных целых чисел')
207 check(algoritm5)
208 print('Результат: ',algoritm5('DEF','ABC', 16))
```

Figure 9: Функция check и проверка работы

Результат работы реализованных алгоритмов

```
In [44]: runfile('C:/GitHub/sciproc-intro/Lab_8/L8_Leonova.py', wdir='C:/GitHub/sciproc-intro/Lab_8')

1. Сложение неотрицательных целых чисел
* 34 12 2 10
Результат: 046
* 1234 567 4 10
Ошибка, числа разной разрядности
Результат: None
* 67890 12345 5 10
Результат: 080235
* 11 10 2 2
Результат: 101
* 1100101 1010101 7 2
Результат: 10111010

2. Вычитание неотрицательных целых чисел
* 34 12 2 10
Результат: 022
* 1234 567 4 10
Ошибка, числа разной разрядности
Результат: None
* 67890 12345 5 10
Результат: 055545
* 11 10 2 2
Результат: 001
* 1100101 1010101 7 2
Результат: 00010000

3. Умножение неотрицательных целых чисел
столбиком
* 34 12 10
Результат: 0408
* 567 1234 10
Результат: 0699678
* 67890 12345 10
Результат: 0838102050
* 11 10 2
Результат: 0110
* 1100101 1010101 2
Результат: 10000110001001

4. Быстрый столбик
* 34 12 10
Результат: 0408
* 567 1234 10
Результат: 0699678
* 67890 12345 10
Результат: 0838102050
* 11 10 2
Результат: 0110
* 1100101 1010101 2
Результат: 10000110001001

5. Деление многоразрядных целых чисел
* 34 12 1 1 10
Результат: 2, 10
* 1234 567 3 2 10
Результат: 02, 100
* 67890 12345 4 4 10
Результат: 5, 6165
* 11 10 1 1 2
Результат: 1, 1
* 1100101 1010101 6 6 2
Результат: 1, 10000
* DEF ABC 2 2 16
Результат: 1, 333

In [45]:
```

Figure 10: Результат выполнения L8_Leonova.py

Цель лабораторной работы была достигнута, 5 алгоритмов для выполнения арифметических операций с большими целыми числами были реализованы на языке программирования Python.