

Целочисленная арифметика многократной точности

Гебриал Ибрам Есам Зекри ¹

2022 Moscow, Russia

¹RUDN University, Moscow, Russian Federation

Цель работы

Ознакомление с алгоритмами целочисленной арифметики многократной точности, а также их последующая программная реализация.

Задачи

1. Реализовать алгоритм сложения неотрицательных целых чисел.
2. Реализовать алгоритм вычитания неотрицательных целых чисел.
3. Реализовать алгоритм умножения неотрицательных целых чисел столбиком.
4. Реализовать алгоритм быстрого столбика.
5. Реализовать алгоритм деления многоразрядных целых чисел.

Реализация

1. Написал блок данных (рис. 1)

```
2 # надо ввести данные сначала
3 u = "12345"
4 v = "56789"
5 b = 10
6 n = 5
7
```

Figure 1: Начальные данные

2. Написал алгоритм сложения неотрицательных целых чисел (рис. 2)

```
8
9 # алгоритм 1
10 j = n
11 k = 0
12
13 w = []
14 for i in range(1, n + 1):
15     w.append((int(u[n - i]) + int(v[n - i]) + k) % b)
16
17     k = (int(u[n - i]) + int(v[n - i]) + k) // b
18     j = j - 1
19 w.reverse()
20 print(w)
21
```

Figure 2: Алгоритм Сложение неотрицательных целых чисел

3. Написал алгоритм вычитания неотрицательных целых чисел (рис. 3)

```
22 # алгоритм 2
23 u = "56789"
24 v = "12345"
25
26 j = n
27 k = 0
28 w = []
29 for i in range(1, n + 1):
30     w.append((int(u[n - i]) - int(v[n - i]) + k) % b)
31     k = (int(u[n - i]) - int(v[n - i]) + k) // b
32     j = j - 1
34 w.reverse()
35 print(w)
36
```

Figure 3: Алгоритм вычитания неотрицательных целых чисел

Алгоритм умножения неотрицательных целых чисел столбиком первая часть

4. Написал алгоритм умножения неотрицательных целых чисел столбиком(рис. 4)(рис. 5)

```
37 # алгоритм 3
38 u = "123456"
39 v = "7890"
40 n = 6
41 m = 4
42 w = []
43 for i in range(m + n):
44     w.append(0)
45     j = m
46 def step6():
47     global j
48     global w
49     j = j - 1
50 if j > 0:
51     step2()
52 if j == 0:
53     print(w)
54 def step2():
55     global v
56     global w
57     global j
58 if j == m:
59     j = j - 1
60 if int(v[j]) == 0:
61     w[j] = 0
62     step6()
```

Figure 4: Алгоритм умножения неотрицательных целых чисел столбиком

Алгоритм умножения неотрицательных целых чисел столбиком первая часть

```
63 def step4():
64     global k
65     global t
66     global i
67     if i == n:
68         i = i - 1
69         t = int(u[i]) * int(v[j]) + w[i + j] + k
70         w[i + j] = t % b
71         k = t / b
72     def step5():
73         global i
74         global w
75         global j
76         global k
77         i = i - 1
78         if i > 0:
79             step4()
80         else:
81             w[j] = k
82     step2()
83     i = n
84     k = 0
85     t = 1
86     step4()
87     step5()
88     step6()
89     print(w)
90
```

Figure 5: Алгоритм умножения неотрицательных целых чисел столбиком вторая часть

5. Написал алгоритм быстрого столбика (рис. 6)

```
91 # алгоритм 4
92 u4 = "12345"
93 n = 5
94 v4 = "6789"
95 m = 4
96 b = 10
97 w1 = []
98 for i in range(m + n + 2):
99     w1.append(0)
100 t1 = 0
101 for s1 in range(0, m + n):
102     for i1 in range(0, s1 + 1):
103         if n - i1 > n or m - s1 + i1 > m or n - i1 < 0 or m - s1 + i1 < 0 or
104             m - s1 + i1 - 1 < 0:
105             continue
106         t1 = t1 + (int(u[n - i1 - 1]) * int(v[m - s1 + i1 - 1]))
107         w1[m + n - s1 - 1] = t1 % b
108         t1 = math.floor(t1 / b)
109     print(w1)
```

Figure 6: Алгоритм быстрого столбика

Алгоритм деления многоразрядных целых чисел

6. Написал алгоритм деления многоразрядных целых чисел (рис. 7)(рис. 8)

```
110 # алгоритм 5
111 u = "12346789"
112 n = 7
113 v = "56789"
114 t = 4
115 b = 10
116 q = []
117 for j in range(n - t):
118     q.append(0)
119 r = []
120 for j in range(t):
121     r.append(0)
122 while int(u) >= int(v) * (b**(n - t)):
123     q[n - t] = q[n - t] + 1
124     u = int(u) - int(v) * (b**(n - t))
125     u = str(u)
126 for i in range(n, t + 1, -1):
127     v = str(v)
128     u = str(u)
129     if int(u[i]) > int(v[t]):
130         q[i - t - 1] = b - 1
131     else:
132         q[i - t - 1] = math.floor((int(u[i]) * b + int(u[i - 1])) /
int(v[t]))
```

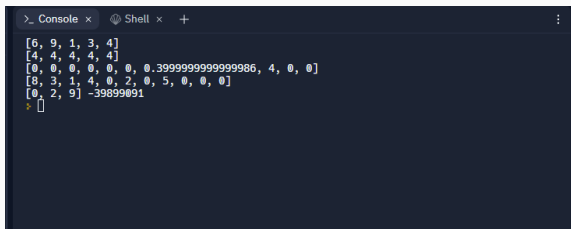
Figure 7: Алгоритм деления многоразрядных целых чисел

Алгоритм деления многоразрядных целых чисел

```
while (int(q[i - t - 1]) * (int(v[t]) * b + int(v[t - 1])) > int(u[i])
*
    (b**2) + int(u[i - 1]) * b + int(u[i - 2])):
    q[i - t - 1] = q[i - t - 1] - 1
    u = (int(u) - q[i - t - 1] * b**(i - t - 1) * int(v))
    if u < 0:
        u = int(u) + int(v) * (b**(i - t - 1))
        q[i - t - 1] = q[i - t - 1] - 1
    r = u
    print(q, r)
```

Figure 8: Алгоритм деления многоразрядных целых чисел

7. Получил результат (рис. 9)



```
>_ Console x Shell x +
[6, 9, 1, 3, 4]
[4, 4, 4, 4, 4]
[0, 0, 0, 0, 0, 0, 0.39999999999999986, 4, 0, 0]
[8, 3, 1, 4, 0, 2, 0, 5, 0, 0, 0]
[0, 2, 9] -39899091
> |
```

Figure 9: Результат алгоритмов

Изучал задачу представления больших чисел, познакомились с вычислительными алгоритмами и реализовали их.

Спасибо за внимание