

Целочисленная арифметика многократной точности

Каримов Зуфар Исматович

2022 Moscow, Russia

RUDN 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)

```
1  import math
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)

```
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
32     k = (int(u[n - i]) - int(v[n - i]) + k) // b
33     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
43 w = []
44 for i in range(m + n):
45     w.append(0)
46     j = m
47
48
49 def step6():
50     global j
51     global w
52     j = j - 1
53     if j > 0:
54         step2()
55     if j == 0:
56         print(w)
57
58
59 def step2():
60     global v
61     global w
62     global i
```

Алгоритм умножения неотрицательных целых чисел столбиком первая часть

```
70 ▼ def step4():
71     global k
72     global t
73     global i
74 ▼ if i == n:
75     i = i - 1
76     t = int(u[i]) * int(v[j]) + w[i + j] + k
77     w[i + j] = t % b
78     k = t / b
79
80
81 ▼ def step5():
82     global i
83     global w
84     global j
85     global k
86     i = i - 1
87 ▼ if i > 0:
88     step4()
89 ▼ else:
90     w[j] = k
91
92
93 step2()
94 i = n
95 k = 0
96 t = 1
97 step4()
98 step5()
99 step6()
```

5. Написал алгоритм быстрого столбика (рис. 6)

```
102 # алгоритм 4
103 u4 = "12345"
104 n = 5
105 v4 = "6789"
106 m = 4
107 b = 10
108 w1 = []
109 ▼ for i in range(m + n + 2):
110     w1.append(0)
111     t1 = 0
112 ▼ for s1 in range(0, m + n):
113     for i1 in range(0, s1 + 1):
114 ▼         if n - i1 > n or m - s1 + i1 > m or n - i1 < 0 or m - s1 + i1 < 0 or m - s1 + i1
- 1 < 0:
115             continue
116         t1 = t1 + (int(u[n - i1 - 1]) * int(v[m - s1 + i1 - 1]))
117     w1[m + n - s1 - 1] = t1 % b
118     t1 = math.floor(t1 / b)
119 print(w1)
120
```

Figure 6: Алгоритм быстрого столбика

Алгоритм деления многоразрядных целых чисел

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

```
121 # алгоритм 5
122 u = "12346789"
123 n = 7
124 v = "56789"
125 t = 4
126 b = 10
127 q = []
128 ▼ for j in range(n - t):
129     q.append(0)
130     r = []
131 ▼ for j in range(t):
132     r.append(0)
133
134 ▼ while int(u) >= int(v) * (b**(n - t)):
135     q[n - t] = q[n - t] + 1
136     u = int(u) - int(v) * (b**(n - t))
137     u = str(u)
138 ▼ for i in range(n, t + 1, -1):
139     v = str(v)
140     u = str(u)
141 ▼ if int(u[i]) > int(v[t]):
142     q[i - t - 1] = b - 1
```

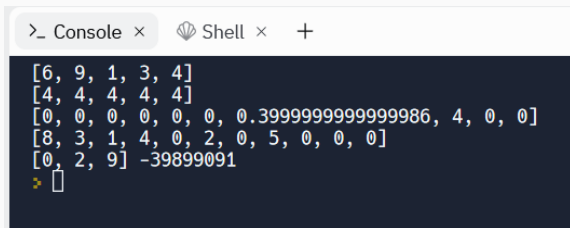
Figure 7: Алгоритм деления многоразрядных целых чисел

Алгоритм деления многоразрядных целых чисел

```
143 ▼ else:
144     q[i - t - 1] = math.floor((int(u[i]) * b + int(u[i - 1])) / int(v[t]))
145
146     while (int(q[i - t - 1]) * (int(v[t]) * b + int(v[t - 1])) > int(u[i]) *
147 ▼         (b**2) + int(u[i - 1]) * b + int(u[i - 2])):
148         q[i - t - 1] = q[i - t - 1] - 1
149     u = (int(u) - q[i - t - 1] * b**(i - t - 1) * int(v))
150 ▼     if u < 0:
151         u = int(u) + int(v) * (b**(i - t - 1))
152         q[i - t - 1] = q[i - t - 1] - 1
153     r = u
154     print(q, r)
155
```

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, 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