Отчёт

по лабораторной работе 8

Кочетов Андрей Владимирович

Содержание

List of Figures

List of Tables

1 Цель работы

Реализовать алгоритм.

2 Задание

Лабораторная работа подразумевает написание программы на языке python, которая реализует целочисленную арифметику многократной точности.

3 Выполнение лабораторной работы

1. Реализовал 1-й алгоритм

```
import math

#1
u = "25534"
v = "34789"
b = 10
n = 5
j=n
k=0
w=list()

for i in range(1,n+1):
    w.append((int(u[n-i]) + int(v[n-i])+k)%b)
    k = (int(u[n-i])+int(v[n-i]) + k) //b
    j = j-1
w.reverse()
print(w)
```

Figure 3.1: рис.1. 1-й алгоритм

2. Реализовал 2-й алгоритм.

```
#2
u = "45678"
v = "23456"

j=n
k=0
w=list()
for i in range(1,n+1):
    w.append((int(u[n-i]) + int(v[n-i])+k)%b)
    k = (int(u[n-i])+int(v[n-i]) + k) //b
    j = j-1
w.reverse()
print(w)
```

Figure 3.2: рис.2. 2-й алгоритм

3. Реализовал 3-й алгоритм

```
53
54 def step4():
55     global k
56     global t
57     global i
58     if i == n:
59         i = i - 1
60     t = int(u[i]) * int(v[j]) + w[i+j] + k
61     w[i+j] = t%b
62     k = t/b
63
64 def step5():
65     global i
66     global w
67     global j
68     global k
69     i = i - 1
70     if i > 0:
71         step4()
72     else:
73         w[j] = k
74
75 def step6():
76     global j
77     global w
78     j = j - 1
79     if j > 0:
80         step2()
81     if j == 0:
82         print(w)
83
84 step2()
85 i=0
86 k=0
87 t=1
88 step4()
89 step5
```

Figure 3.3: рис.3. 3-й алгоритм

4. Реализовал 4-й алгоритм и запустил код.

```
5  i=0
6  k=0
7  t=1
8  step4()
9  step5()
0  step6()
1  print(w)
2
3  #4
4  u = "123456789"
5  n = 7
5  v = "56789"
7  t = 4
8  b=10
9  q=list()
0  for j in range(n-t):
1      q.append(0)
2  r = list()
3  for j in range(t):
4      r.append(0)
5
6  while int(u) >= int(v)*(b**(n-t)):
7      q[n-t] = q[n-t] + 1
8      u = int(u) - int(v) * (b**(n-t))
9  u = str(u)
0  for i in range(n, t+1, -1):
1      v = str(v)
2      u = str(u)
3      if int(u[i]) > int(v[t]):
4          q[i-t-1] = b-1
5      else:
6          q[i-t-1] = math.floor((int(u[i]) * b + int(u[i-1]))/int(v[t]))
7      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])):
8          q[i-t-1] = q[i-t-1] - 1
9      u = (int(u) - q[i-t-1]*b**(i-t-1)*int(v))
0      if u < 0:
1          u = int(u) + int(v) *(b**(i-t-1))
2          q[i-t-1] = q[i-t-1] -1
3  r = u
4  print(q,r)
```

Figure 3.4: рис.4. 4-й алгоритм

4 Выводы

Я написал программный код, который реализует целочисленную арифметику многократной точности.