РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ ПО

ЛАБОРАТОРНОЙ РАБОТЕ №8

дисциплина: Математические основы защиты информации и информационной безопасности

Студент: Хиссен Али Уэддей
Группа: НПМмд-02-20
Ст. билет № 1032209306

Цель работы

Исследование алгоритмов работы с большими целыми числами.

1 Теоретические часть

В данной работе рассмотрим алгоритмы для выполнения арифметических операций с большими целыми числами. Будем считать, что число записано в b - ичной системе счисления, b – натуральное число, $b \geq 2$. Натуральное -разрядное число будем записывать в виде

$$u = u_1 u_2 \dots u_n.$$

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.

1.1 Сложение неотрицательных целых чисел

*Вход. Два неотрицательных числа $u = u_1u_2 \dots u_n$ и $v = v_1v_2 \dots v_n$; разрядность чисел n; основание системы счисления b.

*Выход. Сумма $w = w_0w_1 \dots w_n$, 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 $k = \left[\frac{u_j+v_j+k}{b}\right]$.
3. Присвоить $j = j - 1$. Если $j > 0$, то возвращаемся на шаг 2; если $j = 0$, то присвоить $w_0 = k$ и результат: w.

1.2 Вычитание неотрицательных целых чисел

*Вход. Два неотрицательных числа $u = u_1u_2 \dots u_n$ и $v = v_1v_2 \dots v_n$, $u > v$; разрядность чисел n; основание системы счисления b.

*Выход. Разность $w = w_0w_1 \dots w_n = u - v$.

1. Присвоить $j = n, k = 0$ (k – заём из старшего разряда).
2. Присвоить $w_j = (u_j - v_j + k) \pmod b$; $k = \left[\frac{u_j-v_j+k}{b}\right]$.
3. Присвоить $j = j - 1$. Если $j > 0$, то возвращаемся на шаг 2; если $j = 0$, то результат: w.

1.3 Умножение неотрицательных целых чисел столбиком

*Вход. Числа $u = u_1u_2 \dots u_n$, $v = v_1v_2 \dots v_m$; основание системы счисления b.

*Выход. Произведение $w = uv = w_1w_2 \dots w_{m+n}$.

1. Выполнить присвоения: $w_{m+1} = 0, w_{m+2} = 0, \dots, w_{m+n} = 0, j = m$ (j *перемещается по номерам разрядов числа* v *от младших к старшим*).
2. Если $v_j = 0$, то присвоить $w_j = 0$ и перейти на шаг 6.
3. Присвоить $i = n, k = 0$ (*значение* i *идет по номерам разрядов числа* u, k *отвечает за перенос*).
4. Присвоить $t = u_i \cdot v_j + w_{i+j} + k, w_{i+j} = t \pmod b, k = \left[\frac{t}{b}\right]$.
5. Присвоить $i = i - 1$. Если $i > 0$, то возвращаемся на шаг 4, иначе присвоить $w_j = k$.
6. Присвоить $j = j - 1$. Если $j > 0$, то вернуться на шаг 2. Если $j = 0$, то результат: w.

1.4 Быстрый столбик

*Вход. Числа $u = u_1u_2 \dots u_n$, $v = v_1v_2 \dots v_m$; основание системы счисления b.

*Выход. Произведение $w = uv = w_1w_2 \dots w_{m+n}$.

1. Присвоить $t = 0$.
2. Для s от 0 до $m + n - 1$ с шагом 1 выполнить шаги 3 и 4.
3. Для i от 0 до s с шагом 1 выполнить присвоение $t = t + u_{n-i} \cdot v_{m-s+i}$.

4. Присвоить $w_{m+n-s} = t \pmod b, t = \left[\frac{t}{b}\right]$. Результат: w.

1.5 Деление многоразрядных целых чисел

*Вход. Числа $u = u_n \dots u_1u_0$, $v = v_t \dots v_1v_0, n \geq t \geq 1, v_t \neq 0$.

*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 $q_{i-t-1} = \frac{u_ib+u_{i-1}}{v_t}$. 3.2. пока $q_{i-t-1}(v_tb + v_{t-1}) > u_ib^2 + u_{i-1}b + u_{i-2}$ 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 $u = u + vb^{i-t-1}$, $q_{i-t-1} = q_{i-t-1} - 1$.
4. $r = u$. Результат: q и r.

2 программная часть

2.1 Сложение неотрицательных целых чисел

```
1
2  def Algorithm1():
3
4      u=(input("faite entrer un nombre supperieur a 0 \n"))
5      v=(input("faite entrer un nombre supperieur a 0 \n"))
6      b=int((input("faite entrer un nombre supperieur a 0 \n")))
7      list_u=[i for i in u]
8
9      list_v=[i for i in v]
10
11     list_w=[]
12
13     n=len(list_u)
14
15     j=n-1
16
17     k=0
18
19     if(len(list_u)!=len(list_v)):
20         print("impossible de calculer w")
21
22     else:
23
24         w =''
25
26         while(j>=0):
27
28             resultat=int(list_u[j]) + int(list_v[j]) + k
29
30             list_w.append(resultat%b)
31
32             k=int(resultat/b)
```

```
            k=int(resultat/b)

            j=j-1

        #print(list_w)

        liste_w=[]

        ss=len(list_w)-1

        #print(ss ,k)

        while(ss>=0):

            #liste_w.append(list_w[ss])

            w+=str(list_w[ss])

            ss=ss-1
        w=str(k) + w

        #type(u[0]),list_u + list_v ,liste_w

    return int(w)
```

```
]: 1 Algorithm1()
```

```
faite entrer un nombre supperieur a 0
123
faite entrer un nombre supperieur a 0
123
faite entrer un nombre supperieur a 0
10
```

```
]: 246
```

2.2 Вычитание неотрицательных целых чисел

```
:
```

```python
def Algorithm2():
    u1=(input("faite entrer un nombre supperieur a 0 \n"))
    v1=(input("faite entrer un nombre supperieur a 0 \n"))
    b=int((input("faite entrer un nombre >2  \n")))

    list_u=[i for i in u1]

    list_v=[i for i in v1]

    list_w=[]

    n=len(list_u)

    j=n-1

    k=0

    #print(list_u)

    if(len(list_u)!=len(list_v)):
        print("impossible de calculer w")

    else:

        w =''

        while(j>=0):

            resultat=int(list_u[j]) - int(list_v[j]) + k

            list_w.append(resultat%b)

            k=int(resultat/b)
```

```
            k=int(resultat/b)

            j=j-1

        #print(list_w)

        liste_w=[]

        ss=len(list_w)-1

        #print(ss ,k)

        while(ss>=0):

            w+=str(list_w[ss])

            ss=ss-1

    return int(w)  #,int(u1)-int(v1)
```

```
Algorithm2()
```

```
faite entrer un nombre supperieur a 0
320
faite entrer un nombre supperieur a 0
300
faite entrer un nombre >2
10

20
```

2.3 Умножение неотрицательных целых чисел столбиком

```
def Algorithm3():
    u=(input("faite entrer un nombre supperieur a 0 \n"))
    v=(input("faite entrer un nombre supperieur a 0 \n"))
    b=int((input("faite entrer un nombre >2  \n")))
    list_u=[i for i in u]
    list_v=[i for i in v]
    n=len(list_u)
    m=len(list_v)
    mn=m+n
    list_w=[0]*(mn)

    for j in range(m-1, -1, -1):
        if list_v[j] != 0:
            k = 0
            for i in range (n-1, -1, -1):
                t = int(list_u[i]) * int(list_v[j]) + k + list_w[i+j+1]
                list_w[i + j + 1] = t % b
                k = t // b
            list_w[j] = k

    k=0
    j=m

    return (list_w),

```

```
Algorithm3()
```

```
faite entrer un nombre supperieur a 0
123
faite entrer un nombre supperieur a 0
10
faite entrer un nombre >2
10
```

```
([0, 1, 2, 3, 0],)
```

1.4 Быстрый столбик

```
def Algorithm4():

    u1=(input("faite entrer un nombre supperieur a 0 \n"))
    v1=(input("faite entrer un nombre supperieur a 0 \n"))
    b=int((input("faite entrer un nombre >2  \n")))

    w=''

    list_u=[i for i in u1]

    list_v=[i for i in v1]

    n=len(list_u) - 1

    m=len(list_v) - 1

    mn=m+n


    list_w=[0]*(mn+1)

    t=0

    for s in range(0,mn+1):

        for i in range(0,s+1):

                t= t+ int(list_u[n-i]) * int(list_v[m-s+i])

        #print(t)

        list_w[mn-s]=t%b
```

```
        list_w[mn-s]=t%b

        w=str(list_w[mn-s]) + w
        #list_w.appendst_w[mn-s]+w
        (t%b)

        t=t//b


    return int(w)
```

9]:
```
Algorithm4()
```

```
faite entrer un nombre supperieur a 0
100
faite entrer un nombre supperieur a 0
100
faite entrer un nombre >2
10
```

9]: 10000

1.5 Деление многоразрядных целых чисел

```
def Algorithm5():
    u1=(input("faite entrer un nombre supperieur a 0 \n"))
    v1=(input("faite entrer un nombre supperieur a 0 \n"))
    b=int((input("faite entrer un nombre >2  \n")))

    list_u=[i for i in u1]

    list_v=[i for i in v1]

    n=len(list_u) - 1

    t=len(list_v) - 1

    #ist_q=np.ones((mn+1))

    u_int,v_int=int(u1),int(v1)

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

    while(u_int>=v_int*b**(n-t)):

        list_q[n-t]=list_q[n-t] + 1

        u_int-=v_int*b**(n-t)

    for i in range(n,t,-1):
        if list_u[i] >= list_v[t]:
            list_q[i-t-1] = b-1

        else:

            list_q[i-t-1] = (list_u[i]*b + list_u[i-1] )// list_v[t]
```

```


       while list_q[i-t-1]*(list_v[t]*b + list_v[t-1]) > (list_u[i]*b**2 + list_u[i-1]*b + list_u[i-
           list_q[i-t-1] -= 1
       u_int -= list_q[i-t-1] * (b ** (i-t-1)) * v_int
   if u_int < 0:
       u_int += v_int * b**(i-t-1)
       list_q[i-t-1] -= 1

   q = int("".join(map(str, list_q)))
   r = u_int

   print(list_q,f"q = {q} r = {r}")
```

```
Algorithm5()
```

```
faite entrer un nombre supperieur a 0
55
faite entrer un nombre supperieur a 0
25
faite entrer un nombre >2
```

Вывод

так мы изучили исследование алгоритмов работы с большими целыми, познакомились с вычислительными алгоритмами.