Отчёт по лабораторной работе №8. Целочисленная арифметика многократной точности

Бармина Ольга

2024 September 8th

Российский университет дружбы народов, Москва, Россия

Цели и задачи работы

Цель лабораторной работы

Целью данной лабораторной работы является ознакомление с алгоритмами по воплощению целочисленной арифметики многократной точности, а также программная реализация данных алгоритмов.

Задание

Реализовать рассмотренные в инструкции к лабораторной работе алгоритмы программно.

Алгоритмы:

1. Сложение неотрицательных целых чисел
2. Вычитание неотрицательных целых чисел
3. Умножение неотрицательных целых чисел столбиком
4. Быстрый столбик
5. Деление многоразрядных целых чисел

Ход выполнения и результаты

Вспомогательные действия

```
str2num = {chr(l_ord): (l_ord-ord('A')+10) for l_ord in range(ord('A'),ord('Z')+1)}
for n in '0123456789':
    str2num[n] = int(n)
num2str = {v: k for (k,v) in str2num.items()}

def add_0(u, n,f):
    res = [0]*(n-len(u))
    if f:
        res.extend(u)
        return res
    return "".join([num2str[i] for i in res])

def make_i(u_s, v_s, f=False,f2=True):
    u = [str2num[l] for l in u_s]
    v = [str2num[l] for l in v_s]

    if f:
        if len(u) != len(v):
            if len(u) < len(v):
                u = add_0(u, len(v),f2)
            else:
                v = add_0(v, len(u),f2)
    return u,v
```

Figure 1: Вспомогательные действия для удобства дальнейших вычислений

Алгоритм 1. Сложение неотрицательных целых чисел. Реализация

```python
def add(u,v,b):
    u,v = make_i(u,v, True)
    n = len(u)
    k = 0
    w = []
    for j in range(n-1, -1, -1):
        w.append((u[j]+v[j]+k)%b)
        k = (u[j]+v[j]+k)//b
    w.append(k)
    w.reverse()
    return "".join([num2str[i] for i in w])
```

```python
add("109","452",10)
```

```
'0561'
```

```python
add("109","452",16)
```

```
'055B'
```

Figure 2: Алгоритм 1. Сложение неотрицательных целых чисел

Алгоритм 2. Вычитание неотрицательных целых чисел. Реализация

```
def subtract(u,v,b):
    u,v = make_i(u,v, True)
    n = len(u)
    k = 0
    w = []
    for j in range(n-1, -1, -1):
        w.append((u[j]-v[j]+k)%b)
        k = (u[j]-v[j]+k)//b
    w.append(k)
    w.reverse()
    return "".join([num2str[i] for i in w])
```

```
subtract("865","127",10)
```

```
'0738'
```

```
subtract("865","127",16)
```

```
'073E'
```

Figure 3: Алгоритм 2. Вычитание неотрицательных целых чисел

Алгоритм 3. Умножение неотрицательных целых чисел столбиком. Реализация

```python
def multiply(u,v,b):
    u,v = make_i(u,v, False)
    n = len(u)
    m = len(v)
    w = [0] * (m+n)
    for j in range(m-1,-1,-1):
        if v[j] != 0 :
            k = 0
            for i in range(n-1,-1,-1):
                t = u[i]*v[j] + w[i+j+1] + k
                w[i+j+1] = t % b
                k = t // b
            w[j] = k
    return "".join([num2str[i] for i in w])
```

```python
multiply("15","12",10)
```

```
'0180'
```

```python
multiply("A81","C",16)
```

```
'7E0C'
```

Figure 4: Алгоритм 3. Умножение неотрицательных целых чисел столбиком

Алгоритм 4. Быстрый столбик. Реализация

```python
def multiply_fast(u,v,b):
    u,v = make_i(u,v, False)
    n = len(u)
    m = len(v)
    w = [0] * (m+n)
    t = 0
    for s in range(0, n+m-1):
        for i in range(0, s+1):
            t = t+u[n-i-1]*v[m-s+i-1]
        w[m+n-s-1] = t%b
        t = t//b
    return "".join([num2str[i] for i in w])
```

```python
multiply_fast("15","12",10)
```

```
'0180'
```

```python
multiply_fast("150","52",10)
```

```
'07800'
```

Figure 5: Алгоритм 4. Быстрый столбик

Алгоритм 5. Деление многоразрядных целых чисел. Реализация

```python
def to10(u_s, b, f=False):
    u_tmp = u_s if f else [str2num[l] for l in u_s]
    u = 0
    for i in range(len(u_tmp)):
        u += b**i * u_tmp[len(u_tmp)-i-1]
    return u

def tob(u, b, n=1):
    q, r = u//b, u%b
    w = num2str[r]
    while q >= b:
        q, r = q//b, q%b
        w += num2str[r]
    if q != 0:
        w += num2str[q]
    while len(w)<n:
        w += '0'
    return w[::-1]
```

Figure 6: Алгоритм 5. Деление многоразрядных целых чисел

Алгоритм 5. Деление многоразрядных целых чисел. Реализация

```
def divide(u_s,v_s,b):
    u = u_s
    v = v_s
    u_10 = to10(u,b)
    v_10 = to10(v,b)
    n = len(u)-1
    t = len(v)-1
    if v == '0': return 'impossible'

    q = [0]*(n-t+1)
    while u_10>=v_10 * (b**(n-t)):
        q[n-t] += 1
        u_10 -= v_10 * (b**(n-t))

    u = tob(u_10,b,n+1)
    u,v = make_i(u,v_s)

    for i in range(n,t,-1):
        if u[n-i]>=v[0]:
            q[i-t-1] = b-1
        else:
            q[i-t-1] = (u[n-i]*b + u[n-i-1])//v[0]
        while q[i-t-1]*(v[0]*b+v[1]) > u[n-i]*b*b + u[n-i+1]*b + u[n-i+2]:
            q[i-t-1] -= 1
        u_10 = to10(u,b,True)
        u_10 -= v_10*q[i-t-1]*(b**(i-t-1))
        if u_10 < 0:
            u_10 += v_10*(b**(i-t-1))
            q[i-t-1] -= 1
        u = tob(u_10,b,n+1)
        u = [str2num[l] for l in u]
    return "".join([num2str[i] for i in q[::-1]]), "".join([num2str[i] for i in u])
```

Алгоритм 5. Деление многоразрядных целых чисел. Результат

```
divide('1000','15',10)
```

```
('066', '0010')
```

```
divide('81','27',10)
```

```
('3', '00')
```

```
divide('81','0',10`)
```

```
'impossible'
```

```
divide('81','82',10)
```

```
('0', '81')
```

Figure 8: Алгоритм 5. Деление многоразрядных целых чисел

Вывод

В результате выполнения данной лабораторной работы нам удалось осуществить программно алгоритмы, рассмотренные в описании к лабораторной работе.