Многозначная арифметика

Этот документ содержит алгоритмы для работы с многозначными числами. Каждый алгоритм реализован на языке Python с комментариями и пояснениями.

Сложение многозначных чисел

Функция принимает два числа, представленных массивами цифр, и основание системы счисления. Возвращает сумму в виде массива цифр.

```
def сложение(u, v, b):
    n = max(len(u), len(v))
    u = [0] * (n - len(u)) + u  # Добавляем нули для выравнивания длины
    v = [0] * (n - len(v)) + v  # Добавляем нули для выравнивания длины
    w = []
    перенос = 0

    for i in range(n - 1, -1, -1):
        сумма = u[i] + v[i] + перенос
        w.insert(0, сумма % b)  # Остаток от деления на основание
        перенос = сумма // b   # Целая часть для переноса

    if перенос > 0:
        w.insert(0, перенос)  # Добавляем перенос, если он есть

    return w
```

Вычитание многозначных чисел

Функция принимает два числа, представленных массивами цифр, и основание системы счисления. Возвращает разность в виде массива цифр.

```
def вычитание(u, v, b):
    n = max(len(u), len(v))
    u = [0] * (n - len(u)) + u
    v = [0] * (n - len(v)) + v
    w = []
    займ = 0

    for i in range(n - 1, -1, -1):
        разность = u[i] - v[i] - займ
        if разность < 0:
            разность += b  # Берем в долг из следующего разряда
            займ = 1
        else:
```

```
            займ = 0
        w.insert(0, разность)

    # Убираем ведущие нули
    while len(w) > 1 and w[0] == 0:
        w.pop(0)

    return w
```

Умножение многозначных чисел (столбиком)

Функция принимает два числа, представленных массивами цифр, и основание системы счисления. Возвращает произведение в виде массива цифр.

```
def умножение(u, v, b):
    n, m = len(u), len(v)
    w = [0] * (n + m)  # Результат размером n + m

    for i in range(n - 1, -1, -1):
        перенос = 0
        for j in range(m - 1, -1, -1):
            произведение = u[i] * v[j] + w[i + j + 1] + перенос
            w[i + j + 1] = произведение % b  # Остаток от деления
            перенос = произведение // b  # Перенос в следующий разряд
        w[i + j] += перенос  # Добавляем оставшийся перенос

    # Убираем ведущие нули
    while len(w) > 1 and w[0] == 0:
        w.pop(0)

    return w
```

Быстрое умножение (метод Карацубы)

Быстрое умножение для многозначных чисел. Использует разбиение числа на части для ускорения вычислений.

```
def быстрое_умножение(u, v, b):
    if len(u) == 1 or len(v) == 1:
        return умножение(u, v, b)  # Используем базовое умножение для коротких
чисел

    m = max(len(u), len(v)) // 2
    u_старшая, u_младшая = u[:-m], u[-m:]
    v_старшая, v_младшая = v[:-m], v[-m:]

    z0 = быстрое_умножение(u_младшая, v_младшая, b)  # Младшая * Младшая
```

```
    z1 = быстрое_умножение(сложение(u_старшая, u_младшая, b), сложение(v_старшая,
v_младшая, b), b)
    z2 = быстрое_умножение(u_старшая, v_старшая, b)

    # Комбинируем результаты
    результат = сложение(сложение(z2 + [0] * (2 * m), вычитание(z1, сложение(z2,
z0, b), b) + [0] * m, b), z0, b)
    return результат
```

Деление многозначных чисел

Функция принимает два числа, представленных массивами цифр, и основание системы счисления. Возвращает частное и остаток в виде массивов.

```
def деление(u, v, b):
    n, m = len(u), len(v)
    if m == 0 or (m == 1 and v[0] == 0):
        raise ValueError("Деление на ноль невозможно")

    q = [0] * (n - m + 1)  # Частное
    r = u[:]  # Остаток

    # Нормализация
    v = [0] * (n - m) + v

    for i in range(n - m + 1):
        q[i] = (r[i] * b + (r[i + 1] if i + 1 < n else 0)) // v[i]
        for j in range(m):
            r[i + j] -= q[i] * v[j]
        while r[i] < 0:
            q[i] -= 1
            for j in range(m):
                r[i + j] += v[j]

    return q, r
```

##Примеры использования Сложение:

```
u = [4, 3, 2, 1]  # Число 1234
v = [7, 8, 9]     # Число 987
b = 10
print(сложение(u, v, b))  # Результат: [5, 2, 2, 1] (2221)
```

Вычитание:

```
u = [4, 3, 2, 1]
v = [7, 8, 9]
```

```
b = 10
print(вычитание(u, v, b))  # Результат: [3, 4, 3, 2] (3432)
```

Умножение:

```
u = [4, 3, 2, 1]
v = [7, 8, 9]
b = 10
print(умножение(u, v, b))  # Результат: [3, 5, 2, 2, 1, 0, 9]
```