Шаблон отчёта по лабораторной работе №8

Дисциплина: Математические основы защиты информации и информационной безопасности

Миличевич Александра

Содержание

# Цель работы

### Арифметика в системах счисления: Сложение , Вычитание, Умножение и деление

Этот документ описывает реализацию алгоритмов сложения и вычитания неотрицательных целых чисел в заданной системе счисления.

### 1. Функция addition(num1, num2, base)

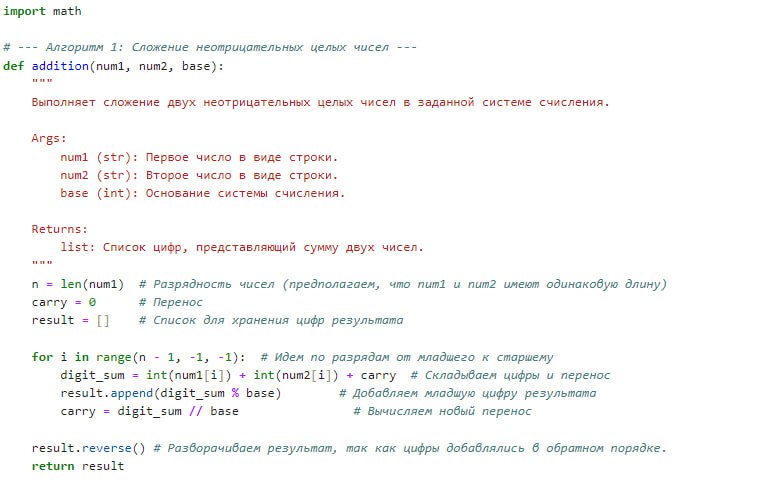
Эта функция выполняет сложение двух неотрицательных целых чисел в заданной системе счисления.

#### Описание:

* **Вход:**
  + num1 (str): Первое число в виде строки.
  + num2 (str): Второе число в виде строки.
  + base (int): Основание системы счисления.
* **Выход:**
  + list: Список цифр, представляющий сумму двух чисел.

#### Как работает:

1. **Инициализация:**
   * Определяется разрядность чисел n (длина num1, предполагается, что num1 и num2 имеют одинаковую длину).
   * Переменная carry (перенос) устанавливается в 0.
   * Создается пустой список result для хранения цифр результата.
2. **Цикл по разрядам:**
   * Цикл for проходит по разрядам чисел от младшего к старшему (справа налево).
   * На каждой итерации:
     + Вычисляется сумма цифр на текущем разряде и перенос: digit\_sum = int(num1[i]) + int(num2[i]) + carry.
     + Младшая цифра результата добавляется в список result: result.append(digit\_sum % base).
     + Вычисляется новый перенос: carry = digit\_sum // base.
3. **Разворот результата:** Список result разворачивается, так как цифры добавлялись в обратном порядке.
4. **Возврат результата:** Возвращается список result, представляющий сумму чисел.



addition

### 2. Функция subtraction(num1, num2, base)

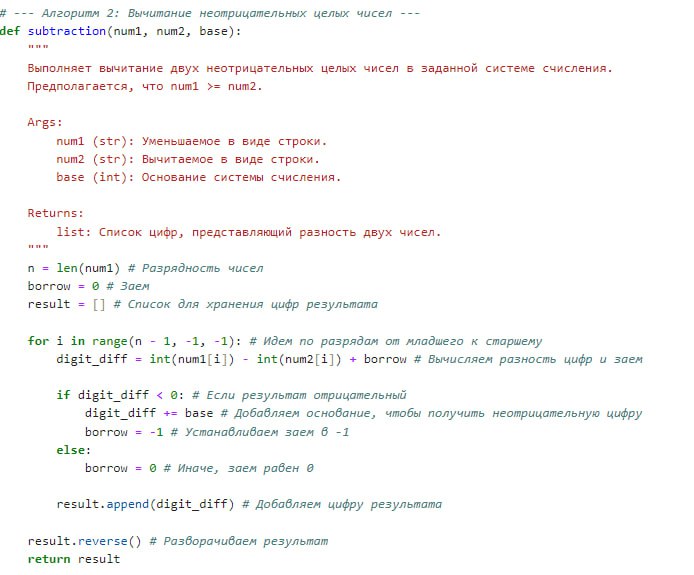
Эта функция выполняет вычитание двух неотрицательных целых чисел в заданной системе счисления. Предполагается, что num1 >= num2.

#### Описание:

* **Вход:**
  + num1 (str): Уменьшаемое в виде строки.
  + num2 (str): Вычитаемое в виде строки.
  + base (int): Основание системы счисления.
* **Выход:**
  + list: Список цифр, представляющий разность двух чисел.

#### Как работает:

1. **Инициализация:**
   * Определяется разрядность чисел n (длина num1).
   * Переменная borrow (заем) устанавливается в 0.
   * Создается пустой список result для хранения цифр результата.
2. **Цикл по разрядам:**
   * Цикл for проходит по разрядам чисел от младшего к старшему (справа налево).
   * На каждой итерации:
     + Вычисляется разность цифр на текущем разряде и заем: digit\_diff = int(num1[i]) - int(num2[i]) + borrow.
     + Если digit\_diff отрицателен:
       - К digit\_diff добавляется основание, чтобы получить неотрицательную цифру: digit\_diff += base.
       - Устанавливается заем в -1: borrow = -1.
     + Иначе (если digit\_diff неотрицателен):
       - Устанавливается заем в 0: borrow = 0.
     + Цифра результата добавляется в список result: result.append(digit\_diff).
3. **Разворот результата:** Список result разворачивается.
4. **Возврат результата:** Возвращается список result, представляющий разность чисел.



substruction

### 3. Функция multiplication(num1, num2, base)

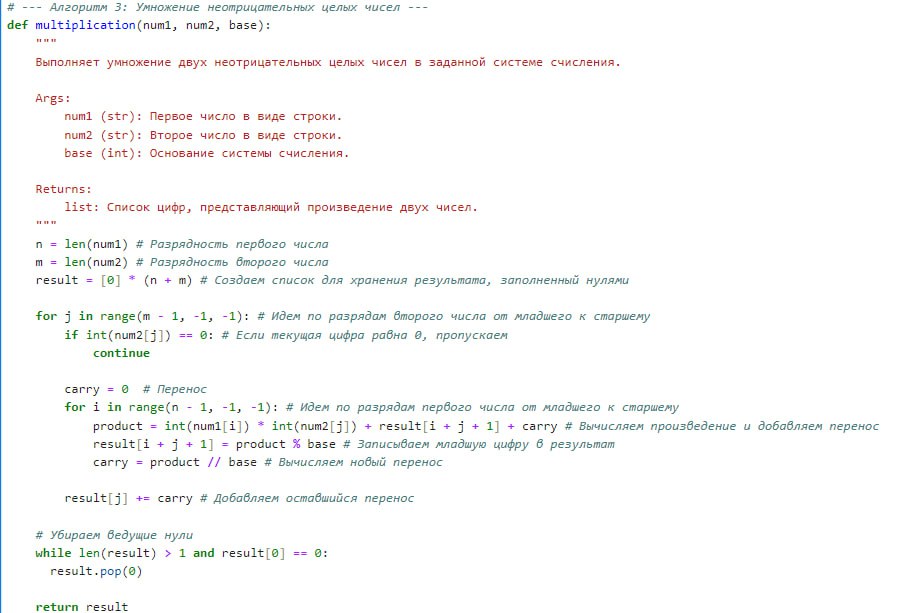
Эта функция выполняет умножение двух неотрицательных целых чисел в заданной системе счисления.

#### Описание:

* **Вход:**
  + num1 (str): Первое число в виде строки.
  + num2 (str): Второе число в виде строки.
  + base (int): Основание системы счисления.
* **Выход:**
  + list: Список цифр, представляющий произведение двух чисел.

#### Как работает:

1. **Инициализация:**
   * Определяется разрядность чисел n и m (длины num1 и num2 соответственно).
   * Создается список result длиной n + m, заполненный нулями, для хранения результата.
2. **Внешний цикл по разрядам второго числа:**
   * Цикл for проходит по разрядам num2 от младшего к старшему (справа налево).
   * Если текущая цифра num2[j] равна 0, то итерация пропускается.
3. **Внутренний цикл по разрядам первого числа:**
   * Цикл for проходит по разрядам num1 от младшего к старшему.
   * На каждой итерации:
     + Вычисляется произведение цифр на текущих разрядах и добавляется к предыдущему результату и переносу: product = int(num1[i]) \* int(num2[j]) + result[i + j + 1] + carry.
     + Младшая цифра произведения записывается в result[i + j + 1]: result[i + j + 1] = product % base.
     + Вычисляется новый перенос: carry = product // base.
4. **Добавление переноса:** Оставшийся перенос carry добавляется к result[j].
5. **Удаление ведущих нулей:** Убираются ведущие нули из result (если есть).
6. **Возврат результата:** Возвращается список result, представляющий произведение чисел.



multiplication

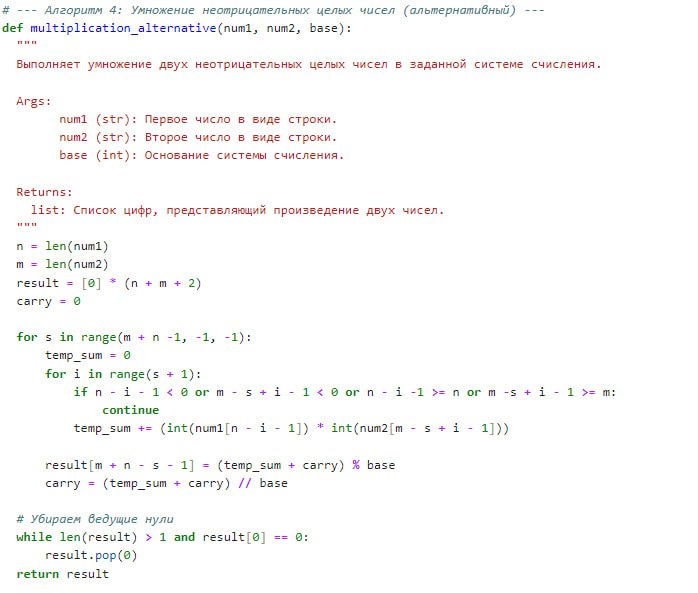
### 4. Функция multiplication\_alternative(num1, num2, base)

Эта функция выполняет альтернативный алгоритм умножения двух неотрицательных целых чисел в заданной системе счисления.

#### Описание:

* **Вход:**
  + num1 (str): Первое число в виде строки.
  + num2 (str): Второе число в виде строки.
  + base (int): Основание системы счисления.
* **Выход:**
  + list: Список цифр, представляющий произведение двух чисел. #### Как работает:

1. **Инициализация:**
   * Определяется разрядность чисел n и m (длины num1 и num2 соответственно).
   * Создается список result длиной n + m + 2, заполненный нулями, для хранения результата.
   * Переменная carry устанавливается в 0.
2. **Внешний цикл по суммам разрядов:**
   * Цикл for проходит по всем возможным суммам разрядов.
   * На каждой итерации вычисляется temp\_sum как сумма произведений соответствующих цифр из num1 и num2.
   * Текущая цифра результата вычисляется как (temp\_sum + carry) % base и записывается в result.
   * carry обновляется как (temp\_sum + carry) // base.
3. **Удаление ведущих нулей:** Убираются ведущие нули из result (если есть).
4. **Возврат результата:** Возвращается список result, представляющий произведение чисел.



multiplication alternative

### 5. Функция division(dividend, divisor, base)

Эта функция выполняет деление неотрицательных целых чисел в заданной системе счисления.

#### Описание:

* **Вход:**
  + dividend (str): Делимое в виде строки.
  + divisor (str): Делитель в виде строки.
  + base (int): Основание системы счисления.
* **Выход:**
  + tuple: Кортеж, содержащий частное (list) и остаток (str).

#### Как работает:

1. **Инициализация:**
   * Определяется длина делимого dividend\_len и делителя divisor\_len.
   * Проверяется, что делитель не равен нулю.
   * Проверяется, что делимое больше или равно делителя.
   * Создается список quotient (частное) и присваивается remainder значение dividend.
2. **Предварительное деление:**

* Выполняется проверка, на случай, если делимое больше делителя на целую степень base.

1. **Основной цикл деления:**
   * Цикл проходит по разрядам делимого (справа налево).
   * На каждой итерации:
   * Проверяется, что текущий разряд делимого входит в границы остатка.
   * Определяется начальное значение для цифры частного.
   * Цикл while уточняет значение цифры частного.
   * Обновляется остаток с учетом новой цифры частного.
2. **Возврат результата:** Возвращается кортеж, содержащий частное (quotient) и остаток (remainder).



division

# Вывод

Эта лабораторная описывает функции addition, subtraction, multiplication, multiplication\_alternative, и division, которые выполняют сложение, вычитание, умножение и деление неотрицательных целых чисел в заданной системе счисления. Эти функции используют стандартные алгоритмы для выполнения этих операций, работая с числами, представленными в виде строк цифр.