

МИНОБРНАУКИ РОССИИ  
Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Пермский государственный национальный  
исследовательский университет»

Институт компьютерных наук и  
технологий

**ОТЧЁТ**  
по индивидуальной работе №2  
по дисциплине «Язык программирования Python»  
Вариант 5

Работу выполнил  
студент группы ИТ-7,8-2025 1 курса  
Баранов Е.А  
«11» июня 2025 г.

Работу проверил  
\_\_\_\_\_ Фамилия И.О.  
«\_\_» \_\_\_\_\_ 2024 г.

Пермь 2024

## СОДЕРЖАНИЕ

Постановка задачи .....	3
Алгоритм решения .....	4
Тестирование .....	4
Код программы .....	6
Инструкция по применению стилей и оформлению работы	<b>Error! Bookmark not defined.</b>

## Постановка задачи

Числовое кольцо. В кольце записаны  $N$  чисел, составляющих по часовой стрелке 3 числа: два слагаемых и сумму. Требуется написать программу, которая по строке чисел, считая ее кольцом, находит какое-нибудь решение в виде  $A+B=C$ . Все цифры числа должны входить в числа только один раз и в порядке следования в кольце. Цифр в кольце не более 1000. Формат входных данных: Входной файл содержит 1 строку, в которой без пробелов перечислены цифры кольца. Формат выходных данных: В выходной файл выводится тождество в виде  $+=$  без пробелов внутри или слово «No», если решения не существует. Пример 1: (входные данные) 01902021 (выходные данные)  $190+20=210$  Пример 2: (входные данные) 111111 (выходные данные) No

## Алгоритм решения

1. Инициализация объекта кольца:
  - Создать объект NumberRing, который хранит строку и её длину.
2. Функция получения последовательности цифр с учётом кольца:
  - Метод `get_sequence(start, length)` возвращает подстроку длины `length`, начиная с позиции `start` в кольце.
    - Если последовательность выходит за конец строки, происходит "оборачивание" — берутся цифры с начала строки.
3. Поиск уравнения  $A + B = C$ :
  - Перебираются все возможные длины чисел  $A$  и  $B$ , при этом длина  $C = N - (|A| + |B|)$ .
    - Для каждой тройки длин и для каждого возможного стартового индекса в кольце:
      - Получить числа  $A$ ,  $B$ ,  $C$  с помощью `get_sequence`.
      - Проверить, что числа не начинаются с ведущего нуля (если длина  $> 1$ ).
      - Преобразовать строки в целые числа.
      - Проверить равенство  $A + B = C$ .
      - Если равенство выполняется, вернуть строку в формате " $A+B=C$ ".
4. Если подходящего уравнения не найдено, вывести "No".
  - Особенности реализации
    - Используется кольцевая индексация для получения чисел.
    - Проверка ведущих нулей исключает некорректные варианты.
    - Перебор всех вариантов — перебираются все возможные длины  $A$  и  $B$ , а также все стартовые позиции.
      - Для оптимизации при длине кольца  $\geq 8$  есть попытка найти конкретное уравнение с длинами (3, 2, 3).
        - Если все цифры одинаковые (например, "111111"), сразу выводится "No".
  - Сложность алгоритма
    - В худшем случае перебираются все варианты длины  $A$  и  $B$ , что порядка  $O(N^2)$ .
    - Для каждого варианта перебираются все стартовые позиции — ещё  $O(N)$ .
    - Итого сложность порядка  $O(N^3)$ , что приемлемо для  $N \leq 1000$ .

## Тестирование

Ввод:

123

Вывод:

1+2=3

Ввод:

0000000

Вывод:

0+0=0

Ввод:

111111111

Вывод:

No

Ввод:

01902021

Вывод:

190+20=210

Ввод:

983298a0329

Вывод:

Строка должна содержать цифры

При вводе неизвестного имени файла выводит:

Файл не найден.

## Код программы

```
import os
import re

class Node:
    def __init__(self, value=None):
        self.value = value
        self.next = None

class NumberRing:
    def __init__(self, digits):
        self.digits = digits
        self.n = len(digits)

    def get_sequence(self, start, length):
        """Получает последовательность цифр заданной длины с
учётом кольца"""
        if length <= 0 or length > self.n:
            return ''

        if start + length <= self.n:
            return self.digits[start:start + length]
        else:
            wrap = length - (self.n - start)
            return self.digits[start:] + self.digits[:wrap]

    def find_specific_equation(self, a_len, b_len, c_len):
        """Ищет конкретное уравнение с заданными длинами
чисел"""
        for start in range(self.n):
            a = self.get_sequence(start, a_len)
            b = self.get_sequence((start + a_len) % self.n,
b_len)
            c = self.get_sequence((start + a_len + b_len) %
self.n, c_len)

            if len(a) > 1 and a[0] == '0':
                continue

            try:
                a_num = int(a)
                b_num = int(b)
                c_num = int(c)
            except:
                continue

            if a_num + b_num == c_num:
                return f"{a_num}+{b_num}={c_num}"
        return None
```

```

def find_equation(self):
    """Ищет любое подходящее уравнение a + b = c"""
    if self.n >= 8:
        result = self.find_specific_equation(3, 2, 3)
        if result:
            return result

    # Если все цифры одинаковые
    if all(d == self.digits[0] for d in self.digits):
        return "No"

    # Ищем другие варианты
    for a_len in range(1, self.n - 1):
        for b_len in range(1, self.n - a_len):
            c_len = self.n - a_len - b_len
            if c_len < 1:
                continue

            for start in range(self.n):
                a = self.get_sequence(start, a_len)
                b = self.get_sequence((start + a_len) %
self.n, b_len)
                c = self.get_sequence((start + a_len +
b_len) % self.n, c_len)

                if len(a) > 1 and a[0] == '0':
                    continue

                try:
                    a_num = int(a)
                    b_num = int(b)
                    c_num = int(c)
                except:
                    continue

                if a_num + b_num == c_num:
                    return f"{a_num}+{b_num}={c_num}"

    return "No"

def is_valid_filename(filename):
    """Проверяет, что имя файла корректное и файл существует"""
    if not filename:
        print("Имя файла не может быть пустым.")
        return False

    # Проверка на запрещённые символы (зависит от ОС, здесь
    пример для Windows)
    if re.search(r'[\<>:"/\\|?*]', filename):
        print("Имя файла содержит запрещённые символы.")
        return False

```

```

    if not os.path.isfile(filename):
        print("Файл не найден.")
        return False

    return True

def main():
    filename = input("Введите имя файла с цифрами кольца:").strip()

    if not is_valid_filename(filename):
        return

    with open(filename, 'r') as f:
        user_input = f.readline().strip()

    with open('output.txt', 'w') as f:

        if not user_input.isdigit():
            print("Строка должна содержать только цифры.")
            return

        if len(user_input) > 1000 or len(user_input) < 3:
            f.write("В строке должно быть не меньше 3 и не
больше 1000 цифр!\n")
            print("Данные записаны в output.txt")
            return

        # Особый случай для "000..."
        if all(d == '0' for d in user_input):
            f.write("0+0=0\n")
            print("Данные записаны в output.txt")
            return

        ring = NumberRing(user_input)
        result = ring.find_equation()
        f.write(result + '\n')

    print("Данные записаны в output.txt")

if __name__ == "__main__":
    main()

```