|  |  |  |
| --- | --- | --- |
| 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](#_Toc153130027)

[Алгоритм решения 3](#_Toc153130028)

[Тестирование 3](#_Toc153130029)

[Код программы 3](#_Toc153130030)

[Инструкция по применению стилей и оформлению работы 4](#_Toc153130031)

# 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 ≥ 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 ≤ 1000.

# Тестирование

Ввод:

  
 Вывод:



Ввод:

  
 Вывод:



Ввод:

  
 Вывод:



Ввод:

  
 Вывод:



Ввод:

  
 Вывод:



При вводе неизвестного имени файла выводит:



# 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=0n")  
 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()