Северо-Кавказский федеральный университет

Институт математики и информационных технологий

ОТЧЕТ

о выполнении лабораторной работы №4

по дисциплине

«Основы Программной Инженерии»

Выполнил:

Маняхин Тимур Александрович

студент 2 курса, ПИЖ-б-о-20-1 группы
бакалавриата «Программная инженерия»

очной формы обучения

Ставрополь, 2021

UML-диаграммы

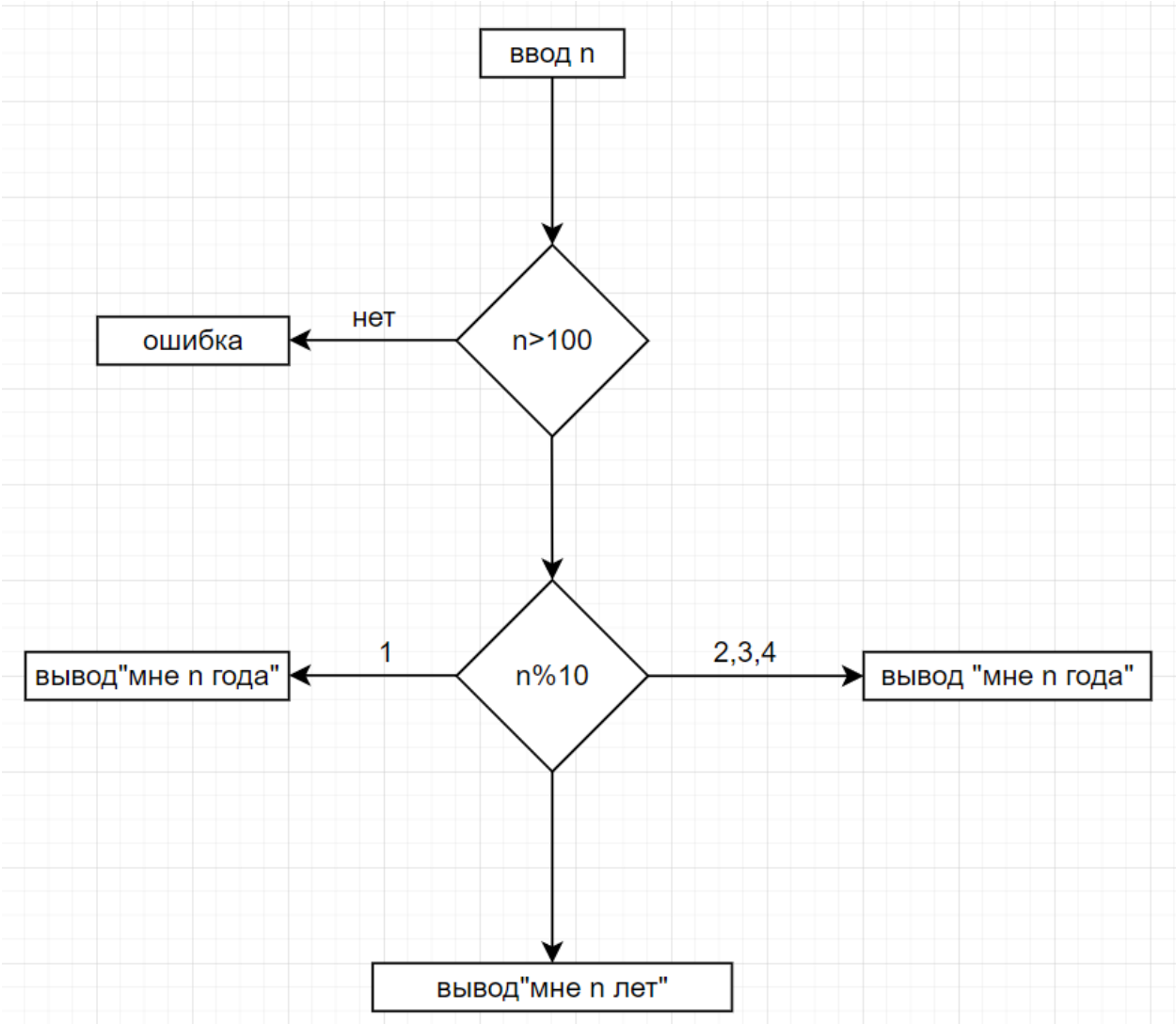

Рисунок 1.1 – диаграмма первого задания

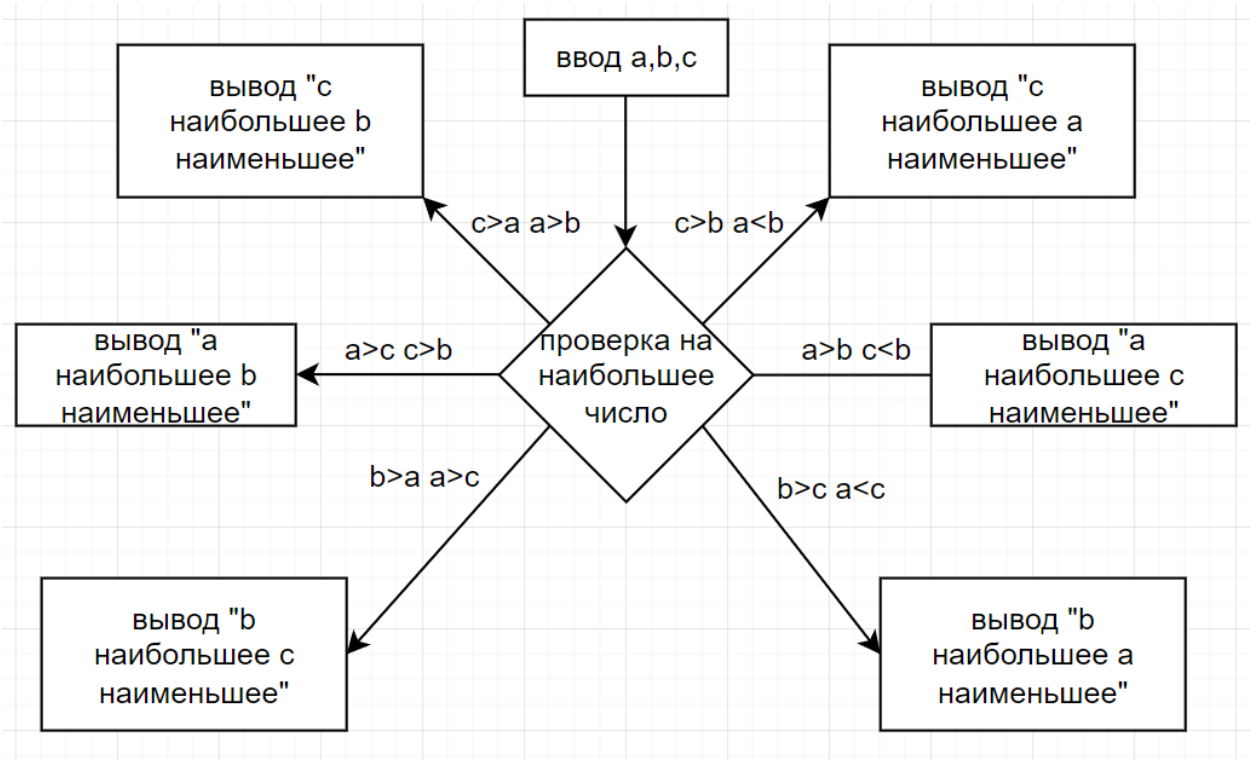

Рисунок 1.2 – диаграмма второго задания

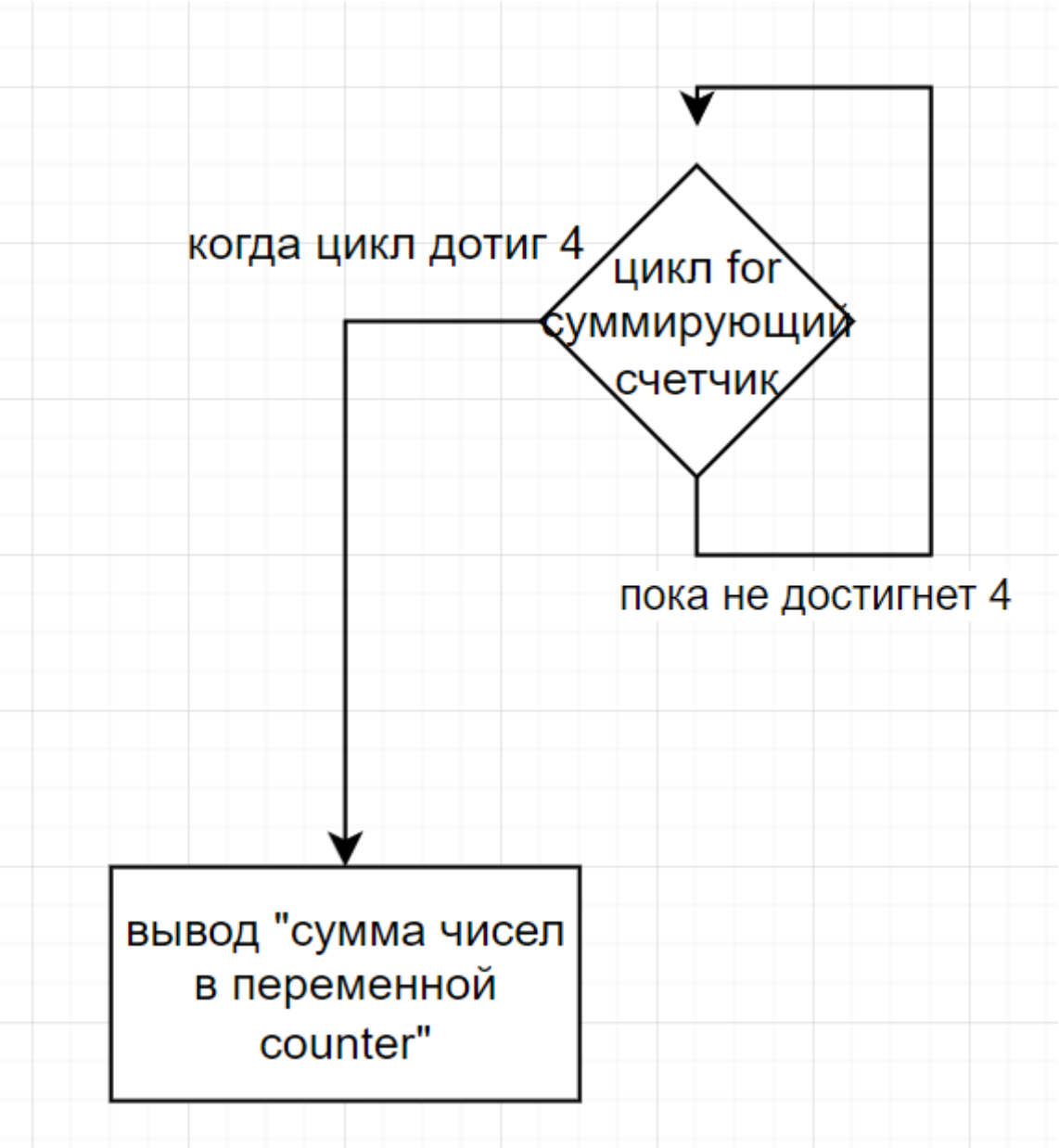

Рисунок 1.3 – диаграмма третьего задания

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# Дано натуральное число n > 100 . Вывести на экран фразу Мне n лет , учитывая, что при
# некоторых значениях n слово «лет» надо заменить на слово «год» или «года».
import sys
n = int(input("n = "))
if n < 100:
    print("Ошибка!", file=sys.stderr)
    exit(1)
elif n % 10 == 1:
    print("мне", n, "год")
elif n % 10 in (2, 3, 4):
    print("мне", n, "года")
else:
    print("мне", n, "лет")
```

Рисунок 2.1 – программа 1

```
n = 1054
мне 1054 года
```

Рисунок 2.2 – результат работы программы 1

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# Составить программу нахождения из трех чисел наибольшего и наименьшего
import math
a = int(input("первое число "))
b = int(input("второе число "))
c = int(input("третье число "))
if (a > b) & (a > c) & (b > c):
    print(a, "наибольшее", c, "наименьшее")
elif (a > b) & (a > c) & (c > b):
    print(a, "наибольшее", b, "наименьшее")
elif (b > a) & (b > c) & (a > c):
    print(b, "наибольшее", c, "наименьшее")
elif (b > a) & (b > c) & (a < c):
    print(b, "наибольшее", a, "наименьшее")
elif (c > a) & (c > b) & (a > b):
    print(c, "наибольшее", b, "наименьшее")
elif (c > a) & (c > b) & (a < b):
    print(c, "наибольшее", a, "наименьшее")
```

Рисунок 2.3 – программа 2

```
первое число 2
второе число 3
третье число 1
3 наибольшее 1 наименьшее
```

Рисунок 2.4 – результат работы программы 2

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
# Вычислить сумму всех n-значных чисел (1 <= n <= 4).
import math
n = [1, 2, 3, 4]
i = 0
counter = 0
for i in range(i, 4):
    counter += n[i]
i += 1
print("сумма всех чисел равна:", counter)
```

Рисунок 2.5 – программа 3

```
сумма всех чисел равна: 10
```

Рисунок 2.6 – результат работы программы 3

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. С помощью UML можно визуализировать, специфицировать, конструировать и документировать артефакты программных систем.

2. Состояние действия – это все атомарные вычисления системы. Состояние деятельности – это составное состояние, поток управления которого включает только другие состояния деятельности и действий.

3. Переход представляется простой линией со стрелкой Точка ветвления представляется ромбом

4. Алгоритм разветвляющейся структуры - это алгоритм, в котором вычислительный процесс осуществляется либо по одной, либо по другой ветви, в зависимости от выполнения некоторого условия.

5. Линейный алгоритм выполняется по одному пути

6. Условный оператор – это оператор который ставит некоторое условие. If, elif

7. > (больше), < (меньше), >= (больше или равно), <= (меньше или равно), == (равно), != (не равно)

8. Простое условие – это единичное условие. (y < 4)

9. Составное условие содержит несколько простых. (y < 4 or d > 10)

10.И (and), Или (or), не (not)

11.Оператор ветвления может содержать другие ветвления

12.Алгоритм циклической структуры - это алгоритм, в котором происходит многократное повторение одного и того же участка программы

13.Циклы: for, while

14.Функция range возвращает последовательность чисел в виде объекта range. Может использоваться в создании числовых последовательностей

15.Range(15, 0, -2)

16.Циклы могут быть вложенными

17.Бесконечный цикл можно сделать на основе неизменяемой переменной. Чтобы выйти из цикла надо изменить переменную

18.Оператор break предназначен для досрочного прерывания работы цикла

19.Оператор continue(используется в циклах) запускает цикл заново, при этом код, расположенный после данного оператора, не выполняется.

20.буферизованный поток stdout предназначен для вывода данных и информационных сообщений, а небуферизованный поток stderr нужен для вывода сообщений об ошибках

21.Для того, чтобы использовать поток stderr необходимо передать его в параметре file функции print

22.завершить программу и передать операционной системе заданный код возврата можно посредством функции exit