МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯРОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №2.2 по дисциплине основы программной инженерии

Выполнила:
Лобанова Елизавета Евгеньевна,
2 курс, группа ПИЖ-б-о-20-1

Проверил:
Доцент кафедры прикладной математики и компьютерной безопасности,
Воронкин Р.А.

Отчет защищен с оценкой_____Дата защиты_____

Ставрополь, 2021 г.

Ход работы

```python
import math

if __name__ == '__main__':
    x = float(input("Value of x? "))
    if x <= 0:
        y = 2 * x * x + math.cos(x)
    elif x < 5:
        y = x + 1
    else:
        y = math.sin(x) - x * x
    print(f"y = {y}")
```

Пример 1

```
Value of x? 5
y = -25.95892427466314
```

Результат 1 программы (1)

```
Value of x? 54
y = -2916.5587890488514
```

Результат 1 программы (2)

```
Value of x? 1
y = 2.0
```

Результат 1 программы (3)

```
Value of x? -3
y = 17.010007503399553
```

Результат 1 программы (4)

```python
import sys

if __name__ == '__main__':
    n = int(input("Введите номер месяца: "))
    if n == 1 or n == 2 or n == 12:
        print("Зима")
    elif n == 3 or n == 4 or n == 5:
        print("Весна")
    elif n == 6 or n == 7 or n == 8:
        print("Лето")
    elif n == 9 or n == 10 or n == 11:
        print("Осень")
    else:
        print("Ошибка!", file=sys.stderr)
        exit(1)
```

Пример 2

```
Введите номер месяца: 6
Лето
```

Результат 2 программы (1)

```
Введите номер месяца: 55
Ошибка!
```

Результат 2 программы (2)

```
Введите номер месяца: 1
Зима
```

Результат 2 программы (3)

```
Введите номер месяца: 11
Осень
```

Результат 2 программы (4)

```
Введите номер месяца: 3
Весна
```

Результат 2 программы (5)

```python
import math

if __name__ == '__main__':
    n = int(input("Value of n? "))
    x = float(input("Value of x? "))
    S = 0.0
    for k in range(1, n + 1):
        a = math.log(k * x) / (k * k)
        S += a

    print(f"S = {S}")
```

Пример 3

```
Value of n? 3
Value of x? 5
S = 2.485978652471746
```

Результат 3 программы (1)

```
Value of n? 40
Value of x? 22
S = 5.8297136137186225
```

Результат 3 программы (2)

```
Value of n? 33
Value of x? 7
S = 3.945693266825918
```

Результат 3 программы (3)

```
Value of n? 5432
Value of x? 123
S = 8.850621251786736
```

Результат 3 программы (4)

```python
import math
import sys

if __name__ == '__main__':
    a = float(input("Value of a? "))
    if a < 0:
        print("Illegal value of a", file=sys.stderr)
        exit(1)
    x, eps = 1, 1e-10
    while True:
        xp = x
        x = (x + a / x) / 2
        if math.fabs(x - xp) < eps:
            break

    print(f"x = {x}\nX = {math.sqrt(a)}")
```

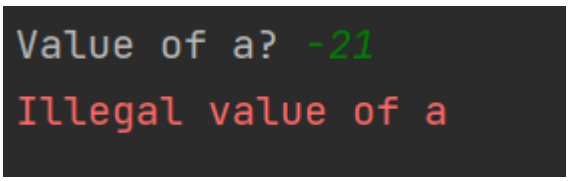
Пример 4

```
Value of a? 4
x = 2.0
X = 2.0
```

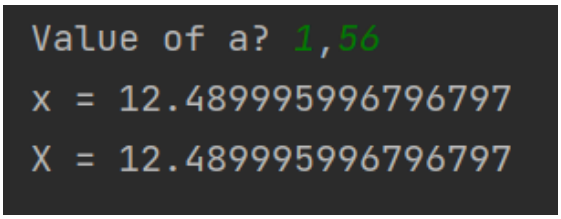
Результат 4 программы (1)

```
Value of a? 58
x = 7.615773105863909
X = 7.615773105863909
```

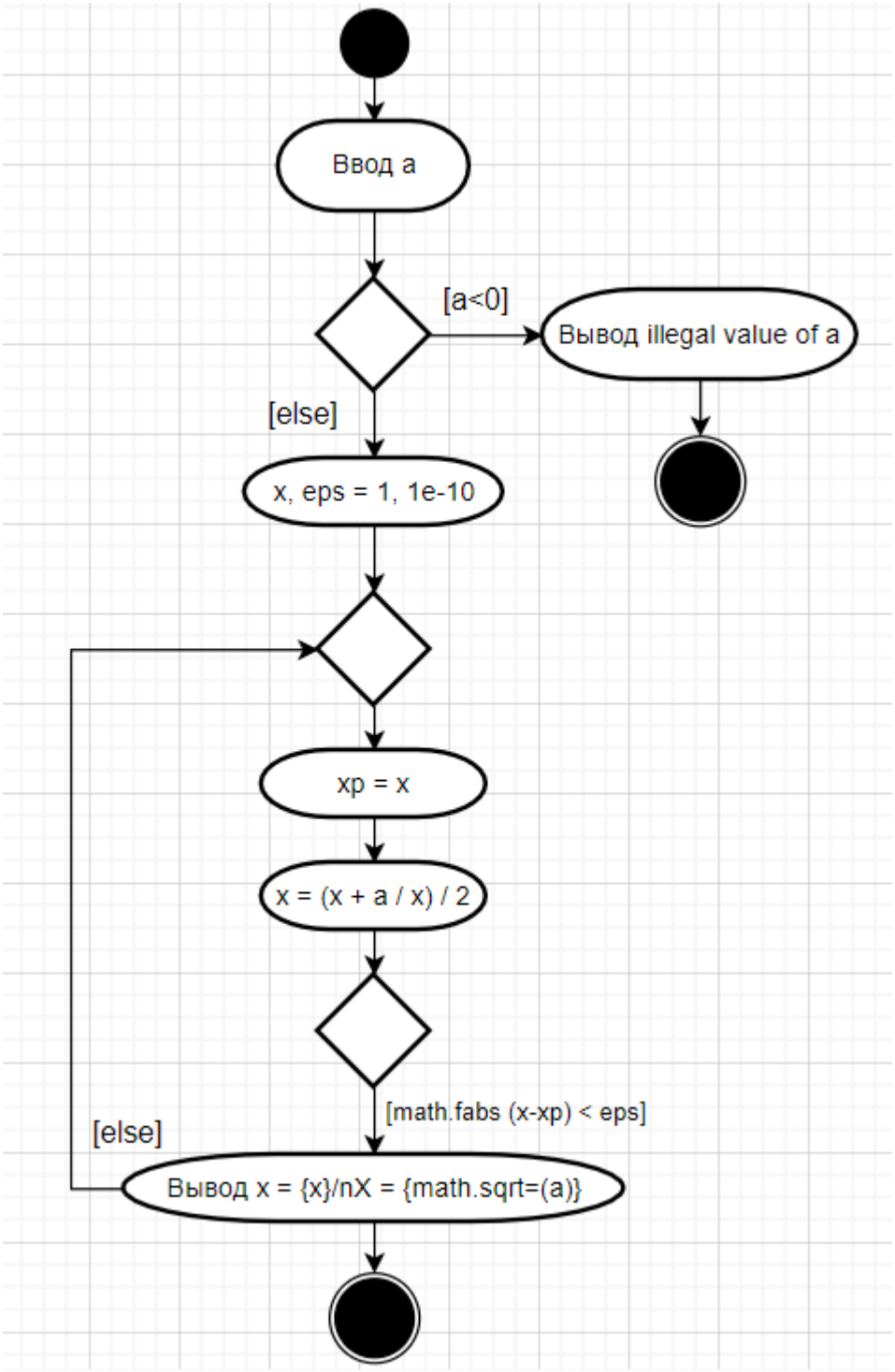
Результат 4 программы (2)



Результат 4 программы (3)



Результат 4 программы (4)



UML – Диаграмма 4 программы

```python
import math
import sys
# Постоянная Эйлера.
EULER = 0.5772156649015328606
# Точность вычислений.
EPS = 1e-10

if __name__ == '__main__':
    x = float(input("Value of x? "))
    if x == 0:
        print("Illegal value of x", file=sys.stderr)
        exit(1)
    a = x
    S, k = a, 1
    # Найти сумму членов ряда.
    while math.fabs(a) > EPS:
        a *= x * k / (k + 1) ** 2
        S += a
        k += 1
    # Вывести значение функции.
    print(f"Ei({x}) = {EULER + math.log(math.fabs(x)) + S}")
```

Пример 5

```
Value of x? 7
Ei(7.0) = 191.50474333549477
```

Результат 5 программы (1)

```
Value of x? 688
Ei(688.0) = 9.0708447773895 77e+295
```

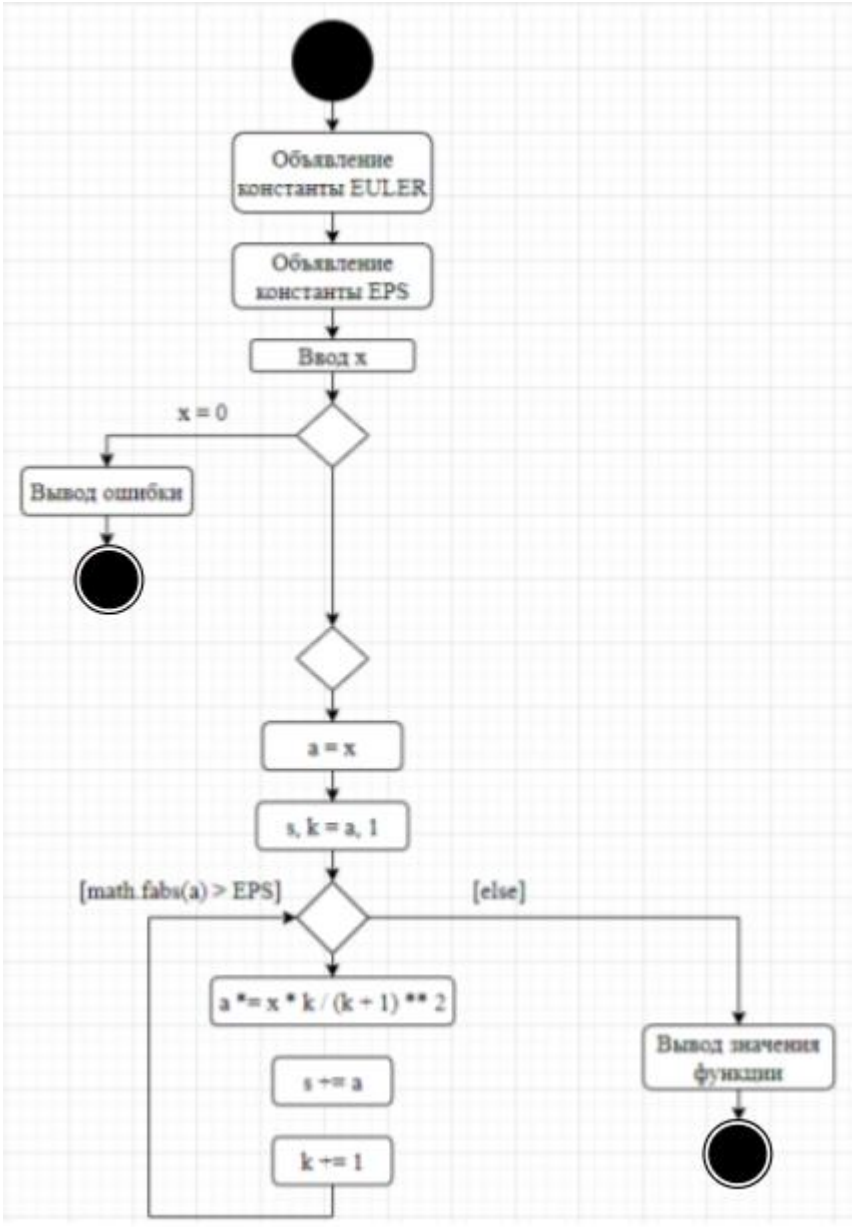
Результат 5 программы (2)

```
Value of x? -4
Ei(-4.0) = -0.00377935240732 10794
```

Результат 5 программы (3)

```
Value of x? 65
Ei(65.0) = 2.6489327386345484e+26
```

Результат 5 программы (4)



UML – Диаграмма 5 программы

```python
x1 = int(input("Price of one roll of wallpaper = "))
x2 = int(input("The price of one can of paint = "))

price = float(8 * x1 + 2 * x2)

if 200 <= price <= 500:
    price *= 0.97
elif 500 < price <= 800:
    price *= 0.95
elif 800 < price <= 1000:
    price *= 0.93
elif price > 1000:
    price *= 0.91

print("Discount price = %.2f" % price)
```

Индивидуальное задание 1

```
Price of one roll of wallpaper = 340
The price of one can of paint = 290
Discount price = 3003.00
```

Результат 1 индивидуальной программы (1)

```
Price of one roll of wallpaper = 444
The price of one can of paint = 501
Discount price = 4144.14
```

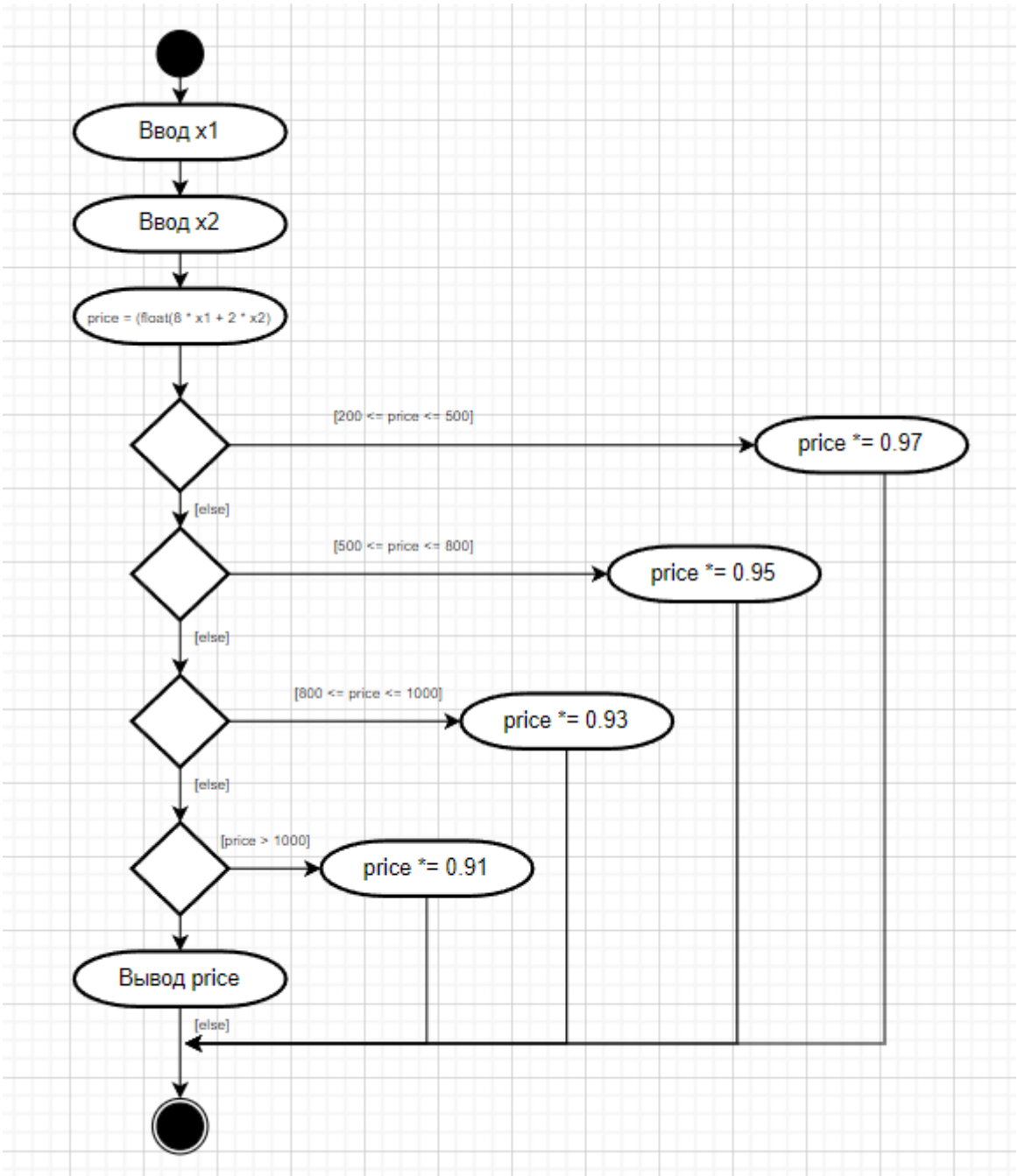
Результат 1 индивидуальной программы (2)

```
Price of one roll of wallpaper = 150
The price of one can of paint = 150
Discount price = 1365.00
```

Результат 1 индивидуальной программы (3)

```
Price of one roll of wallpaper = 75
The price of one can of paint = 50
Discount price = 665.00
```

Результат 1 индивидуальной программы (4)



UML – Диаграмма 1 индивидуальной программы

```python
import math

if __name__ == '__main__':
    x1 = float(input("Enter x1 = "))
    x2 = float(input("Enter x2 = "))
    y1 = float(input("Enter y1 = "))
    y2 = float(input("Enter y2 = "))
    r1 = float(input("Enter r1 = "))
    r2 = float(input("Enter r2 = "))

    d = float(math.sqrt(((x1-x2) ** 2) + ((y1-y2) ** 2)))

    if (d == r1 + r2) or (d + r2 == r1) or (d + r1 == r2):
        print("1 point of intersection; d = %.2f" % d)
    elif (d >= r1 + r2) or (d + r2 < r1) or (d + r1 < r2):
        print("No intersection points; d = %.2f" % d)
    elif (x1 == x2) and (y1 == y2) and (r1 == r2):
        print("The circles match; d = %.2f" % d)
    else:
        print("2 points of intersection; d = %.2f" % d)
```

Индивидуальное задание 2

```
Enter x1 = 3
Enter x2 = 5
Enter y1 = 8
Enter y2 = 7
Enter r1 = 5
Enter r2 = 1
No intersection points; d = 2.24
```

Результат 2 индивидуальной программы (1)

```
Enter x1 = 5
Enter x2 = 65
Enter y1 = 78
Enter y2 = 33
Enter r1 = 2
Enter r2 = 12
No intersection points; d = 75.00
```

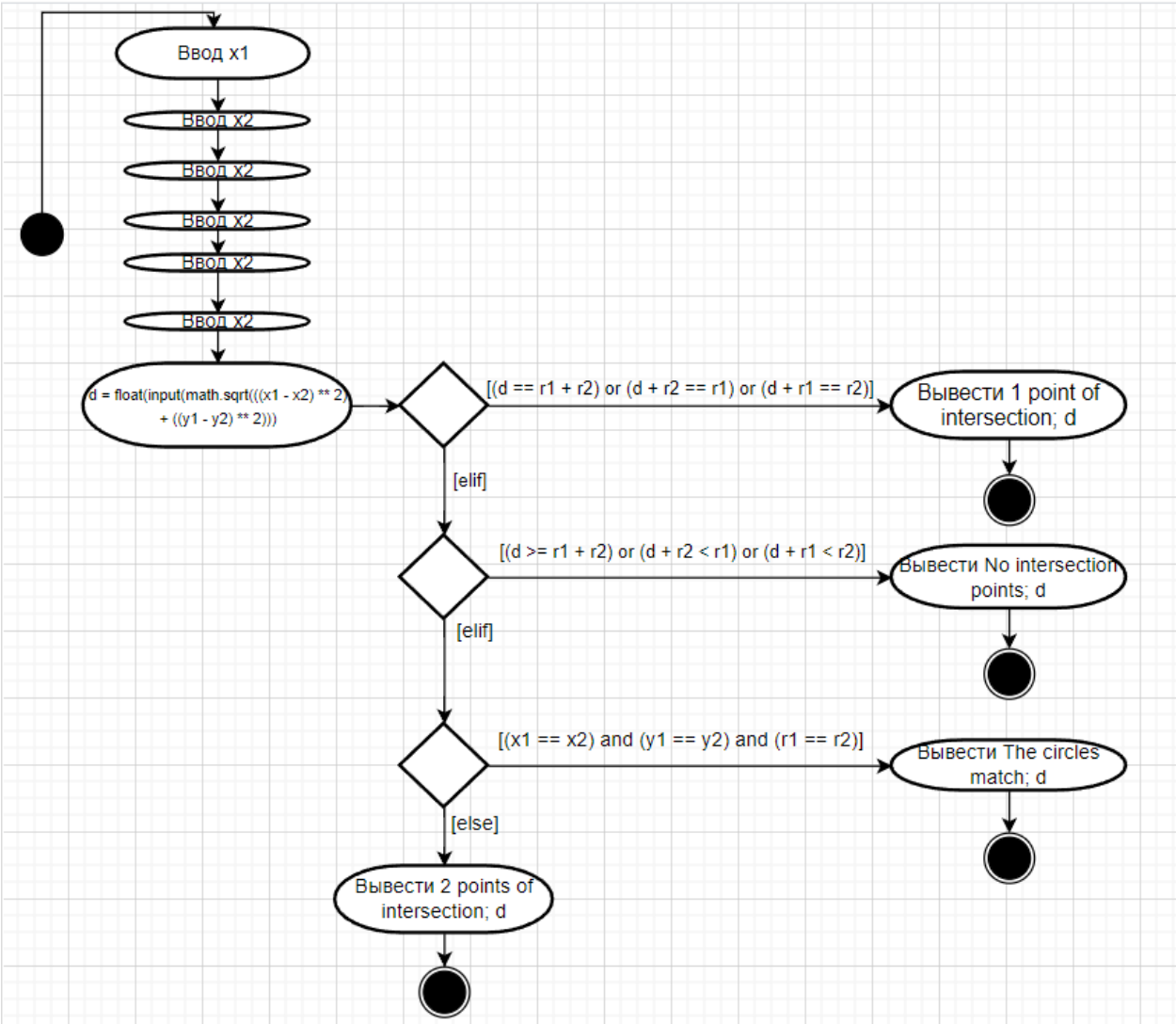
Результат 2 индивидуальной программы (2)

```
Enter x1 = 3
Enter x2 = 2
Enter y1 = 5
Enter y2 = 1
Enter r1 = 4
Enter r2 = 2
2 points of intersection; d = 4.12
```

Результат 2 индивидуальной программы (3)

```
Enter x1 = 5
Enter x2 = 55
Enter y1 = 5
Enter y2 = 67
Enter r1 = 22
Enter r2 = 21
No intersection points; d = 79.65
```

Результат 2 индивидуальной программы (4)



UML – Диаграмма 2 индивидуальной программы

```
k1 = 0
k2 = 0
k5 = 0
k10 = 0
k50 = 0
k100 = 0
k500 = 0

s = int(input("Введите сумму: "))

while s > 0:
    while s >= 500:
        s -= 500
        k500 += 1
    else:
        print("Покупатель отдаст: ", k500, "купюр по 500р")
    while s >= 100:
        s -= 100
        k100 += 1
    else:
        print("Покупатель отдаст: ", k100, "купюр по 100р, ")
```

Индивидуальное задание 3

```
    while s >= 50:
        s -= 50
        k50 += 1
    else:
        print("Покупатель отдаст: ", k50, "купюр по 50р, ")
    while s >= 10:
        s -= 10
        k10 += 1
    else:
        print("Покупатель отдаст: ", k10, "купюр по 10р, ")
    while s >= 5:
        s -= 5
        k5 += 1
```

Индивидуальное задание 3

```
while s >= 2:
    s -= 2
    k2 += 1
else:
    print("Покупатель отдаст: ", k2, "купюр по 2р, ")
while s >= 1:
    s -= 1
    k1 += 1
else:
    print("Покупатель отдаст: ", k1, "купюр по 1р.")
```

Индивидуальное задание 3

```
Введите сумму: 4538
Покупатель отдаст:  9 купюр(у) по 500р.
Покупатель отдаст:  0 купюр(у) по 100р.
Покупатель отдаст:  0 купюр(у) по 50р.
Покупатель отдаст:  3 купюр(у) по 10р.
Покупатель отдаст:  1 купюр(у) по 5р.
Покупатель отдаст:  1 купюр(у) по 2р.
Покупатель отдаст:  1 купюр(у) по 1р.
```

Результат 3 индивидуальной программы (1)

```
Введите сумму: 5654
Покупатель отдаст:  11 купюр(у) по 500р.
Покупатель отдаст:  1 купюр(у) по 100р.
Покупатель отдаст:  1 купюр(у) по 50р.
Покупатель отдаст:  0 купюр(у) по 10р.
Покупатель отдаст:  0 купюр(у) по 5р.
Покупатель отдаст:  2 купюр(у) по 2р.
Покупатель отдаст:  0 купюр(у) по 1р.
```

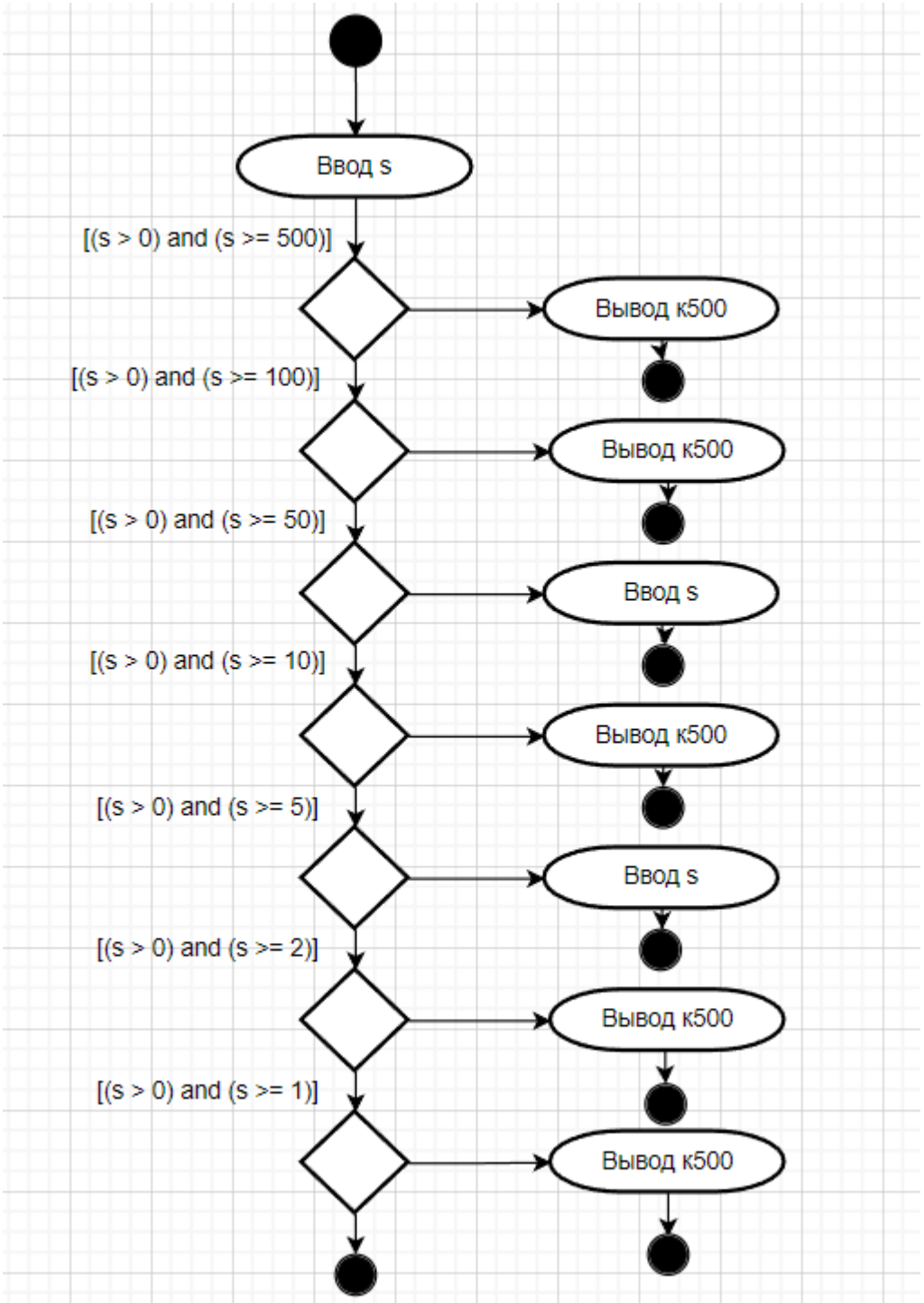
Результат 3 индивидуальной программы (2)

```
Введите сумму: 632
Покупатель отдаст:  1 купюр(у) по 500р.
Покупатель отдаст:  1 купюр(у) по 100р.
Покупатель отдаст:  0 купюр(у) по 50р.
Покупатель отдаст:  3 купюр(у) по 10р.
Покупатель отдаст:  0 купюр(у) по 5р.
Покупатель отдаст:  1 купюр(у) по 2р.
Покупатель отдаст:  0 купюр(у) по 1р.
```

Результат 3 индивидуальной программы (3)

```
Введите сумму: 2098760
Покупатель отдаст:  4197 купюр(у) по 500р.
Покупатель отдаст:  2 купюр(у) по 100р.
Покупатель отдаст:  1 купюр(у) по 50р.
Покупатель отдаст:  1 купюр(у) по 10р.
Покупатель отдаст:  0 купюр(у) по 5р.
Покупатель отдаст:  0 купюр(у) по 2р.
Покупатель отдаст:  0 купюр(у) по 1р.
```

Результат 3 индивидуальной программы (4)



UML – Диаграмма 2 индивидуальной программы

Ответы на вопросы:

1. Диаграммы деятельности - это один из пяти видов диаграмм, применяемых в UML для моделирования динамических аспектов поведения системы. Диаграмма деятельности - это, по существу, блок-схема, которая показывает, как поток управления переходит от одной деятельности к другой, однако, по сравнению с последней, у ней есть явные преимущества: поддержка многопоточности и объектно-ориентированного проектирования. Диаграмма деятельности (Activity diagram) показывает поток переходов от одной деятельности к другой.

2. Вы можете вычислить выражение, в результате чего изменяется значение некоторого атрибута или возвращается некоторое значение. Также, например, можно выполнить операцию над объектом, послать ему сигнал или даже создать его или уничтожить. Все эти выполняемые атомарные вычисления называются состояниями действия, поскольку каждое из них есть состояние системы, представляющее собой выполнение некоторого действия. Состояния действия не могут быть подвергнуты декомпозиции. Кроме того, они атомарны. Это значит, что внутри них могут происходить различные события, но выполняемая в состоянии действия работа не может быть прервана. Обычно предполагается, что длительность одного состояния действия занимает неощутимо малое время. В противоположность этому состояния деятельности могут быть подвергнуты дальнейшей декомпозиции, вследствие чего выполняемую деятельность можно представить с помощью других диаграмм деятельности. Состояния деятельности не являются атомарными, то есть могут быть прерваны. Предполагается, что для их завершения требуется заметное время. Можно считать, что состояние действия - это частный вид состояния деятельности, а конкретнее – такое состояние, которое не может быть подвергнуто дальнейшей декомпозиции. А состояние деятельности можно представлять себе как составное состояние, поток управления которого включает только другие состояния деятельности и действий.

3. В UML переход представляется простой линией со стрелкой. Точка ветвления представляется ромбом. В точку ветвления может входить ровно один переход, а выходить – два или более.

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 последовательно независимо от чего-либо, а алгоритм ветвления выполняется определенные действия в зависимости от выполнения условия или условий.

6. Оператор ветвления «if» позволяет выполнить определенный набор инструкций в зависимости от некоторого условия. Возможны следующие варианты использования: 1) Конструкция «if» 2) Конструкция «if» - «else» 3) Конструкция «if» - «elif» - «else»

7. , <=, >=, ==, !=.

8. Логические выражения являются простыми, если в них выполняется только одна логическая операция. «x > 15» или «a != b»

9. В составных условиях используется 2 и более логические операции. «x > 8 and y <= 3»

10. and и or

11. Да, может.

12. Алгоритм циклической структуры - это алгоритм, в котором происходит многократное повторение одного и того же участка программы. Такие повторяемые участки вычислительного процесса называются циклами.

13. Цикл «while» и цикл «for».

14. Функция range возвращает неизменяемую последовательность чисел в виде объекта range. Параметры функции: start - с какого числа начинается последовательность. По умолчанию - 0 stop - до какого числа продолжается последовательность чисел Указанное число не включается в диапазон. step - с каким шагом растут числа. По умолчанию 1 Функция range хранит только информацию о значениях start, stop и step и вычисляет значения по мере необходимости. Это значит, что независимо от размера диапазона, который описывает функция range, она всегда будет занимать фиксированный объем памяти.

15. range(15, 0, -2).

16. Да, могут.

17. Пример бесконечного цикла: a = 0 while a >= 0: if a == 7: break a += 1 print("A") Оператор «break» предназначен для досрочного прерывания работы цикла «while».

18. Оператор «break» предназначен для досрочного прерывания работы цикла «while».

19. Оператор «continue» запускает цикл заново, при этом код, расположенный после данного оператора, не выполняется.

20. В операционной системе по умолчанию присутствуют стандартные потоки вывода на консоль: буферизованный поток stdout для вывода данных и информационных сообщений, а также небуферизованный поток stderr для вывода сообщений об ошибках. По умолчанию функция print использует поток stdout. Хорошим стилем программирования является наличие вывода ошибок в стандартный поток stderr поскольку вывод в потоки stdout и stderr может обрабатываться как операционной системой, так и сценариями пользователя по-разному.

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

22. В Python завершить программу и передать операционной системе заданныq код возврата можно посредством функции exit. В данном примере выполняется вызов exit(1) , что приводит к немедленному завершению программы и операционной системе передается 1 в качестве кода возврата, что говорит о том, что в процессе выполнения программы произошли ошибки.