

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования «СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ  
УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №2.2

Дисциплина: «Основы кроссплатформенного программирования»

Тема: «Условные операторы и циклы в языке Python»

Выполнила: студентка 1 курса,  
группы ИВТ-б-о-21-1  
Диченко Дина Алексеевна

Ставрополь 2022

**Цель:** приобретение навыков программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Освоить операторы языка Python версии 3.x if , while , for , break и continue , позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.

### Выполнение работы:

1. Создала открытый репозиторий.

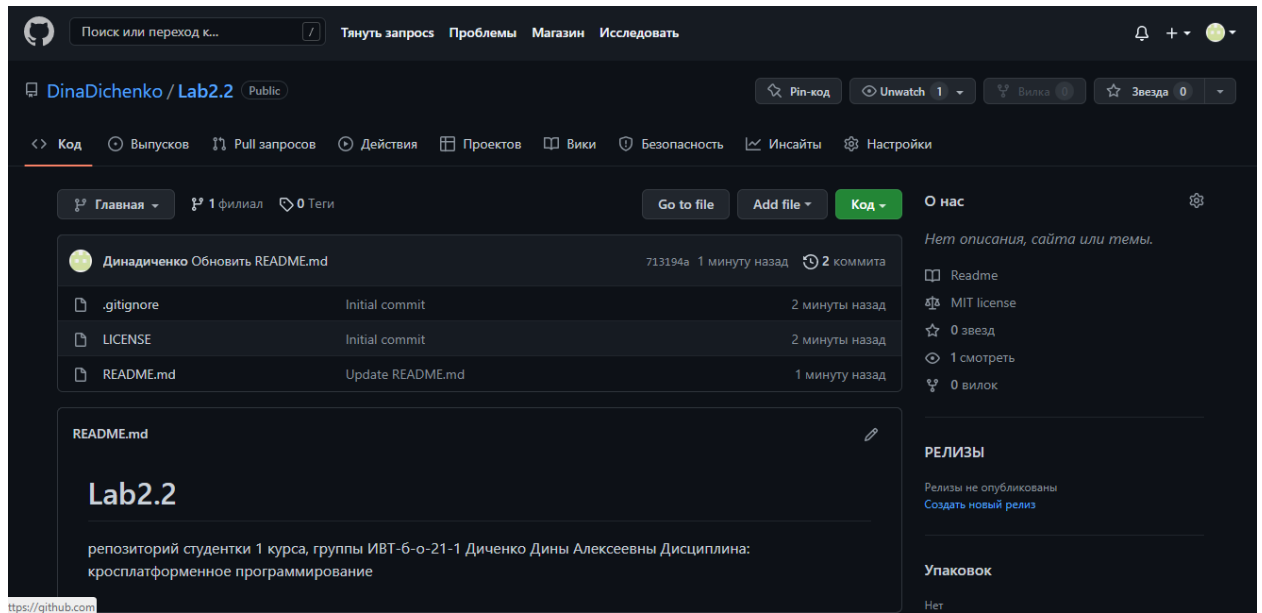


Рисунок 1. Создание репозитория

2. Клонировала созданный репозиторий.

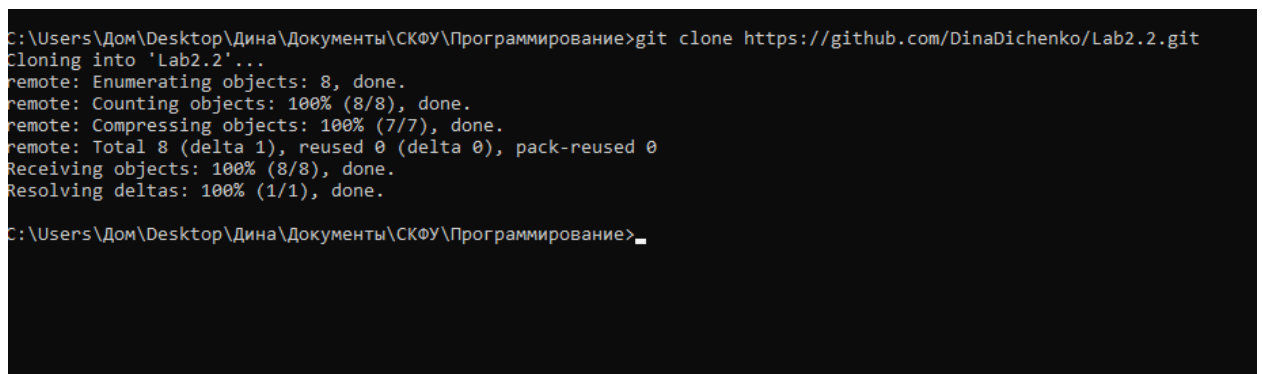


Рисунок 2. Клонирование репозитория

3. Дополнила файл .gitignore необходимыми правилами для работы с IDE PyCharm.

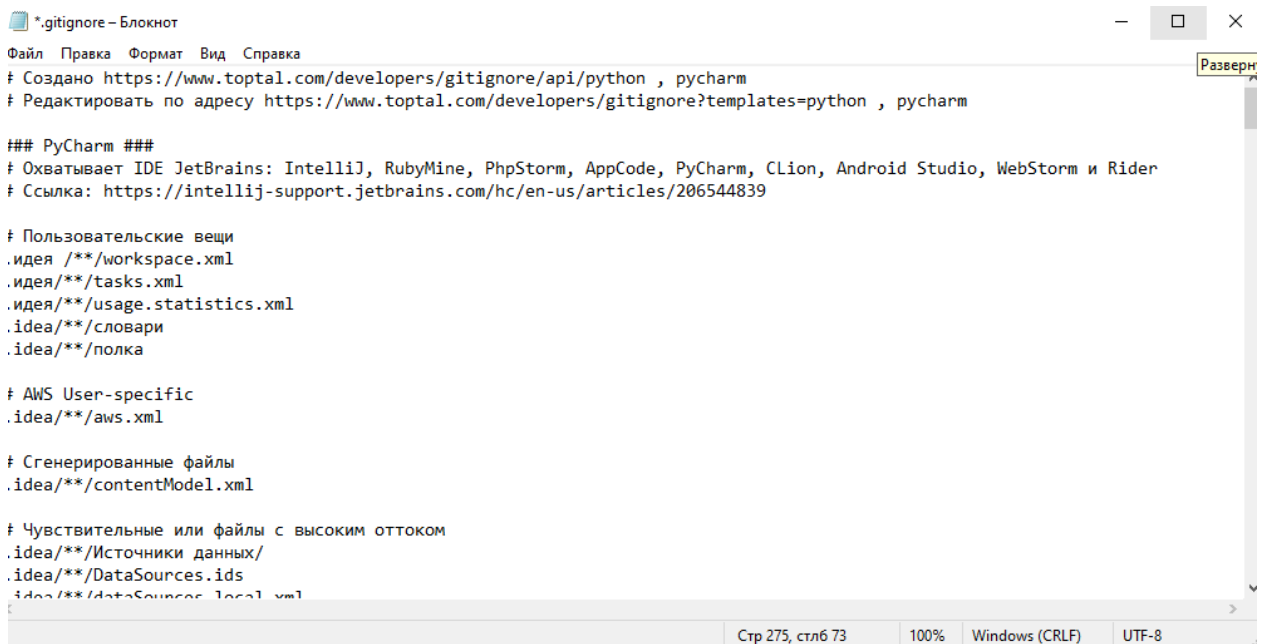


Рисунок 3. Редактирование файла .gitignore

4. Организовала свой репозиторий в соответствии с моделью ветвления git-flow.

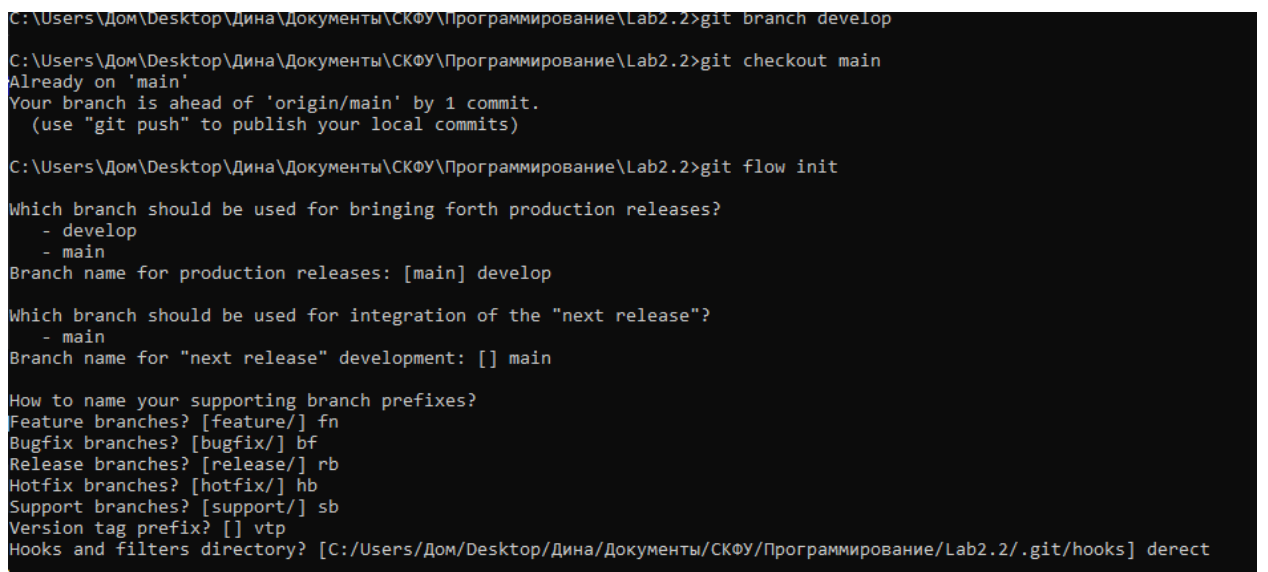


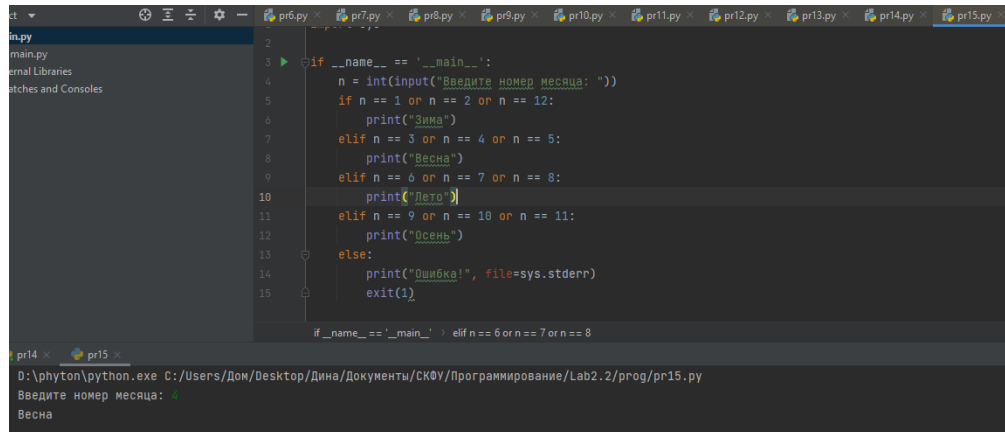
Рисунок 4. Организация репозитория в соответствии с моделью git-flow

5. Создала проект PyCharm в папке репозитория.

Имя	Дата изменения	Тип
.git	06.05.2022 12:24	Папка с фай
prog	06.05.2022 12:39	Папка с фай
.gitignore	06.05.2022 12:10	Текстовый д
LICENSE	06.05.2022 12:07	Файл
README	06.05.2022 12:07	Файл "MD"

Рисунок 5. Проект PyCharm в папке репозитория

## 6. Проработала примеры.



```
1 if __name__ == '__main__':
2     n = int(input("ВВЕДИТЕ НОМЕР МЕСЯЦА: "))
3     if n == 1 or n == 2 or n == 12:
4         print("Зима")
5     elif n == 3 or n == 4 or n == 5:
6         print("Весна")
7     elif n == 6 or n == 7 or n == 8:
8         print("Лето")
9     elif n == 9 or n == 10 or n == 11:
10        print("Осень")
11    else:
12        print("Ошибка!", file=sys.stderr)
13        exit(1)
14
15 if __name__ == '__main__': > elif n == 6 or n == 7 or n == 8
```

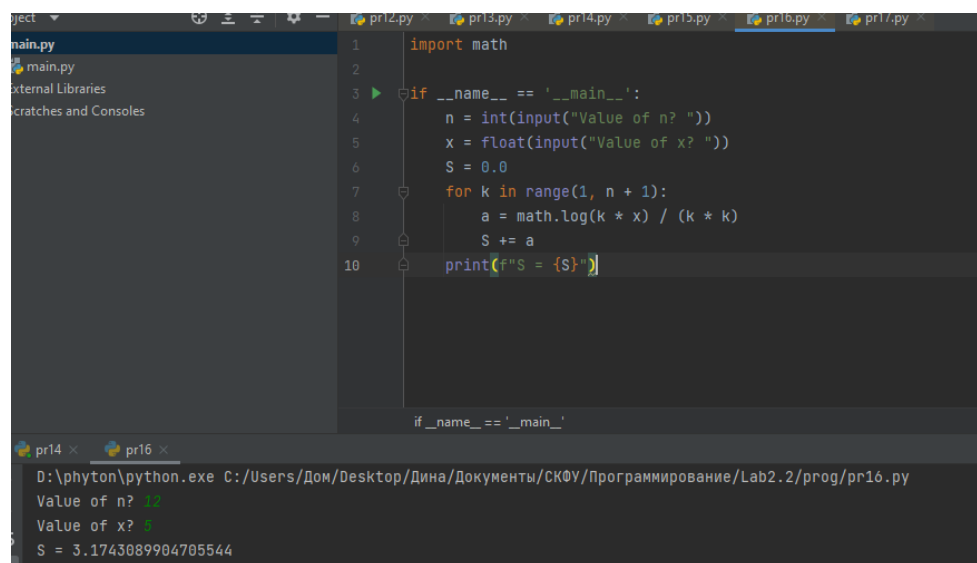
pr14 x pr15 x

D:\phyton\python.exe C:/Users/Дом/Desktop/Дина/Документы/СКОУ/Программирование/Lab2.2/prog/pr15.py

Введите номер месяца: 6

Весна

Рисунок 7. Проработка примеров



```
1 import math
2
3 if __name__ == '__main__':
4     n = int(input("Value of n? "))
5     x = float(input("Value of x? "))
6     S = 0.0
7     for k in range(1, n + 1):
8         a = math.log(k * x) / (k * k)
9         S += a
10    print(f"S = {S}")
```

pr14 x pr16 x

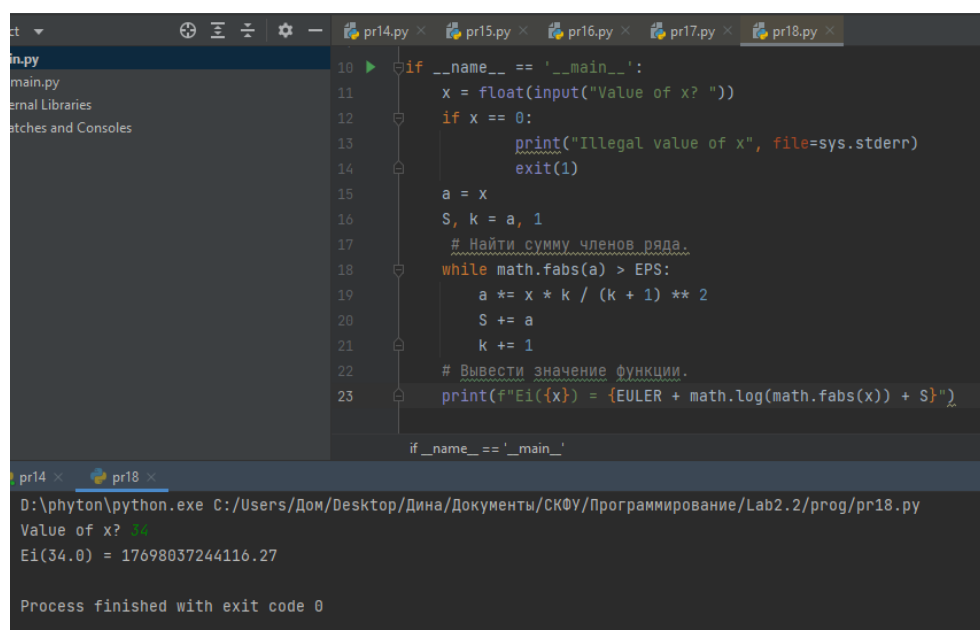
D:\phyton\python.exe C:/Users/Дом/Desktop/Дина/Документы/СКОУ/Программирование/Lab2.2/prog/pr16.py

Value of n? 10

Value of x? 5

S = 3.1743089904705544

Рисунок 8. Проработка примеров



```
10 if __name__ == '__main__':
11     x = float(input("Value of x? "))
12     if x == 0:
13         print("Illegal value of x", file=sys.stderr)
14         exit(1)
15     a = x
16     S, k = a, 1
17     # Найти сумму членов ряда.
18     while math.fabs(a) > EPS:
19         a *= x * k / (k + 1) ** 2
20         S += a
21         k += 1
22     # Вывести значение функции.
23     print(f"Ei({x}) = {EULER + math.log(math.fabs(x)) + S}")
```

pr14 x pr18 x

D:\phyton\python.exe C:/Users/Дом/Desktop/Дина/Документы/СКОУ/Программирование/Lab2.2/prog/pr18.py

Value of x? 34

Ei(34.0) = 17698037244116.27

Process finished with exit code 0

Рисунок 9. Проработка примеров

```

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab2.2>git commit -m "primer"
[main 30c68d8] primer
11 files changed, 109 insertions(+)
create mode 100644 prog/.idea/.gitignore
create mode 100644 prog/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 prog/.idea/misc.xml
create mode 100644 prog/.idea/modules.xml
create mode 100644 prog/.idea/program.iml
create mode 100644 prog/.idea/vcs.xml
create mode 100644 prog/pr1.py
create mode 100644 prog/pr2.py
create mode 100644 prog/pr3.py
create mode 100644 prog/pr4.py
create mode 100644 prog/pr5.py

C:\Users\Дом\Desktop\Дина\Документы\СКФУ\Программирование\Lab2.2>git push
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 2 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (16/16), 2.74 KiB | 700.00 KiB/s, done.
Total 16 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DinaDichenko/Lab2.2.git
   bfaaa40..30c68d8  main -> main

```

Рисунок 10. Сохранила изменения в репозитории

7. Построила UML-диаграмму деятельности для примеров 4 и 5.

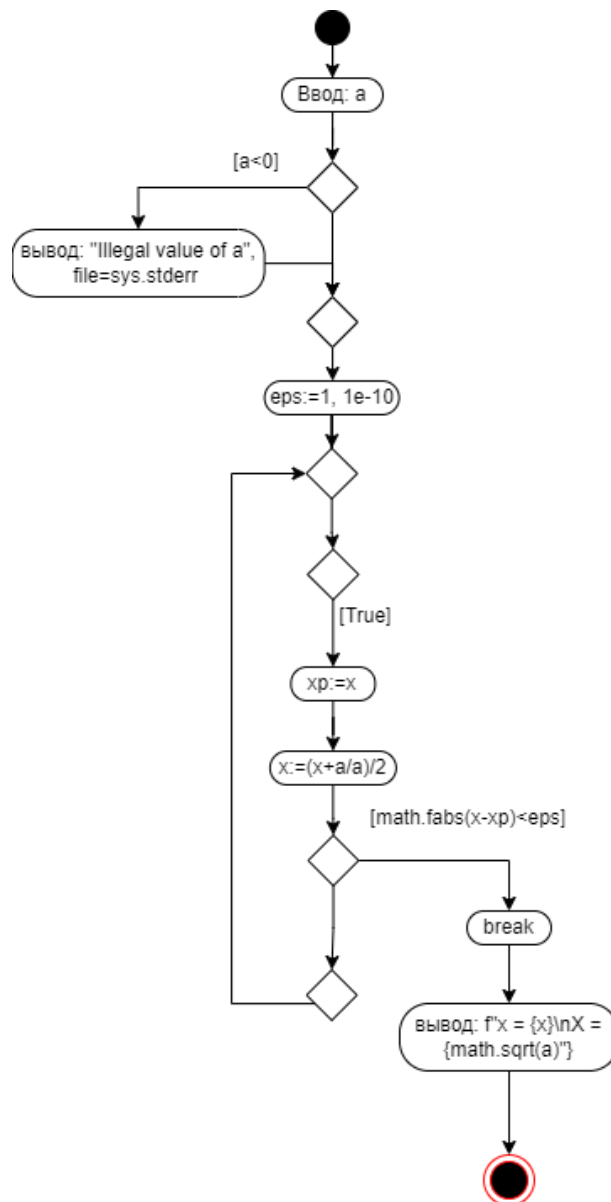


Рисунок 11. UML-диаграмма деятельности к примеру 4

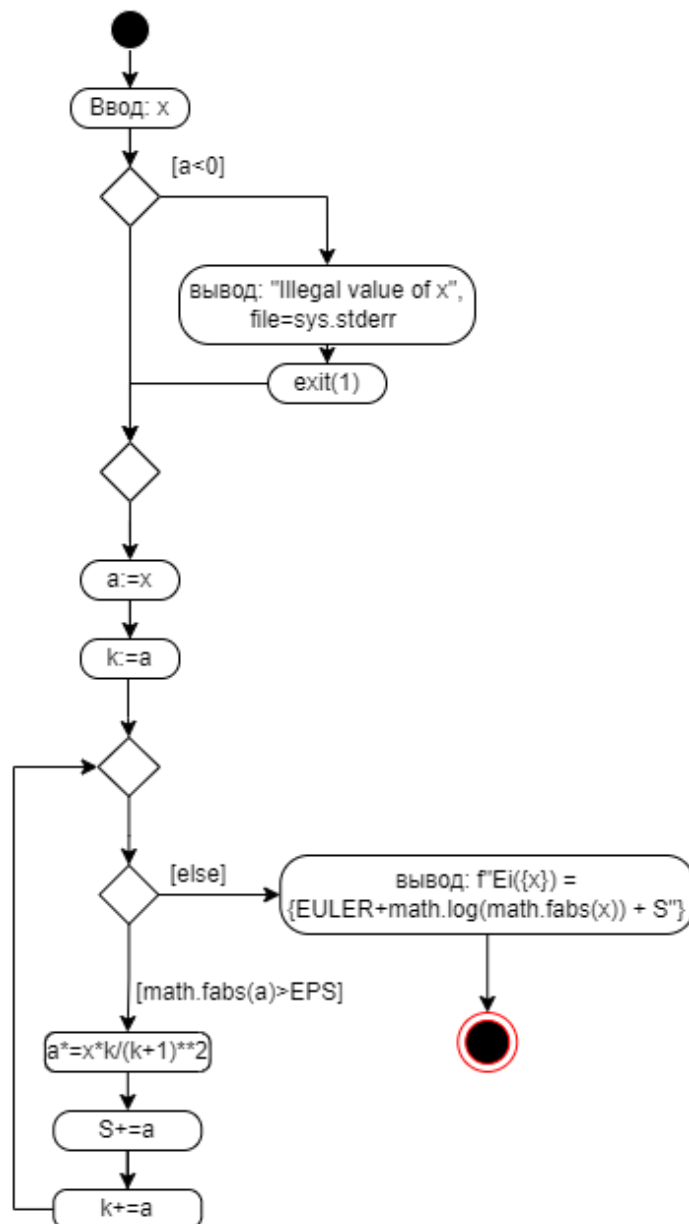


Рисунок 12. UML-диаграмма деятельности к примеру 5

8. Выполнила индивидуальные задания.

Вариант 7.

Задание 1. С клавиатуры вводится цифра (от 1 до 12). Вывести на экран название месяца, соответствующего цифре.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-#
3
4  import sys
5
6  if __name__ == '__main__':
7      a = int(input("Enter the month number: "))
8
9      if a == 1:
10         print("January")
11     elif a == 2:
12         print("February")
13     elif a == 3:
14         print("March")
15     elif a == 4:
16         print("April")
17     elif a == 5:
18         print("May")
19     elif a == 6:
20         print("June")
21     elif a == 7:
22         print("July")
23     elif a == 8:
24         print("August")
25     elif a == 9:
26         print("September")
27     elif a == 10:
28         print("October")
29     elif a == 11:
30         print("November")
31     elif a == 12:
32         print("December")
33     else:
34         print("Invalid month number")
35
36 if __name__ == '__main__': else

```

ind1 x  
D:\phyton\python.exe C:/Users/Дом/Desktop/Дина/Документы/СКОУ/Программирование/Lab2.2/prog/ind1.py  
Enter the month number: 5  
May

Рисунок 13. Результат работы 1 индивидуальной программы

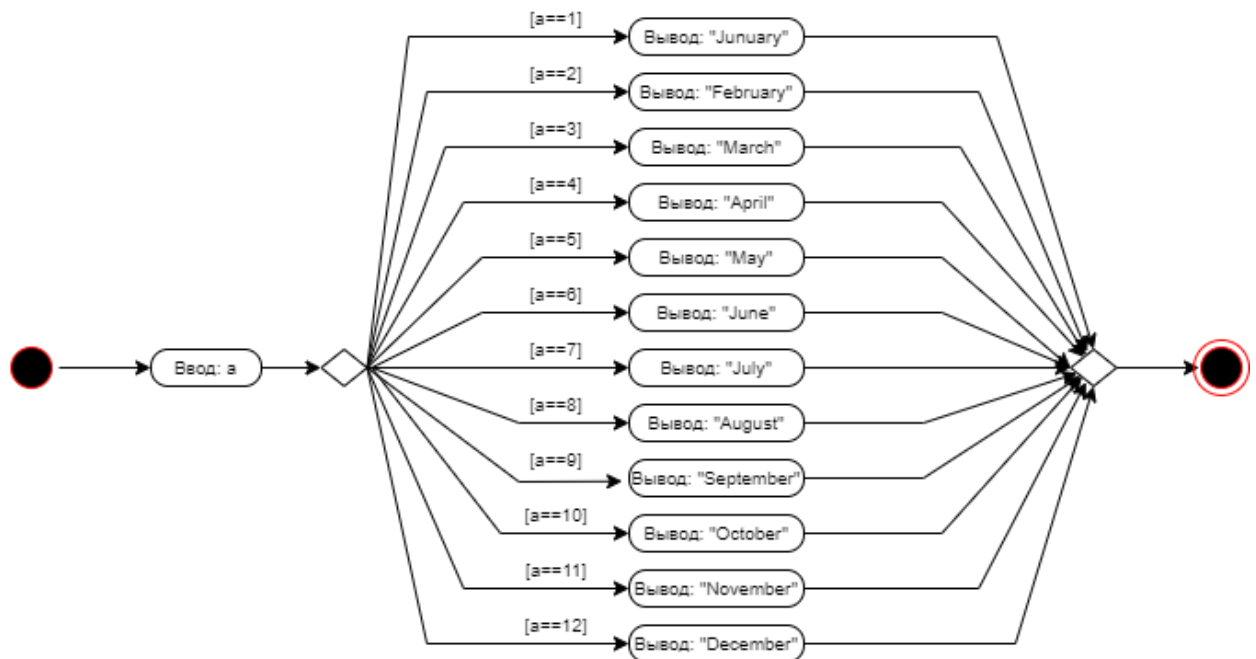


Рисунок 14. UML-диаграмма деятельности 1 индивидуального задания

Задание 2. Провести исследование биквадратного уравнения  $ax^4+bx^2+c=0$  ( $a \neq 0$ ), где,  $a$ ,  $b$  и  $c$  - действительные числа. Если действительных корней нет, то об этом должно быть выдано сообщение, иначе должны быть выданы 2 или 4 действительных корня.

```

1.py
ind1.py
External Libraries
atches and Consoles

3
4 import ..
5
6
7 if __name__ == '__main__':
8     a = float(input("a = "))
9     b = float(input("b = "))
10    c = float(input("c = "))
11    dis = pow(b, 2) - 4*a*c
12    if dis < 0:
13        print("there are no roots")
14    elif dis == 0:
15        x = sqrt((-b)/(2*a))
16        print("x1 = ", x, "x2 = ", -x)
17    elif dis > 0:
18        x1 = pow((-b-pow((dis), 0.5)) / (2 * a), 0.5)
19        x2 = pow((-b+pow((dis), 0.5)) / (2 * a), 0.5)
20        print("x1 = ", x1, "x2 = ", -x1, "x3 = ", x2, "x4 = ", -x2)
21    else:
22        print("Error")

ind2
D:\python\python.exe C:/Users/Дом/Desktop/Дина/Документы/СКОУ/Программирование/Lab2.2/prog/ind2.py
a = 1
b = -5
c = 4
x1 = 1.0 x2 = -1.0 x3 = 2.0 x4 = -2.0

```

Рисунок 15. Результат работы 2 индивидуальной программы

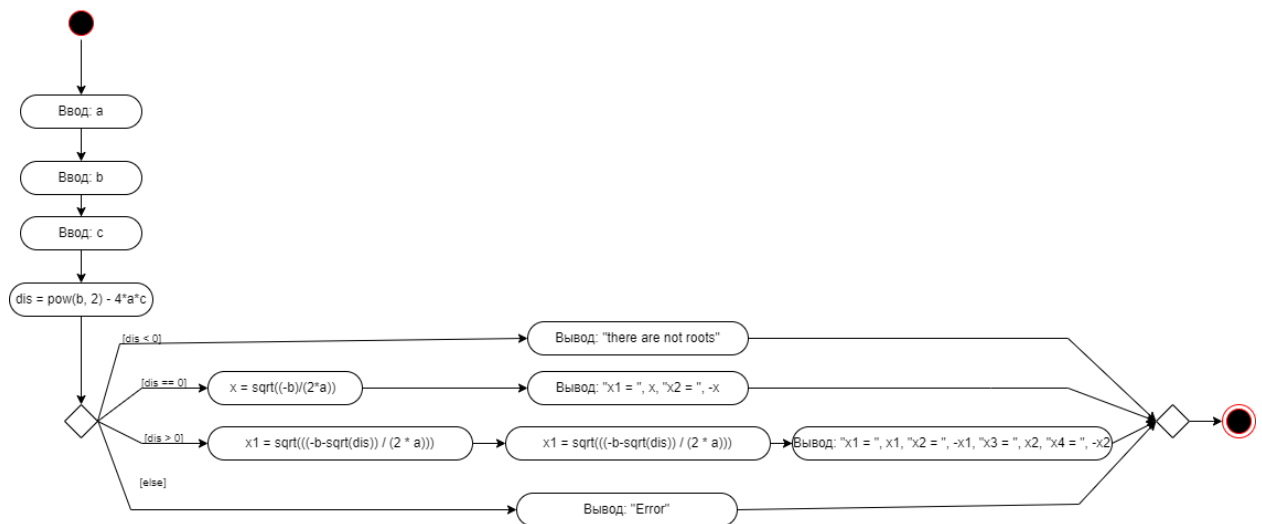


Рисунок 16. UML-диаграмма деятельности 2 индивидуального задания

Задание 3. Определить среди всех двузначных чисел те, которые делятся на сумму своих цифр.



```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import math
5  import sys
6
7  if __name__ == '__main__':
8      for i in range(10, 100):
9          if (i % (i % 10 + i // 10)) == 0:
10             print(i)
11

```

Рисунок 17. Результат работы 3 индивидуальной программы

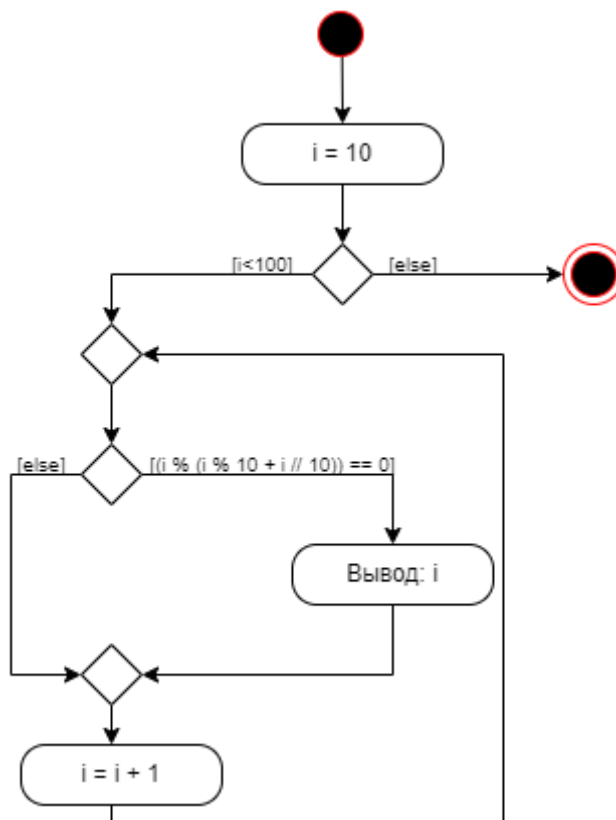


Рисунок 18. UML-диаграмма деятельности 3 индивидуального задания

Задание повышенной сложности. Функция Бесселя первого рода, значение также должно вводиться с клавиатуры.

$$I_n(x) = \left(\frac{x}{2}\right)^n \sum_{k=0}^{\infty} \frac{(x^2/4)^k}{k!(k+n)!}.$$

```

1  C:\usr\bin\env python3
2  C# -*- coding: utf-8 -*-
3
4  import ...
5
6
7  if __name__ == '__main__':
8      eps = 1e-10
9      k = 0
10     n = int(input("Enter n: "))
11     x = float(input("Enter x: "))
12
13     s = 0
14     a = 1/math.factorial(n)
15     while abs(a) > eps:
16         s += a
17
18     if __name__ == '__main__':

```

D:\phyton\python.exe C:/Users/Дом/Desktop/Дина/Документы/СКФУ/Программирование/Lab2.2/prog/slo.py  
Enter n: 50  
Enter x: 43  
The Bessel function of the first kind = 0.5816811860235157  
Process finished with exit code 0

Рисунок 19. Результат работы программы

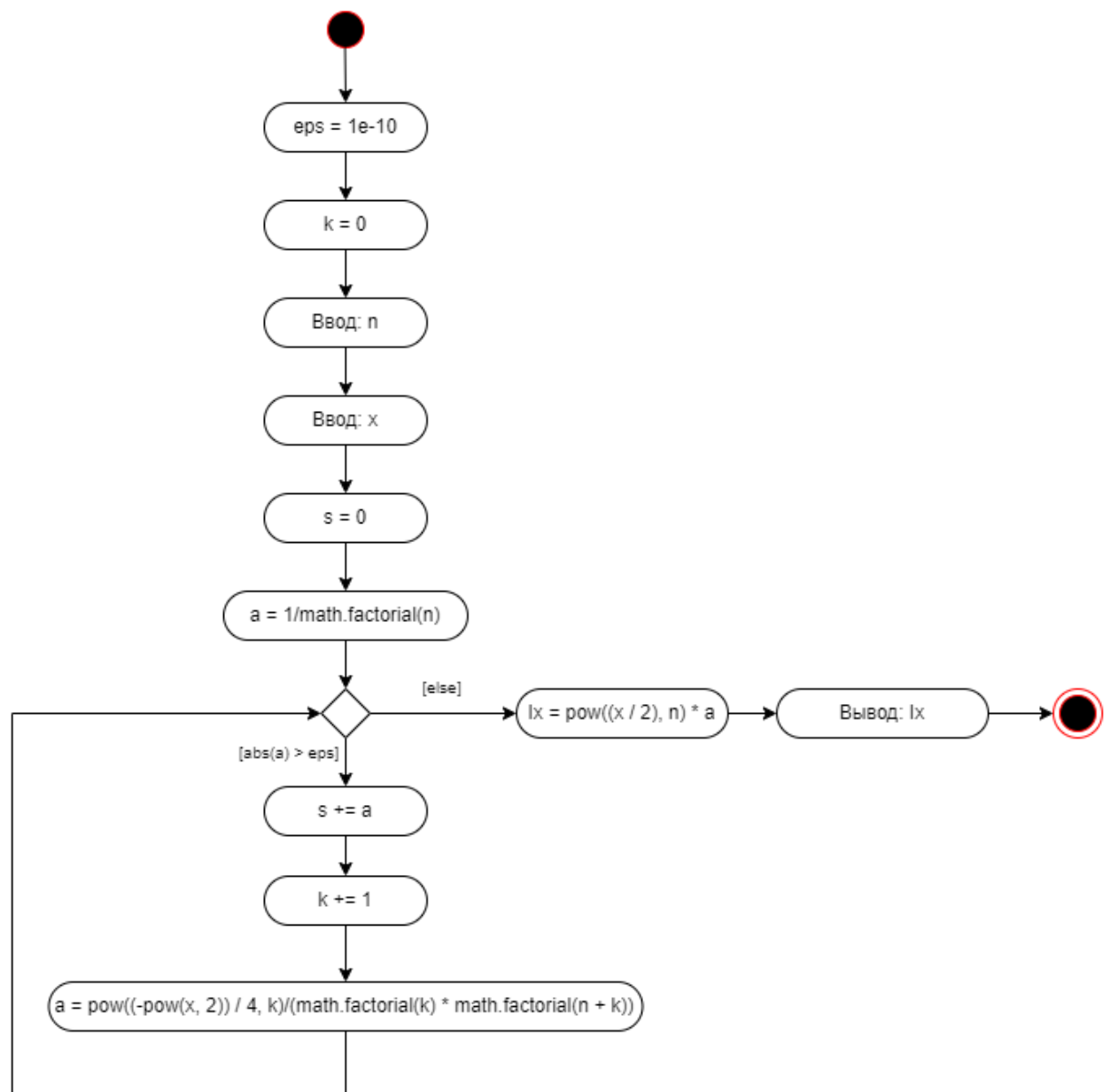


Рисунок 20. UML-диаграмма деятельности

## **Ответы на вопросы:**

### **1. Для чего нужны диаграммы деятельности UML?**

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.

### **2. Что такое состояние действия и состояние деятельности?**

Состояние действия и состояние деятельности. В потоке управления, моделируемом диаграммой деятельности, происходят различные события. 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.

### **3. Какие нотации существуют для обозначения переходов и ветвлений в диаграммах деятельности?**

Когда действие или деятельность в некотором состоянии завершается, поток

управления сразу переходит в следующее состояние действия или деятельности. Для описания этого потока используются переходы, показывающие путь из одного состояния действия или деятельности в другое. В UML переход представляется простой линией со стрелкой.

Поток управления должен где-то начинаться и заканчиваться (разумеется, если это не бесконечный поток, у которого есть начало, но нет конца). Как показано на рисунке, вы можете задать как начальное состояние

(закрашенный кружок), так и конечное (закрашенный кружок внутри окружности).

4. Какой алгоритм является алгоритмом разветвляющейся структуры?

Состояние действия и состояние деятельности. В потоке управления, моделируемом диаграммой деятельности, происходят различные события. 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.

5. Чем отличается разветвляющийся алгоритм от линейного?

Линейный алгоритм - алгоритм, все этапы которого выполняются однократно и строго последовательно. Разветвляющийся алгоритм - алгоритм, содержащий хотя бы одно условие, в результате проверки которого ЭВМ обеспечивает переход на один из двух возможных шагов.

6. Что такое условный оператор? Какие существуют его формы?

Условные операторы – это специальные конструкции, благодаря которым в программе происходит ветвление в зависимости от условий.

Условный оператор имеет полную и краткую формы.

7. Какие операторы сравнения используются в Python?

If, elif, else.

8. Что называется простым условием? Приведите примеры.

Условные операторы – это специальные конструкции, благодаря которым в программе происходит ветвление в зависимости от условий.

Пример: `a == b`

9. Что такое составное условие? Приведите примеры.

Составное условие – логическое выражение, содержащее несколько простых условий объединенных логическими операциями. Это операции not, and, or.

Пример: (a == b or a == c)

10. Какие логические операторы допускаются при составлении сложных условий?

Not, and, or.

11. Может ли оператор ветвления содержать внутри себя другие ветвления?

Может.

12. Какой алгоритм является алгоритмом циклической структуры?

Циклический алгоритм — это вид алгоритма, в процессе выполнения которого одно или несколько действий нужно повторить.

13. Типы циклов в языке Python.

В Python есть 2 типа циклов: - цикл while, - цикл for.

Оператор for выполняет указанный набор инструкций заданное количество раз, которое определяется количеством элементов в наборе.

Оператор цикла while выполняет указанный набор инструкций до тех пор, пока условие цикла истинно.

14. Назовите назначение и способы применения функции range .

Функция range генерирует серию целых чисел, от значения start до stop, указанного пользователем. Мы можем использовать его для цикла for и обходить весь диапазон как список.

15. Как с помощью функции range организовать перебор значений от 15 до 0 с шагом 2?

for i in range(15, 0, -2).

16. Могут ли быть циклы вложенными?

Могут.

17. Как образуется бесконечный цикл и как выйти из него?

Бесконечный цикл в программировании — цикл, написанный таким образом, что условие выхода из него никогда не выполняется. Чтобы выйти из цикла нужно использовать оператор `break`.

18. Для чего нужен оператор `break`?

Используется для выхода из цикла.

19. Где употребляется оператор `continue` и для чего он используется?

Оператор `continue` используется только в циклах. В операторах `for` , `while` , `do while` , оператор `continue` выполняет пропуск оставшейся части кода тела цикла и переходит к следующей итерации цикла.

20. Для чего нужны стандартные потоки `stdout` и `stderr`?

Ввод и вывод распределяется между тремя стандартными потоками: `stdin` — стандартный ввод (клавиатура), `stdout` — стандартный вывод (экран), `stderr` — стандартная ошибка (вывод ошибок на экран)

21. Как в Python организовать вывод в стандартный поток `stderr`?

Указать в `print(..., file=sys.stderr)`.

22. Каково назначение функции `exit`?

Функция `exit()` модуля `sys` - выход из Python.

**Вывод:** в результате выполнения работы были приобретены навыки программирования разветвляющихся алгоритмов и алгоритмов циклической структуры; освоены операторы языка Python версии 3.x `if` , `while` , `for` , `break` и `continue` , позволяющие реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.