МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ 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

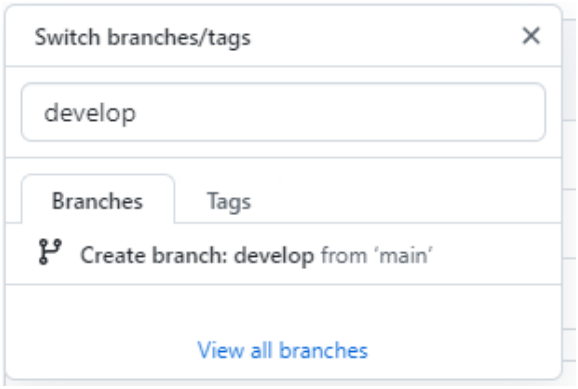
Выполнил:
Шальнев Владимир Сергеевич,
2 курс, группа ПИЖ-б-о-20-1,

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

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2021 г.

ВЫПОЛНЕНИЕ:



Создание ветки develop

```
F:\pythonProject>git clone https://github.com/HAXF13D/laboratory-5
Cloning into 'laboratory-5'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (11/11), done.
```

Клонирование репозитория

```
F:\pythonProject\laboratory-5>git branch develop
```

Переход на ветку develop

```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? 3
y = 4.0
```

Значение №1

```
Value of x? -5
y = 50.28366218546323
```

Значение №2

```
Value of x? 10
y = -100.54402111088937
```

Значение №3

```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: 10
Осень
```

Значение №1

```
Введите номер месяца: 1
Зима
```

Значение №2

```
Введите номер месяца: 3
Весна
```

Значение №3

```
Введите номер месяца: 6
Лето
```

Значение №4

```
Введите номер месяца: 15
Ошибка!
```

Значение №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? 10
Value of x? 2
S = 1.6927197774199492
```

Значение №1

```
Value of n? 10
Value of x? 20
S = 5.26119185280723
```

Значение №2

```
Value of n? 0
Value of x? 1
S = 0.0
```

Значение №3

```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? 5
x = 2.23606797749979
X = 2.23606797749979
```

Значение №1

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

Значение №2

```
Value of a? -1
Illegal value of a
```

Значение №3

```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? 10
Ei(10.0) = 2492.228976241855
```

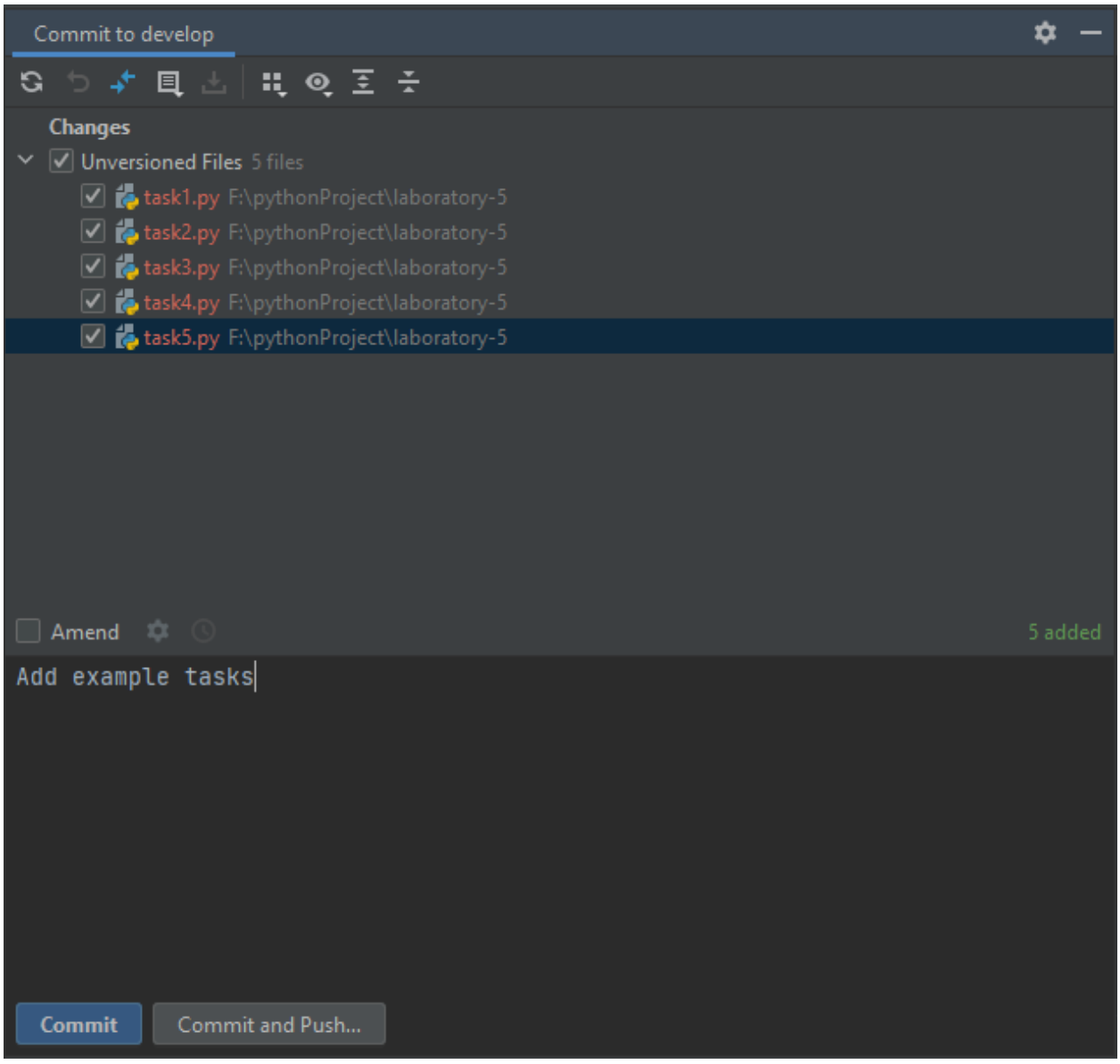
Значение №1

```
Value of x? 0
Illegal value of x
```

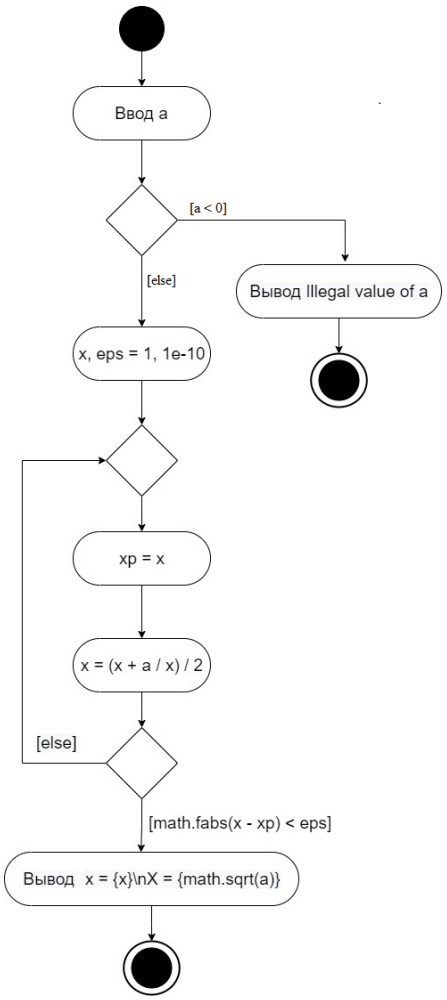
Значение №2

```
Value of x? 100
Ei(100.0) = 2.7155527448538826e+41
```

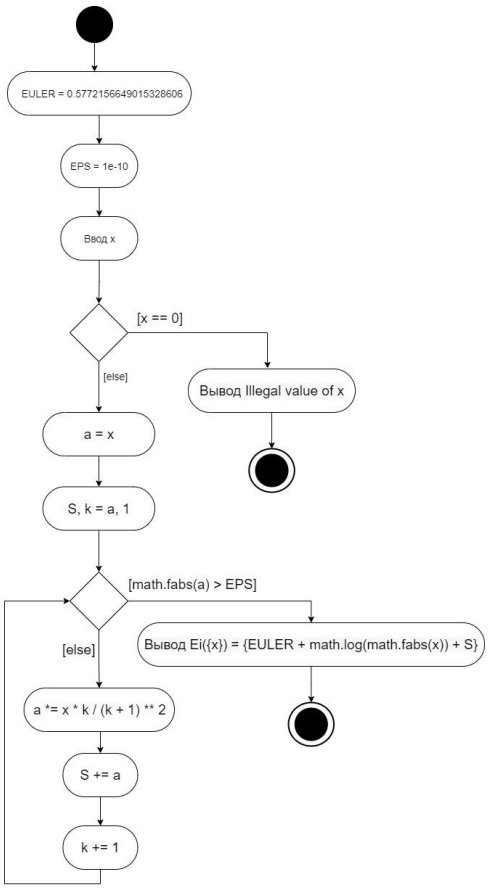
Значение №3



Коммит изменений



UML-диаграмма примера 4



UML-диаграмма примера 5

```
"""Variant 12"""

x1 = int(input("Price per roll of wallpaper: "))
x2 = int(input("Price for a can of paint: "))

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

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

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

Индивидуальное задание 1

```
Price per roll of wallpaper: 10
Price for a can of paint: 10
Total price = 100.00
```

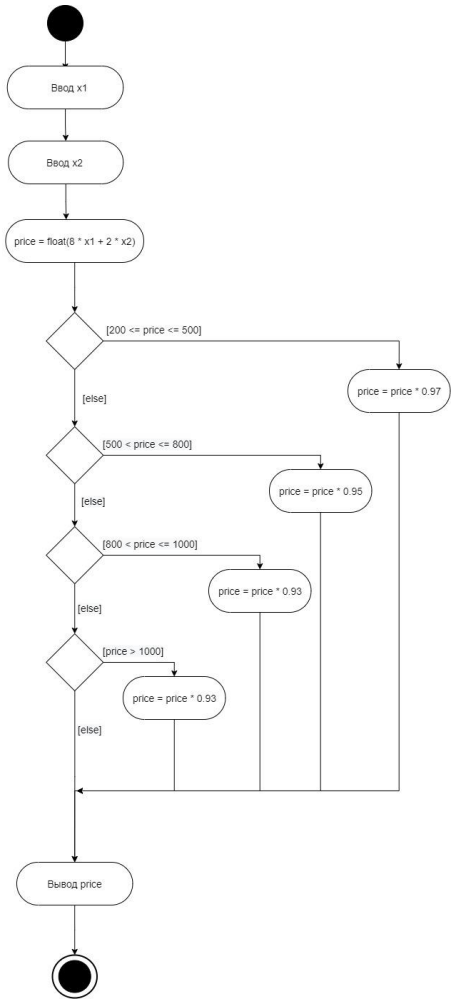
Значение №1

```
Price per roll of wallpaper: 20
Price for a can of paint: 20
Total price = 194.00
```

Значение №2

```
Price per roll of wallpaper: 500
Price for a can of paint: 500
Total price = 4550.00
```

Значение №3



UML-диаграмма индивидуального задания 1

```python
"""Variant 2"""

x = int(input("x: "))
y = int(input("y: "))

if x * x * y > x * y * y:
    ma = x * x * y
else:
    ma = x * y * y

if x - y < x + 2 * y:
    mi = x - y
else:
    mi = x + 2 * y

u = ma ** 2 + mi ** 2
print(u)
```

Индивидуальное задание 2

```
x: 10
y: 5
250025
```

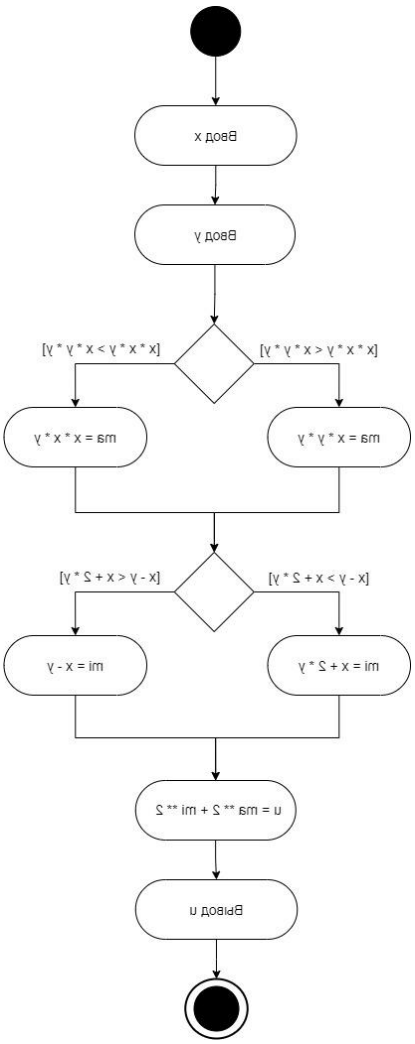
Значение №1

```
x: 12
y: 32
150995344
```

Значение №2

```
x: 10
y: 1
10081
```

Значение №3



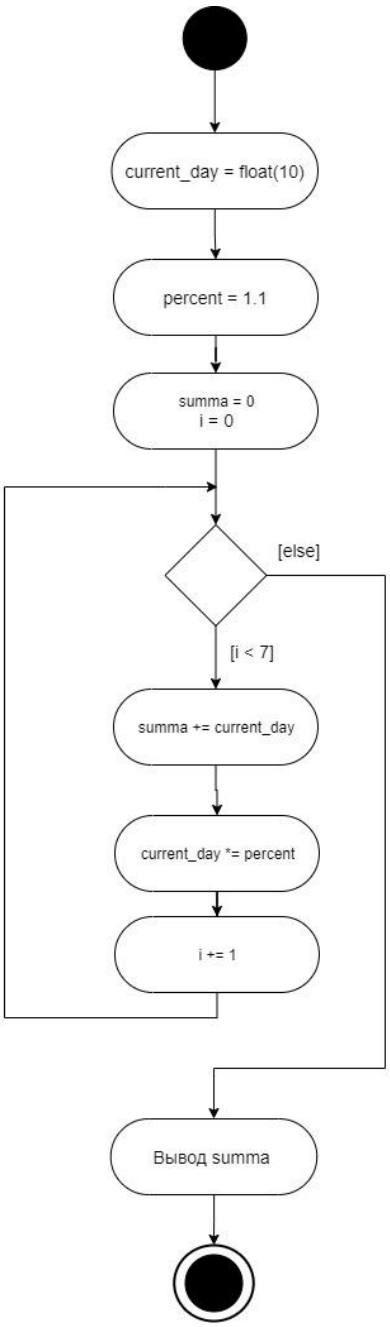
UML-диаграмма индивидуального задания 2

```
"""Variant 3"""

current_day = float(10)
percent = 1.1
summa = 0

for i in range(7):
    summa += current_day
    current_day *= percent

print("Ran in 7 days: %.3f" % summa)
```

Индивидуальное задание 3

```
Ran in 7 days: 94.872
```

Результат работы программы



UML-диаграмма индивидуального задания 3

```python
""" Variant 7 """

from math import factorial as fact
from math import fabs

EPS = 1e-10

x = float(input("x = "))
n = int(input("n = "))
k = 0
var = (x / 2) ** n
summa = float(0)

while True:
    prev_sum = summa
    temp = ((x ** 2) / 4) ** k
    summa = temp / (fact(k) * fact(k + n))
    k += 1
    if fabs(prev_sum - summa) < EPS:
        break

summa = var * summa

print(summa)
```

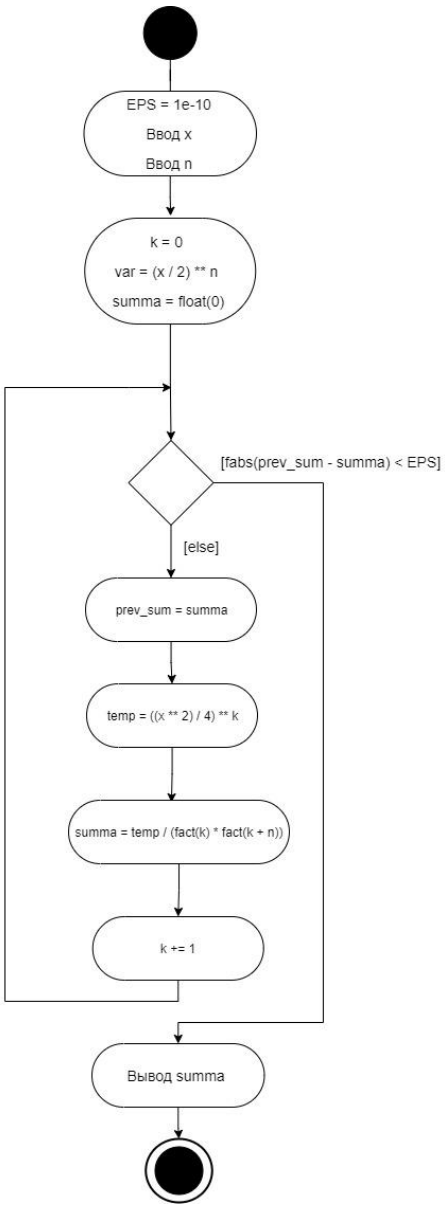
Задание повышенной сложности

```
x = 3
n = 6
2.1285253347032532e-12
```

Значение №1

```
x = 8
n = 15
0.0008211079533830857
```

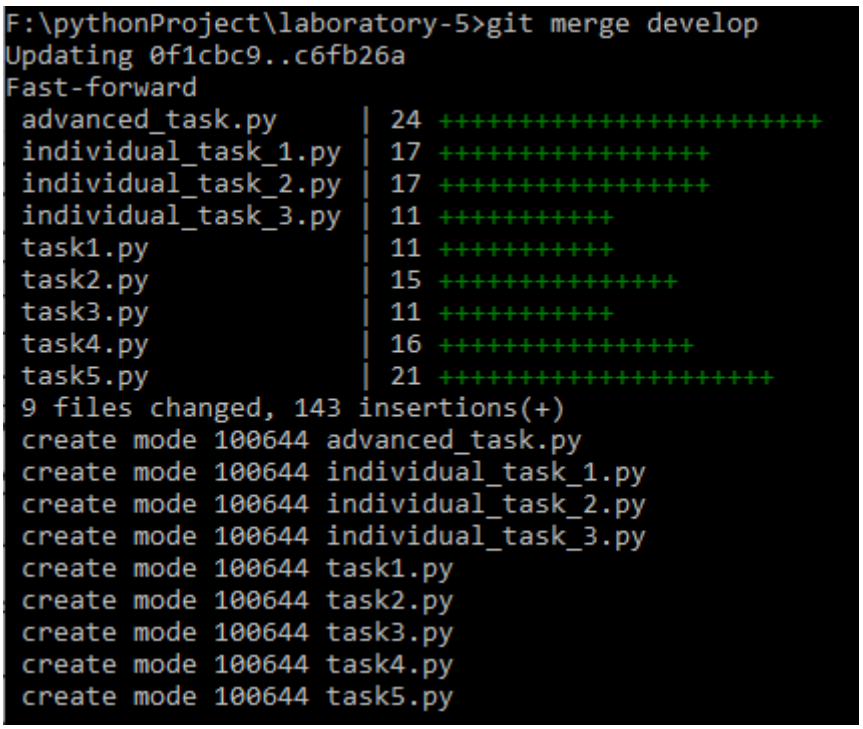
Значение №2



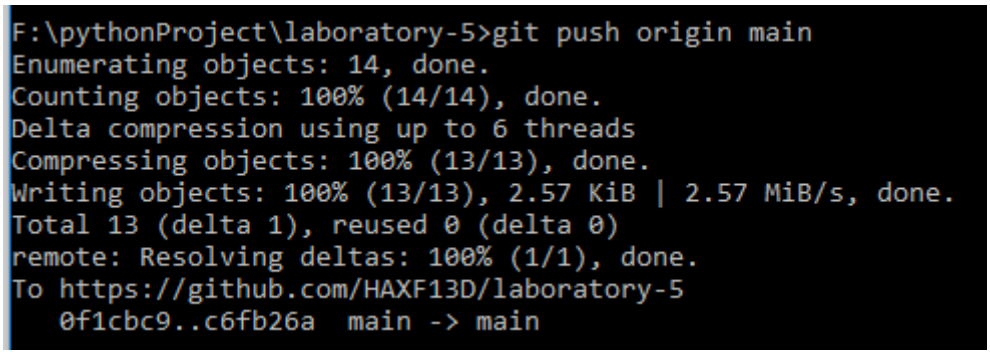
UML-диаграмма задания повышенной сложности

```
F:\pythonProject\laboratory-5>git add .

F:\pythonProject\laboratory-5>git commit -m "Add individual tasks"
[develop c6fb26a] Add individual tasks
 4 files changed, 69 insertions(+)
 create mode 100644 advanced_task.py
 create mode 100644 individual_task_1.py
 create mode 100644 individual_task_2.py
 create mode 100644 individual_task_3.py
```

Коммит изменений



Слияние двух веток



Отправка изменения на сервер

ОТВЕТЫ НА ВОПРОСЫ:

1. Диаграммы деятельности - это один из пяти видов диаграмм, применяемых в UML для моделирования динамических аспектов поведения системы. Диаграмма деятельности - это, блок-схема, которая показывает, как поток управления переходит 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.

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 == 8:

break

```
a += 2
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 заданный код возврата можно посредством функции exit.