

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №5
дисциплины
«Основы кроссплатформенного программирования»
Вариант 4

Выполнил:
Борцов Богдан Михайлович
2 курс, группа ИТС-б-о-23-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи», очная
форма обучения

(подпись)

Проверил:
Доцент департамента цифровых,
робототехнических систем и
электроники Воронкин Р.А.

(подпись)

Ставрополь, 2024 г.

Тема: Условные операторы и циклы в языке Python

Цель: исследовать условные операторы и циклы в языке Python

Порядок выполнения работы:

1. Изучил теоретический материал.
2. Приступил к выполнению заданий.
3. Создал общедоступны репозиторий.
4. Выполнил клонирование репозитория
5. Изучил рекомендации PEP-8
6. Создал проект
7. Приступил к проработке примеров

```
#!/usr/bin/env python3
```

```
# -*- coding: utf-8 -*-
```

```
import math
```

```
if __name__ == '__main__':
```

```
    x = float(input("Value of x? "))
```

```
    if x <= 0:
```

```
        y = 2 * x * x + math.cos(x)
```

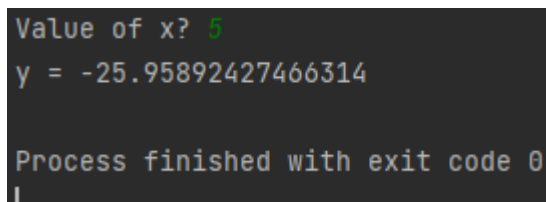
```
    elif x < 5:
```

```
        y = x + 1
```

```
    else:
```

```
        y = math.sin(x) - x * x
```

```
    print(f'y = {y}')
```



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

Process finished with exit code 0
|
```

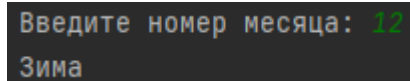
Рисунок 1. Результат первого примера

```

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)

```



```

Введите номер месяца: 12
Зима

```

Рисунок 2. Результат второго примера

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import math

if __name__ == '__main__':
    n = int(input("Value of n? "))
    x = float(input("Value of x? "))
    S = 0.0

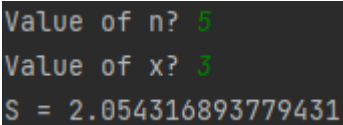
    for k in range(1, n + 1):

```

```

a = math.log(k * x) / (k * k)
S += a
print(f"S = {S}")

```



```

Value of n? 5
Value of x? 3
S = 2.054316893779431

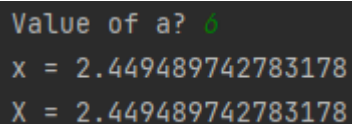
```

Рисунок 3. Результат третьего примера

```

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)}")

```



```

Value of a? 6
x = 2.449489742783178
X = 2.449489742783178

```

Рисунок 4. Результат четвертого примера

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

```

```

import math

```

```

import sys

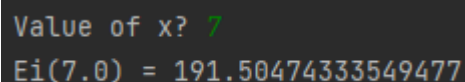
# Постоянная Эйлера.
EULER = 0.5772156649015328606

# 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}")

```



```

Value of x? 7
Ei(7.0) = 191.50474333549477

```

Рисунок 5. Результат пятого примера

8. Зафиксировал изменения в репозитории
9. Приступил к выполнению индивидуального задания

```

# Ввод натурального числа
n = int(input("Enter a natural number (n): "))

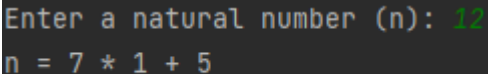
```

```

# Вычисление частного и остатка
k = n // 7
r = n % 7

# Вывод результата
if r == 0:
    print(f"n = 7 * {k}")
else:
    print(f"n = 7 * {k} + {r}")

```



Enter a natural number (n): 12
n = 7 * 1 + 5

Рисунок 6. Результат задания 1

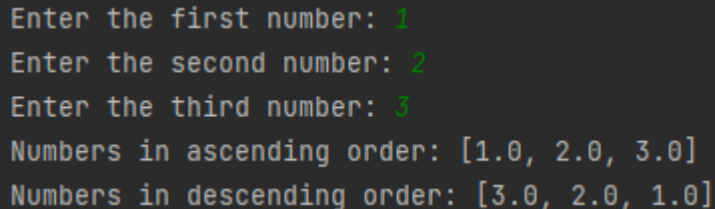
```

# Ввод трёх действительных чисел
a = float(input("Enter the first number: "))
b = float(input("Enter the second number: "))
c = float(input("Enter the third number: "))

# Сортировка чисел
numbers = [a, b, c]
ascending_order = sorted(numbers) # По возрастанию
descending_order = sorted(numbers, reverse=True) # По убыванию

# Вывод результата
print(f"Numbers in ascending order: {ascending_order}")
print(f"Numbers in descending order: {descending_order}")

```



Enter the first number: 1
Enter the second number: 2
Enter the third number: 3
Numbers in ascending order: [1.0, 2.0, 3.0]
Numbers in descending order: [3.0, 2.0, 1.0]

Рисунок 7. Результат задания 2

```

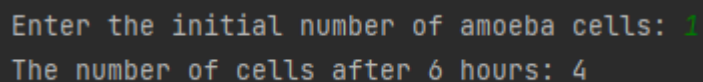
# Ввод начального количества клеток
initial_cells = int(input("Enter the initial number of amoeba cells: "))

```

```
# Количество делений за 6 часов (каждые 3 часа)
divisions = 6 // 3

# Вычисление общего числа клеток
final_cells = initial_cells * (2 ** divisions)

# Вывод результата
print(f"The number of cells after 6 hours:
{final_cells}")
```



```
Enter the initial number of amoeba cells: 1
The number of cells after 6 hours: 4
```

Рисунок 8. Результат задания 3

10. Зафиксировал изменения

Контрольные вопросы

1. Для чего нужны диаграммы деятельности UML? Диаграммы деятельности UML используются для моделирования бизнес-процессов и алгоритмов.

Они помогают визуализировать последовательность действий, условия и потоки управления в системе.

2. Что такое состояние действия и состояние деятельности?

Состояние действия — это конкретное действие или операция, выполняемая в процессе. Состояние деятельности — это более общее состояние, которое может включать в себя несколько действий.

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

Переходы обозначаются стрелками. Ветвления обозначаются ромбами, где условия перехода указываются на стрелках.

4. Какой алгоритм является алгоритмом разветвляющейся структуры?

Алгоритм разветвляющейся структуры — это алгоритм, который принимает решение на основе условий и выполняет разные действия в зависимости от результата.

5. Чем отличается разветвляющийся алгоритм от линейного?

Линейный алгоритм выполняет действия последовательно, без условий, а разветвляющийся алгоритм включает условия, которые определяют, какие действия выполнять.

6. Что такое условный оператор? Какие существуют его формы? Условный оператор позволяет выполнять разные действия в зависимости от истинности условия. Основные формы: if if-else if-elif-else

7. Какие операторы сравнения используются в Python?

= равно, != не равно, >, <, >=, <=.

8. Что называется простым условием?

Простое условие — это условие, состоящее из одного логического выражения.

9. Что такое составное условие?

Составное условие — это условие, состоящее из нескольких логических выражений, объединенных логическими операторами.

10. Какие логические операторы допускаются при составлении сложных условий?

And, or, not

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

Да, оператор ветвления может содержать другие ветвления, создавая вложенные условия.

12. Какой алгоритм является алгоритмом циклической структуры?

Алгоритм циклической структуры выполняет одно и то же действие несколько раз, пока выполняется определенное условие.

13. Типы циклов в языке Python.

for — цикл, который перебирает элементы последовательности. while — цикл, который выполняется, пока истинно заданное условие.

14. Назовите назначение и способы применения функции range.

Функция range используется для генерации последовательности чисел. Она часто применяется в циклах для итерации по числовым диапазонам.

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

сору for i in range(15, -1, -2)

16. Могут ли быть циклы вложенными?

Да, циклы могут быть вложенными, что позволяет создавать более сложные структуры итерации.

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

Бесконечный цикл образуется, когда условие цикла всегда истинно. Выйти из него можно с помощью оператора break или прерывания программы.

18. Для чего нужен оператор break?

Оператор `break` используется для немедленного выхода из цикла, прекращая его выполнение.

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

Оператор `continue` используется для пропуска текущей итерации цикла и перехода к следующей.

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

`stdout` используется для вывода информации, `stderr` используется для вывода сообщений об ошибках.

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

через модуль `sys`

22. Каково назначение функции `exit`?

Функция `exit` используется для завершения программы. Она может принимать код завершения, который указывает на успешное или неуспешное завершение.

Вывод: В ходе выполнения лабораторной работы были исследованы конструкции ветвления и циклов в Python для решения задач разного уровня сложности. Реализованы алгоритмы обработки чисел, работы с датами, определения свойств чисел и геометрических объектов. Также проработаны программы с использованием математических функций и моделирования реальных процессов, таких как расчёт стоимости, вычисление траекторий и анализ данных. Особое внимание уделено разработке структурированных решений, составлению UML-диаграмм деятельности и применению оптимальных конструкций языка. Работа позволила закрепить навыки программирования и систематизировать подход к решению задач.