

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

Институт цифрового развития
Кафедра инфокоммуникаций

«Условные операторы и циклы в языке Python»

ОТЧЕТ
по лабораторной работе №5
дисциплины
«Основы программной инженерии»

Выполнил:

Гълбачева Доротея Андреева
2 курс, группа ПИЖ-б-о-21-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил:

(подпись)

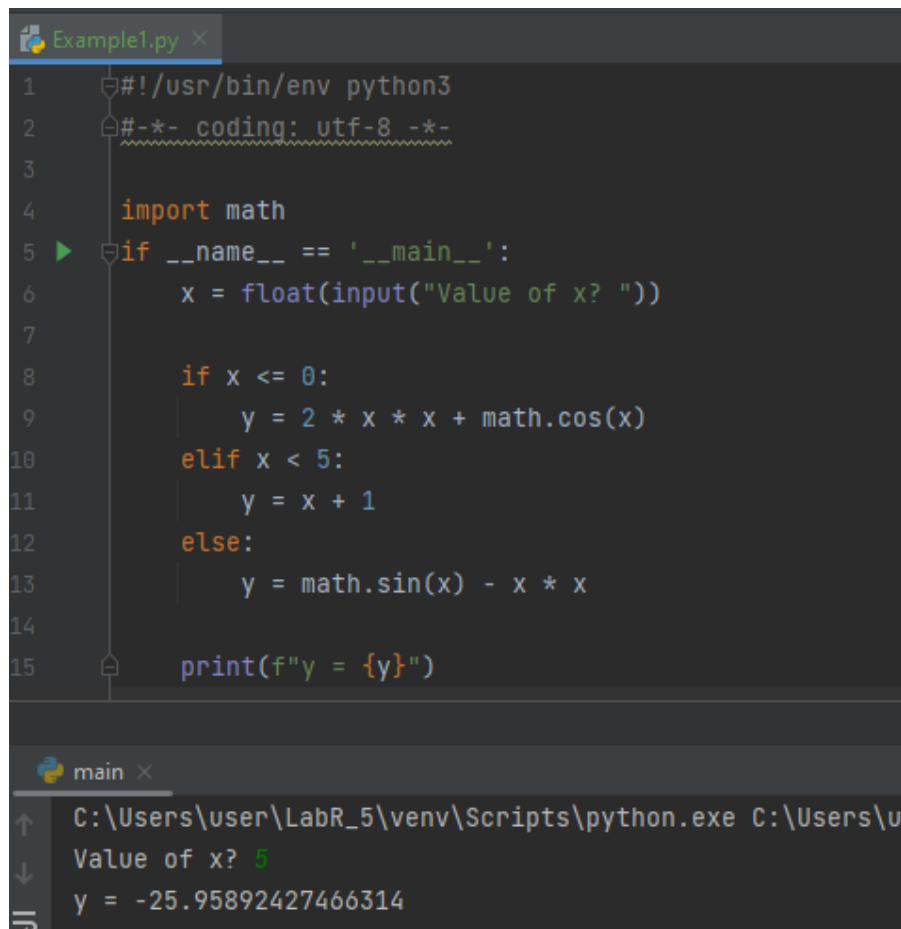
Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

Цель работы: 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.

Методика и порядок выполнения работы:

1. Проработка примеров из лабораторной работы.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import math
5  if __name__ == '__main__':
6      x = float(input("Value of x? "))
7
8      if x <= 0:
9          y = 2 * x * x + math.cos(x)
10     elif x < 5:
11         y = x + 1
12     else:
13         y = math.sin(x) - x * x
14
15     print(f"y = {y}")
```

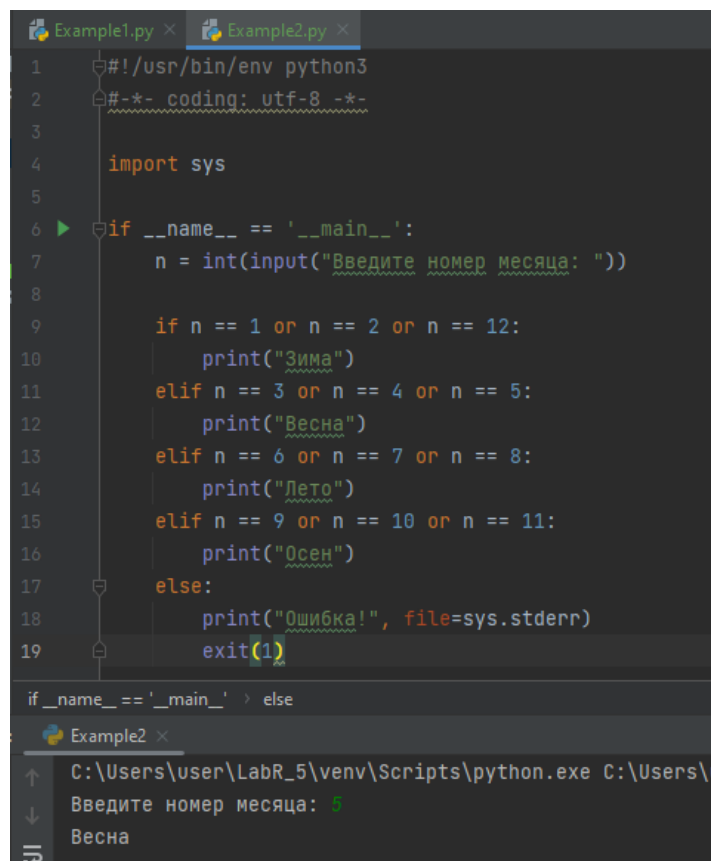
main ×

C:\Users\user\LabR_5\venv\Scripts\python.exe C:\Users\user\LabR_5\venv\Scripts\python.exe

Value of x? 5

y = -25.95892427466314

Рисунок 5.1 – Пример №1



```
1  #!/usr/bin/env python3
2  #-*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      n = int(input("Введите номер месяца: "))
8
9      if n == 1 or n == 2 or n == 12:
10         print("Зима")
11     elif n == 3 or n == 4 or n == 5:
12         print("Весна")
13     elif n == 6 or n == 7 or n == 8:
14         print("Лето")
15     elif n == 9 or n == 10 or n == 11:
16         print("Осен")
17     else:
18         print("Ошибка!", file=sys.stderr)
19     exit(1)
```

if __name__ == '__main__' > else

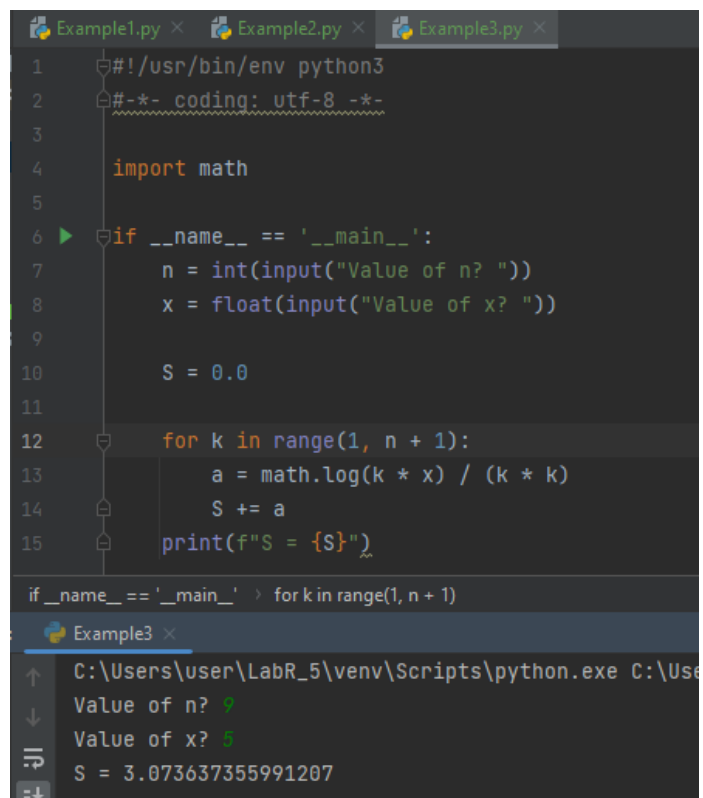
Example2 ×

↑ C:\Users\User\LabR_5\venv\Scripts\python.exe C:\Users\U

↓ Введите номер месяца: 5

Весна

Рисунок 5.2 – Пример №2



```
1  #!/usr/bin/env python3
2  #-*- coding: utf-8 -*-
3
4  import math
5
6  if __name__ == '__main__':
7      n = int(input("Value of n? "))
8      x = float(input("Value of x? "))
9
10     S = 0.0
11
12     for k in range(1, n + 1):
13         a = math.log(k * x) / (k * k)
14         S += a
15     print(f"S = {S}")
```

if __name__ == '__main__' > for k in range(1, n + 1)

Example3 ×

↑ C:\Users\User\LabR_5\venv\Scripts\python.exe C:\Use

↓ Value of n? 9

Value of x? 5

S = 3.073637355991207

Рисунок 5.3 – Пример №3

```

Example1.py x Example2.py x Example3.py x Example4.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import math
5  import sys
6
7  if __name__ == '__main__':
8      a = float(input("Value of a? "))
9      if a < 0:
10         print("Illegal value of a", file=sys.stderr)
11         exit(1)
12
13         x, eps = 1, 1e-10
14         while True:
15             xp = x
16             x = (x + a / x) / 2
17             if math.fabs(x - xp) < eps:
18                 break
19
20         print(f"x = {x}\nx = {math.sqrt(a)}")

```

if __name__ == '__main__' > while True

Example4 x

↑ x = 7.3484692283495345

↓ x = 7.3484692283495345

Рисунок 5.4 – Пример №4

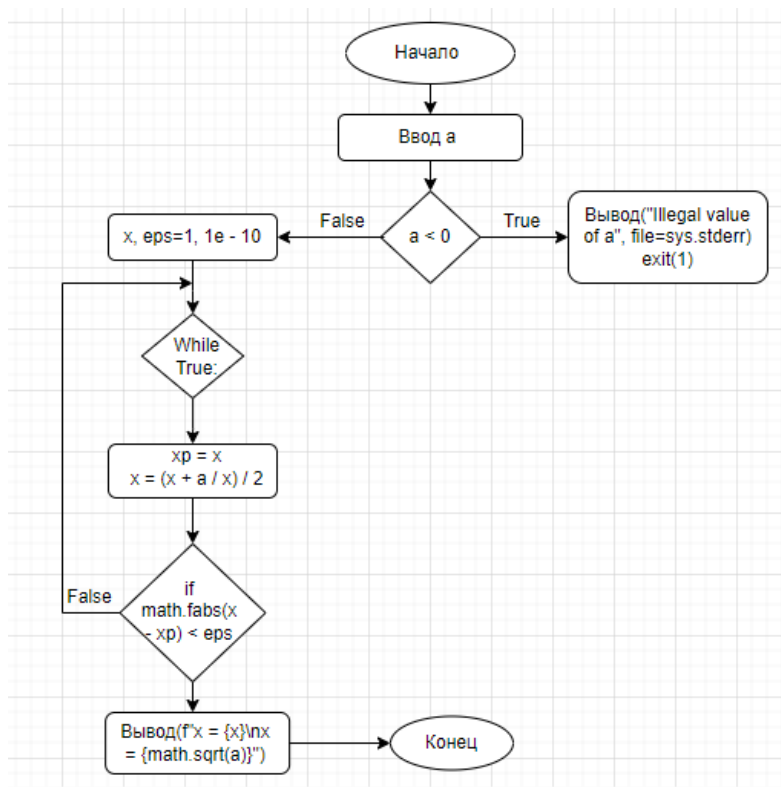


Рисунок 5.5 – UML диаграме примера №4

```

# 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}")

```

Example5 x

Value of x? 30

Ei(30.0) = 368973209407.2745

Рисунок 5.6 – Пример №5

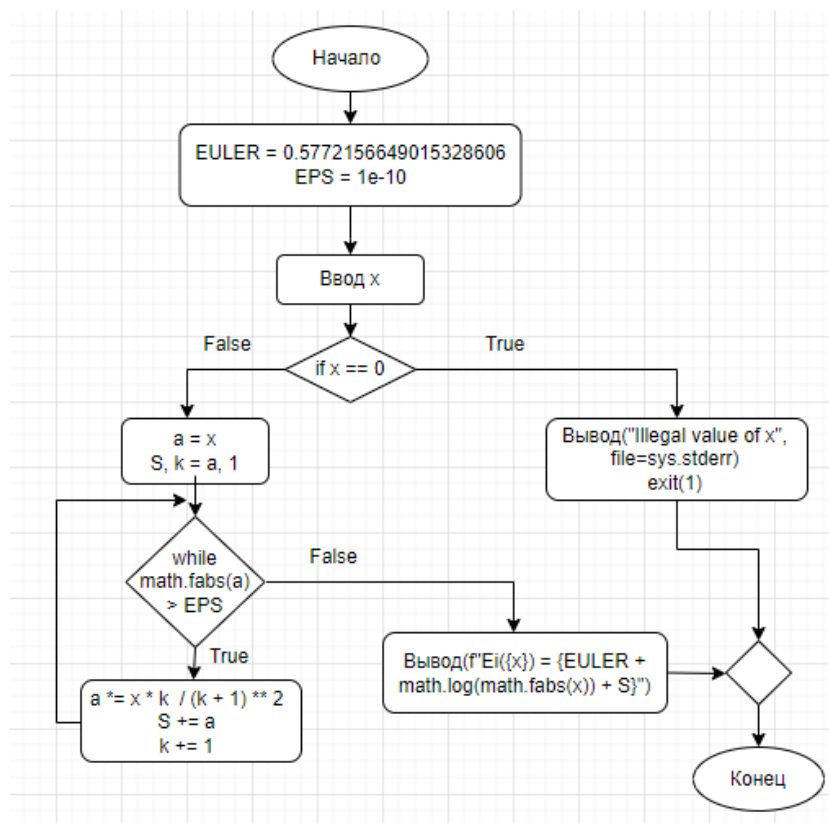


Рисунок 5.6 – UML диаграмма примера №5

Индивидуальные задания:

Задание 1:

8. Дано число m ($1 \leq m \leq 12$). Определить полугодие, на которое приходится месяц с номером m и количество дней в том месяце (год не високосный).

```
Example1.py x Example2.py x Example3.py x Example4.py x Example5.py x Individual.py
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == '__main__':
    m = int(input("Введите номер месяца: "))

    print("Первое полугодие" if m in range(1,7) else "Второе полугодие")

    if m == 1 or m == 3 or m == 5 or m == 7 or m == 8 or m == 10 or m == 12:
        print("Месяц имеет 31 дня")
    elif m == 2:
        print("Месяц имеет 28 дней")
    else:
        print("Месяц имеет 30 дней")

name__ == '__main__' > else
Individual x
C:\Users\user\LabR_5\venv\Scripts\python.exe C:\Users\user\LabR_5\Individual.py
Введите номер месяца: 9
Второе полугодие
Месяц имеет 30 дней
```

Рисунок 5.7 – Код и результат работы программы №1

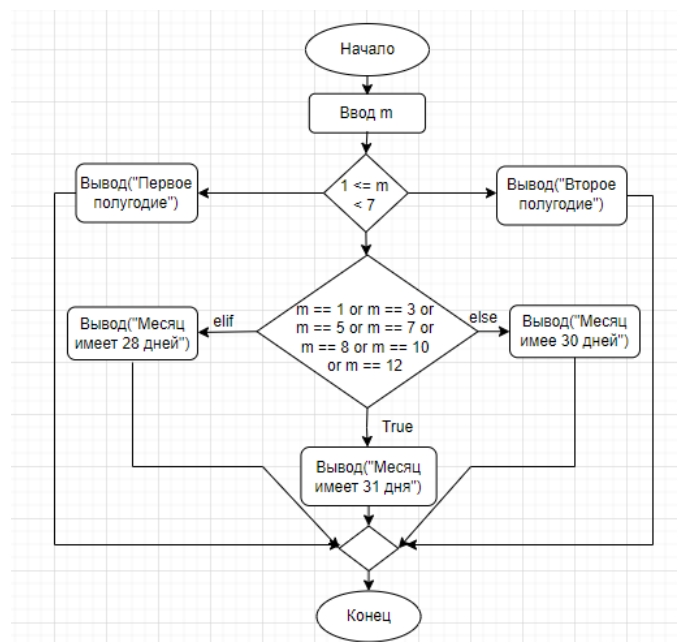
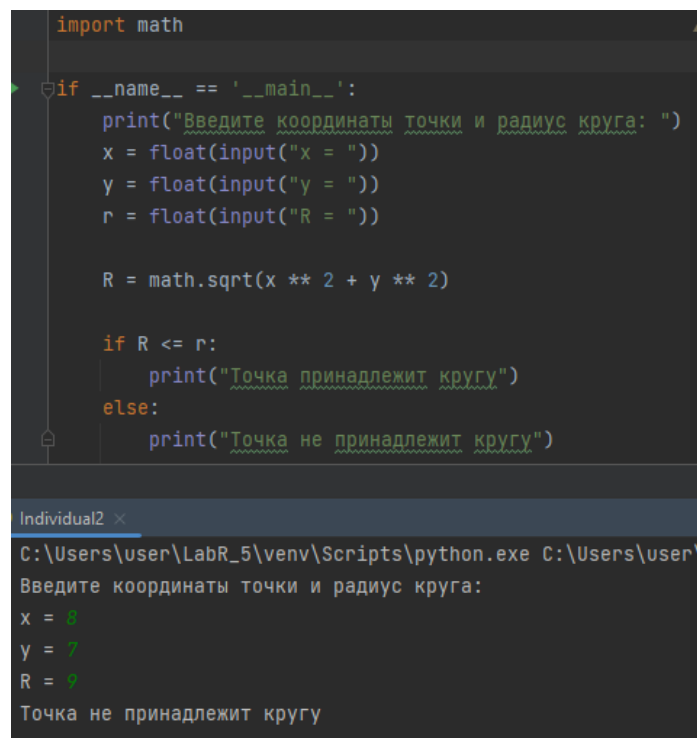


Рисунок 5.8 – UML диаграмма примера №1

Задание 2:



```
import math

if __name__ == '__main__':
    print("Введите координаты точки и радиус круга: ")
    x = float(input("x = "))
    y = float(input("y = "))
    r = float(input("R = "))

    R = math.sqrt(x ** 2 + y ** 2)

    if R <= r:
        print("Точка принадлежит кругу")
    else:
        print("Точка не принадлежит кругу")
```

Individual2 x

C:\Users\user\LabR_5\venv\Scripts\python.exe C:\Users\user\

Введите координаты точки и радиус круга:

x = 8

y = 7

R = 9

Точка не принадлежит кругу

Рисунок 5.9 – Код и результат работы программы №2

Задание 3:

8. Сумма цифр трехзначного числа кратна 7. Само число также делится на 7. Найти все такие числа.

```
Example4.py x Example5.py x Individual.py x Individual2.py x Individual3 x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import math
5
6  if __name__ == '__main__':
7      for num in range(100, 1000):
8          a = num // 100
9          b = (num % 100) // 10
10         c = num % 10
11         sum = a + b + c
12
13         if sum % 7 == 0 and num % 7 == 0:
14             print(num)
```

if __name__ == '__main__' > for num in range(100, 1000) > if sum % 7 == 0 and num % 7 == 0

Individual3 x

C:\Users\user\LabR_5\venv\Scripts\python.exe C:\Users\user\LabR_5\venv\Scripts\python.exe C:\Users\user\LabR_5\venv\Scripts\python.exe

133
266
322
329
392
399
455
511
518
581

Рисунок 5.11 - Код и результат работы программы №3

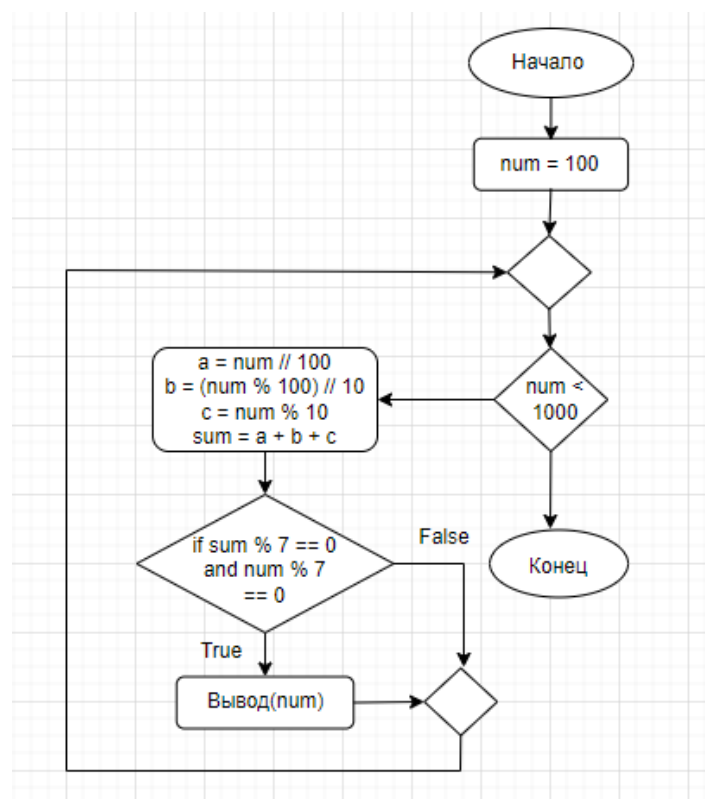


Рисунок 5.12 – UML диаграмма примера №3

Контрольные вопросы:

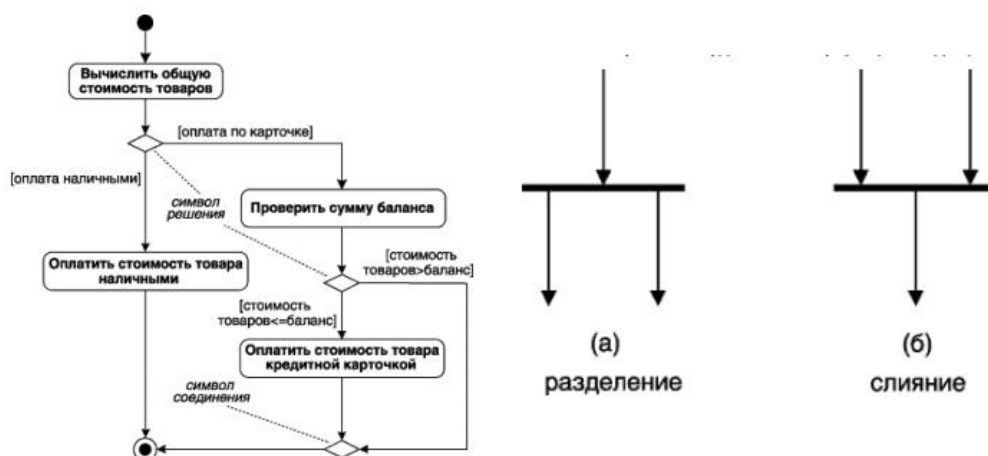
1. Для чего нужны диаграммы деятельности UML?

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?

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. 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.

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



4. Какой алгоритм является алгоритмом разветвляющейся структуры?

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

5. Чем отличается разветвляющийся алгоритм от линейного?

Наличием условных операторов

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

Команда, которая выполняется только при каком-либо условии

7. Какие операторы сравнения используются в Python?

Оператор `<` , «меньше»; оператор `<=` , «меньше или равно»; оператор `==` , «равно»; оператор `!=` , «не равно»; оператор `>` , «больше»; оператор `>=` , «больше или равно».

8. Что называется простым условием? Приведите примеры.

Простым условием (отношением) называется выражение, составленное из двух арифметических выражений или двух текстовых величин (иначе их еще называют операндами), связанных одним из знаков приведенных в ответе на 7 вопрос

9. Что такое составное условие? Приведите примеры.

Составное условие – логическое выражение, содержащее несколько простых условий, объединенных логическими операциями.

10. Какие логические операторы допускаются при составлении сложных условий?

Логическое И, логическое ИЛИ, логическое отрицание

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

Да

12. Какой алгоритм является алгоритмом циклической структуры?

Алгоритм циклической структуры — это алгоритм, в котором происходит многократное повторение одного и того же участка

программы. Такие повторяемые участки вычислительного процесса называются циклами.

13. Типы циклов в языке Python.

В Python есть два вида циклов: `for` и `while`

14. Назовите назначение и способы применения функции `range`
Функция `range` возвращает неизменяемую последовательность чисел в виде объекта `range`.

Синтаксис функции: `range(stop)` `range(start, stop[, step])` `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

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

`range(0, 15, 2)`

16. Могут ли быть циклы вложенными?

Вложенный цикл - цикл который выполняется внутри другого цикла. Обычно вложенные циклы используются для работы с двумя измерениями. Да могут

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

Пример бесконечного цикла: `a = 0 while a == 0: print("A")` Выйти из такого цикла можно при помощи оператора `break`

18. Для чего нужен оператор `break` ?

Оператор `break` предназначен для досрочного прерывания работы цикла `while`.

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

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

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

В операционной системе по умолчанию присутствуют стандартных потока вывода на консоль: буферизованный поток `stdout` для вывода данных и информационных сообщений, а также небуферизованный поток `stderr` для вывода сообщений об ошибках.

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

По умолчанию функция `print` использует поток `stdout`. Для того, чтобы использовать поток `stderr` необходимо передать его в параметре `file` функции `print`.

22.Каково назначение функции `exit` ?

Если в процессе выполнения программы произошли ошибки, программа должна передать операционной системе код возврата отличный от нуля. В Python завершить программу и передать операционной системе заданный код возврата можно посредством функции `exit`.