

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №3

«Условные операторы и циклы в языке Python»

Выполнил студент группы ИТС-б-о-21-1

Гайибов Хасан Ммамадиерович

« » _____ 20__ г.

Подпись студента _____

Проверил: Доцент, к.т.н, доцент кафедры
инфокоммуникаций

Воронкин А. В.

Работа защищена с оценкой: _____

(подпись)

Ставрополь, 2022

Лабораторная работа 3

Условные операторы и циклы в языке Python

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

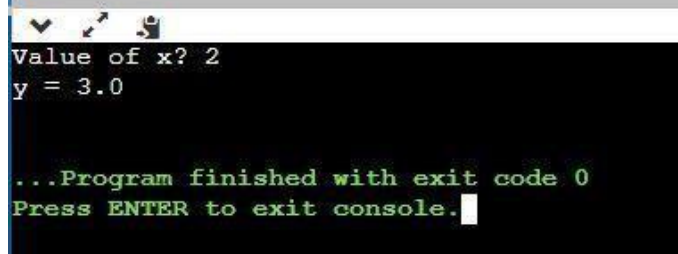
Ход работы:

Создадим общедоступный репозиторий -

Работа с примерами:

Пример 1:

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



```
Value of x? 2
y = 3.0

...Program finished with exit code 0
Press ENTER to exit console.
```

Рис1. Окно вывода для Примера 1.

Пример 2:

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

Введите номер месяца 08
Summer

...Program finished with exit code 0
Press ENTER to exit console.

Рис 2. Окно вывода для Примера 2.

Пример 3:

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

Value of n? 15
Value of x?6
S= 3.528022671951105

...Program finished with exit code 0
Press ENTER to exit console.

Рис3. Окно вывода для Примера 3.

Пример 4:

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

Value of a? 10
x = 3.162277660168379
X = 3.1622776601683795
...Program finished with exit code 0
Press ENTER to exit console.

Рис 4. Окно вывода для Примера 4.

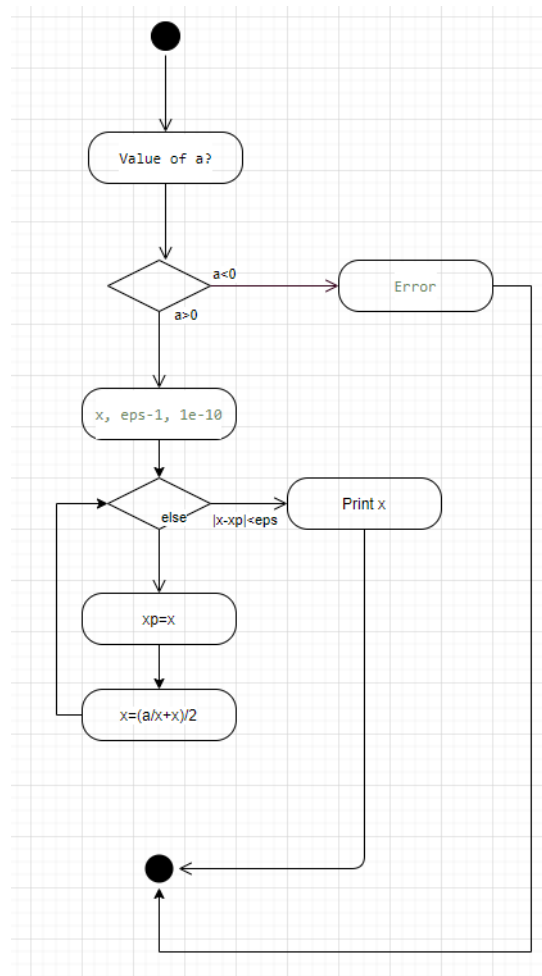


Рис 5. UML диаграмма для примера 4.

Пример 5:

```
1  #!/usr/bin/env python3
2  # -*- кодирование: utf-8 -*-
3  импорт математики
4  импорт sys
5  ЭЙЛЕР = 0,5772156649015328606
6  EPS = 1e-10
7  if __name__ == '__main__':
8      x = float(input("Значение x?"))
9      если x==0:
10         print ("Незаконное значение x", file= sys.stderr)
11         выход(1)
12         a=x
13         S, k = a, 1
14         в то время как math.fabs(a)>EPS:
15             a*= x*k / (k+1)**2
16             S+=a
17             k+=1
18         print(f"Ei({x}) = {EULER + math.log(math.fabs(x))+S}")
```

Value of x? 5
Ei(5.0) = 40.18527535579794
...Program finished with exit code 0
Press ENTER to exit console.

Рис 6. Окно вывода для Примера 5.

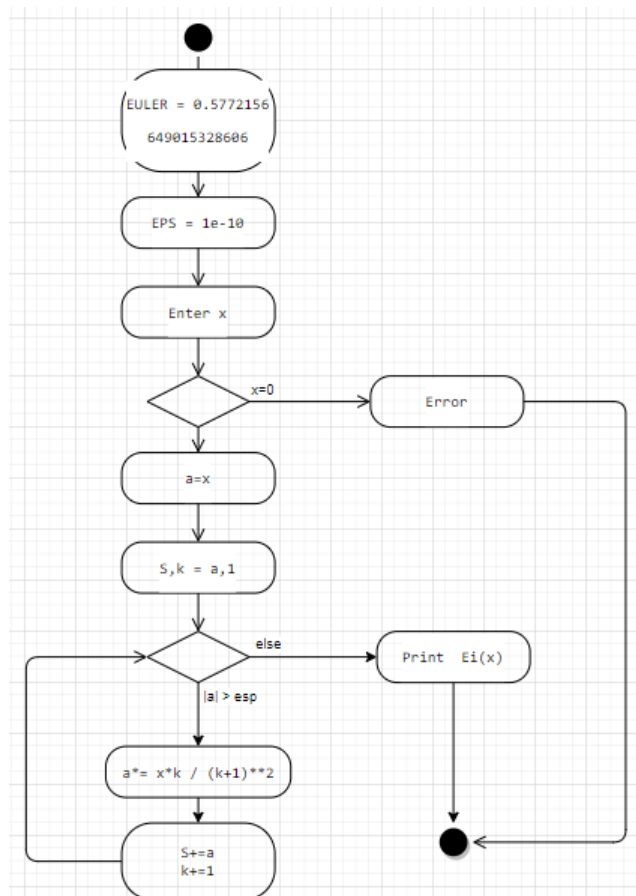


Рис 7. UML диаграмма для примера 5.

Индивидуальные задания:

Вариант 7

Задание 1.

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

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 nb=int(input())
4 m_th={1:'Январь',2:'Февраль',3:'Март',4:'Апрель',5:'Май',6:'Июнь',7:'Июль',8:'Август',9:'Сентябрь',10:'Октябрь',11:'Ноябрь',12:'Декабрь'}
5 if nb<13:
6     print(m_th[nb])
7 else:print("Error")
```




Рис 8. Окно вывода для Задания 1.

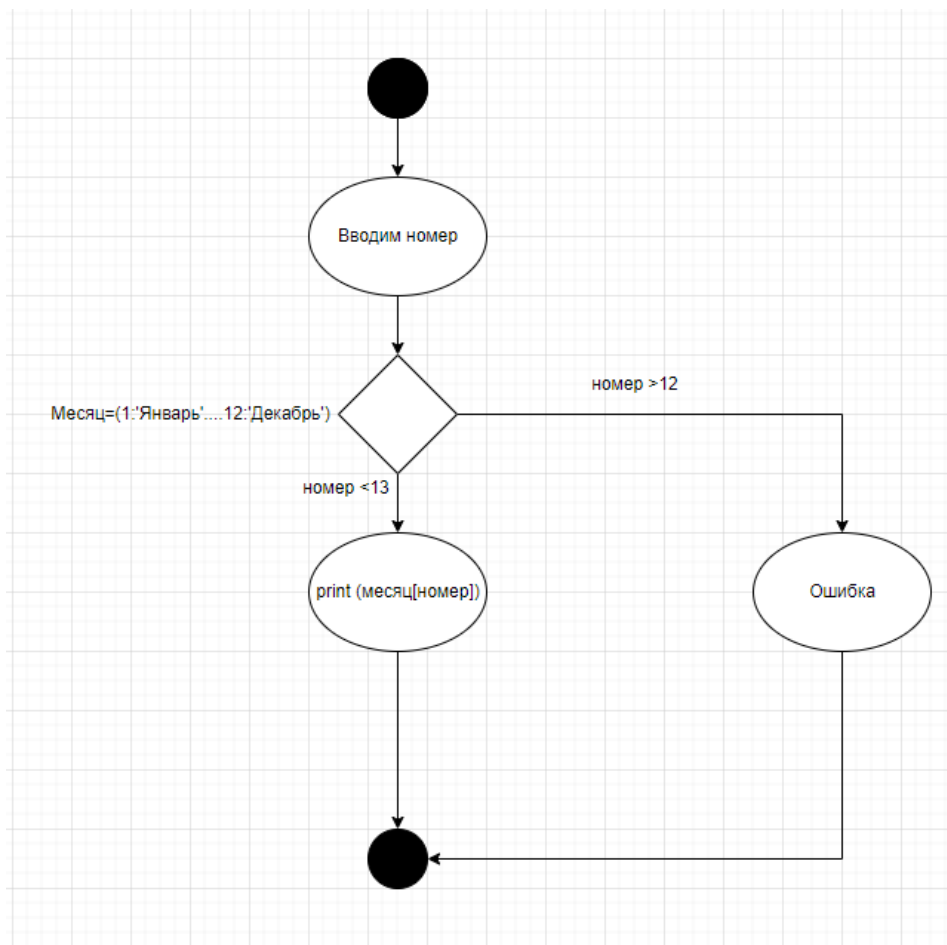


Рис 9. UML диаграмма для Задания 1.

Задание 2.

7. 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  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  a, b, c = map(int, input('Введите a, b, c через пробел: ').split())
4  D = b*b - 4*a*c
5
6  x = []
7
8  if D < 0:
9      print('Действительных корней нет')
10 else:
11     t1 = (-b+D**(1/2))/(2*c)
12     t2 = (-b-D**(1/2))/(2*c)
13
14     if t1 >= 0: x.append(t1**(1/2))
15     if t2 >= 0: x.append(t2**(1/2))
16
17     print('Действительные корни:', *x, sep=' '+chr(177))
```

Введите a, b, c через пробел: 1 2 3
Действительных корней нет

Рис 10. Окно вывода для Задания 2.

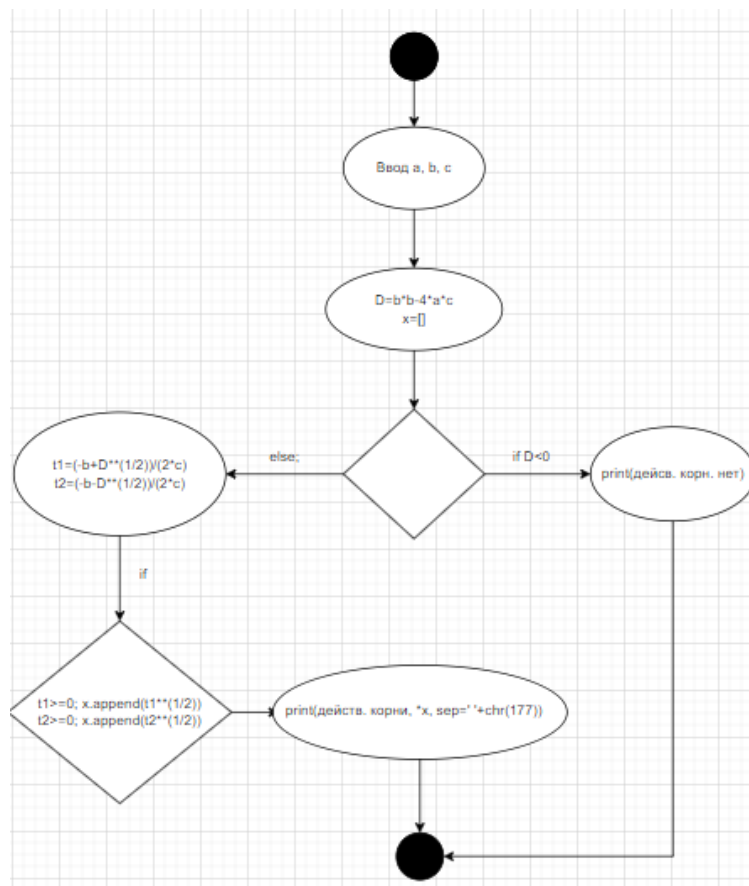


Рис 11. UML диаграмма для Задания 2.

Задание 3.

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

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 i = input('Введите число: ')
4 print('Да' if int(i)%sum(map(int, list(i)))==0 else 'Нет')
```




Рис 12. Окно вывода для Задания 3.

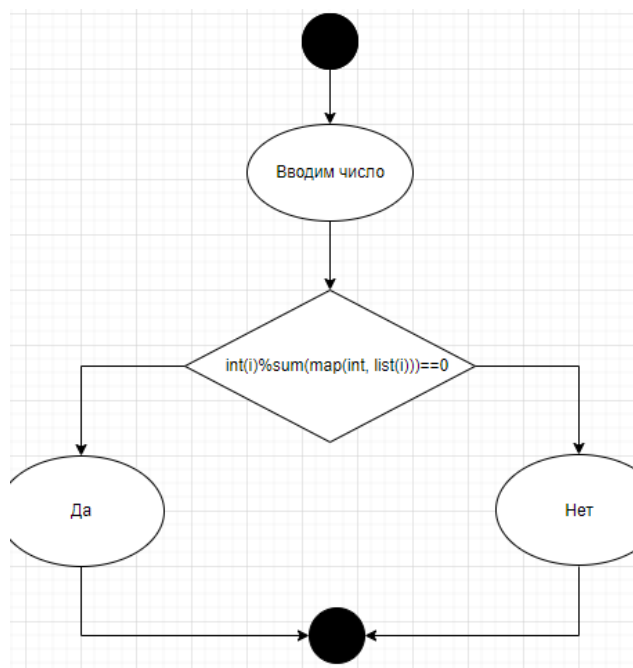


Рис 13. UML диаграмма для Задания 3.

Ответы на вопросы

1. Для чего нужны диаграммы деятельности UML?

Ответ: С помощью UML можно визуализировать, специфицировать, конструировать и документировать артефакты программных систем. UML пригоден для моделирования любых систем: от информационных систем масштаба предприятия до распределенных Web-приложений и даже встроенных систем реального времени.

2. Что такое состояние действия и состояние деятельности?

Ответ: Состояния действия изображаются прямоугольниками с закругленными краями. Внутри такого символа можно записывать произвольное выражение. 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. 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. Какие нотации существуют для обозначения переходов и ветвлений в диаграммах деятельности?

Ответ: 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?

Ответ: 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?

Ответ: В программе разветвляющейся структуры имеется один или несколько условных операторов. Линейный алгоритм представлен в виде прямой структуры, а разветвляющийся расходится на определённые варианты.

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

Ответ: Оператор ветвления if позволяет выполнить определенный набор инструкций в зависимости от некоторого условия.

7. Какие операторы сравнения используются в Python?

Ответ:

Больше (>)

Меньше (<)

Равно(==)

Не равно(!= или \neq)

8. Что называется простым условием? Приведите примеры.

Ответ: Для меня простое условие это наличие логических функций И и ИЛИ.

Логические выражения типа `kByte >= 1023` являются простыми, так как в них выполняется только одна логическая операция.

```
>>> x = 8
```

```
>>> y = 13
```

```
>>> y < 15 and x
```

```
> 8 False
```

9. Что такое составное условие? Приведите примеры.

Ответ: Составное условие – логическое выражение, содержащее несколько простых условий объединенных логическими операциями. Это операции `not`, `and`, `or`. Где `False` - ложь, `True` - истина.

10. Какие логические операторы допускаются при составлении сложных условий?

Ответ: В таких случаях используются специальные операторы, объединяющие два и более простых логических выражения. Широко используются два оператора – так называемые логические И (`and`) и ИЛИ (`or`). Чтобы получить `True` при использовании оператора `and`, необходимо, чтобы результаты обоих простых выражений, которые связывает данный оператор, были истинными. Если хотя бы в одном случае результатом будет `False`, то и все сложное выражение будет ложным. Чтобы получить `True` при использовании оператора `or`, необходимо, чтобы результат хотя бы одного простого выражения, входящего в состав сложного, был истинным. В случае оператора `or` сложное выражение становится ложным лишь тогда, когда ложны оба составляющие его простые выражения.

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

Ответ: нет

12. Какой алгоритм является алгоритмом циклической структуры?

Ответ: Алгоритм циклической структуры –это алгоритм,в котором предусмотрено неоднократное выполнение одной и той же последовательности действий.

13. Типы циклов в языке Python.

Ответ: Цикл WHILE и цикл FOR

14. Назовите назначение и способы применения функции range.

Ответ: 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

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

```
>>>range (15)
```

```
Range (0,15)
```

```
>>>list (range(0,15,2))
```

```
[0,1,2,3...,14,15]
```

Ответ:

16. Могут ли быть циклы вложенными?

Ответ: Да. Вложенные циклы Цикл называется вложенным, если он размещается внутри другого цикла. На первом проходе, внешний цикл вызывает внутренний, который исполняется до своего завершения, после чего управление передается в тело внешнего цикла.

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

Ответ: Бесконечный цикл образуется тогда, когда при выборе пути, выбирается путь с постоянным циклом.

18. Для чего нужен оператор break ?

Ответ: Оператор break предназначен для досрочного прерывания работы цикла while.

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

Ответ: При работе с циклами используются операторы `break` и `continue`.

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

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

Ответ: Рассмотрим каким образом реализован вывод ошибок в приведенной выше программе. 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. По умолчанию функция `print` использует поток `stdout`.

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

Ответ: Для того, чтобы использовать поток `stderr` необходимо передать его в параметре `file` функции `print`. Само же определение потоков `stdout` и `stderr` находится в стандартном пакете Python `sys`. 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

22. Каково назначение функции `exit`

Ответ: 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`.

Вывод: приобрёл навыки программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Освоил операторы языка Python.3 `if`, `while`, `for`, `break`, `continue`, позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры