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:  Сотников Андрей Александрович  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.

Проработка примеров из лабораторной работы:

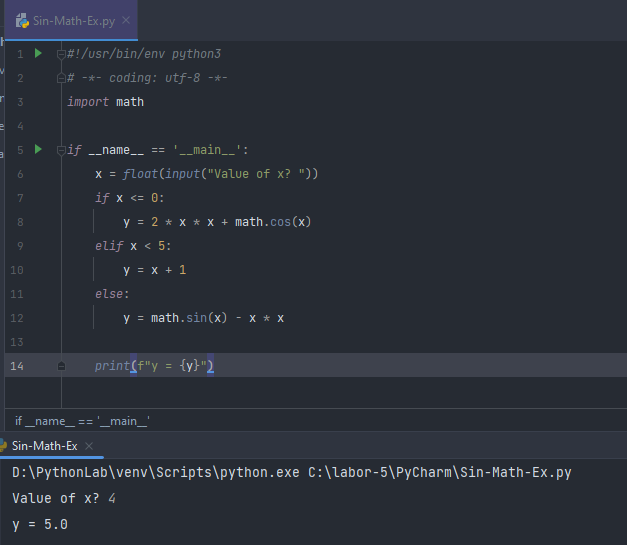


Рисунок 1 – Пример №1

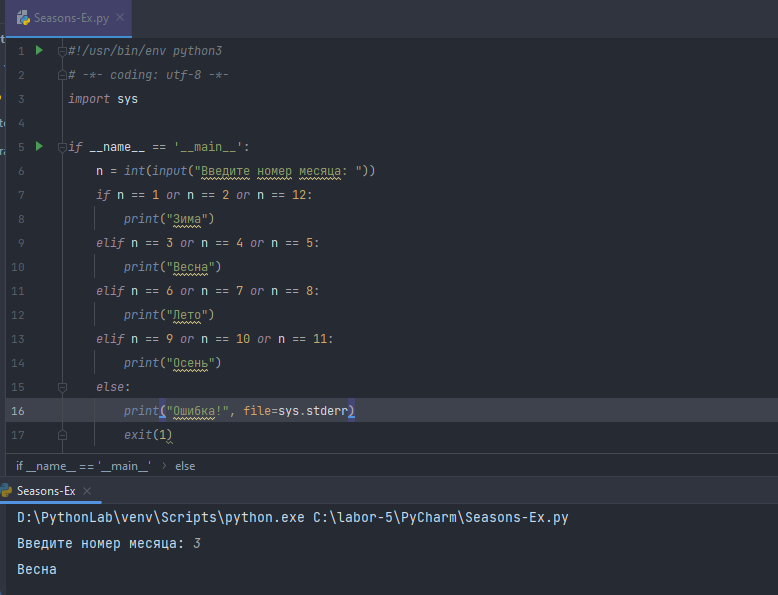


Рисунок 2 – Пример №2

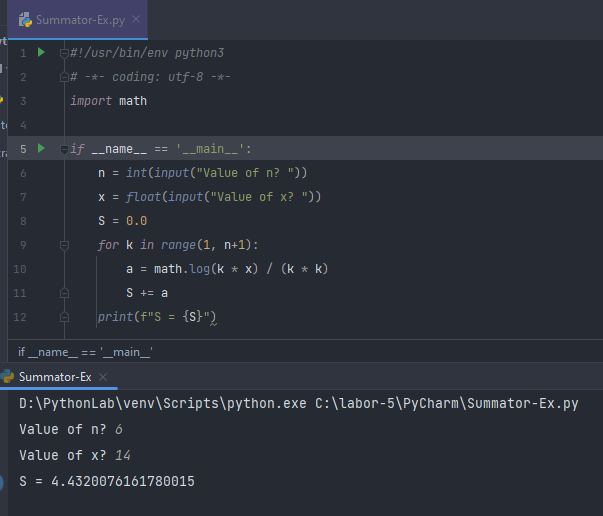


Рисунок 3 – Пример №3

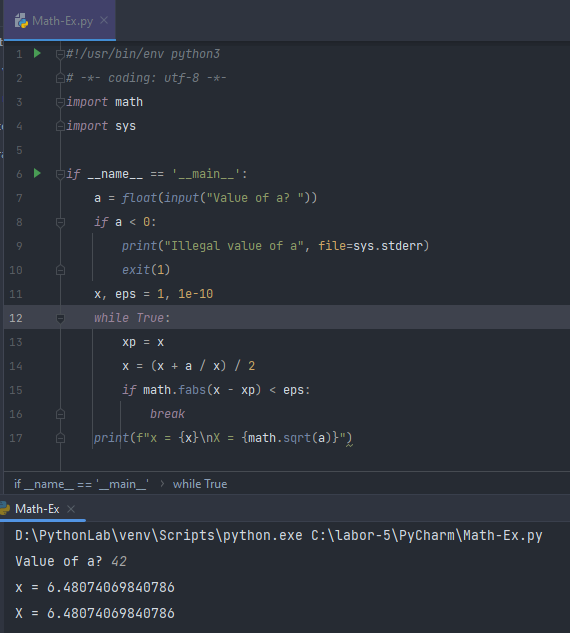


Рисунок 4.1 – Пример №4

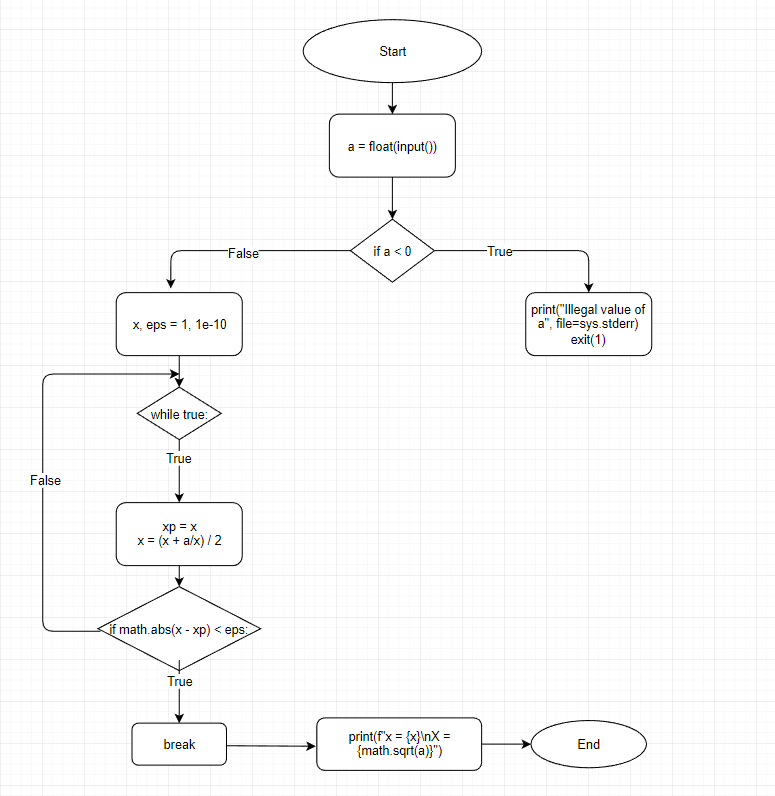


Рисунок 4.2 – UML диаграмма примера №4

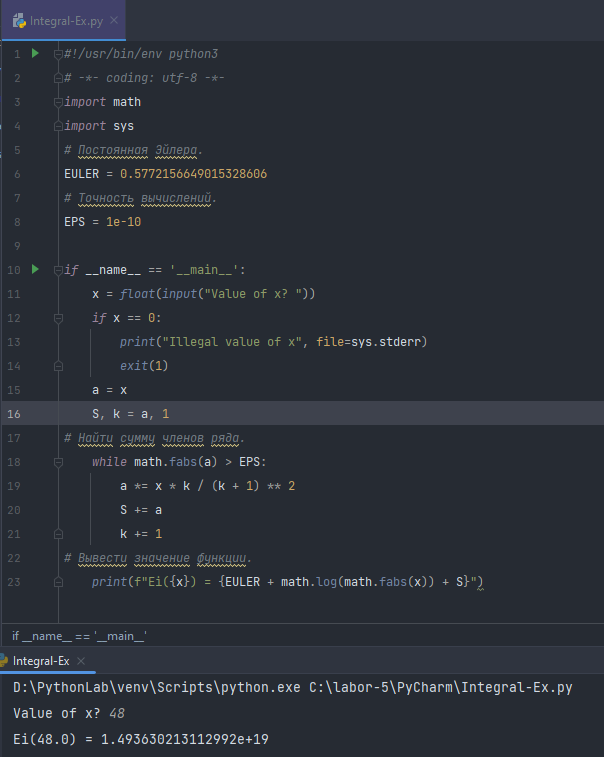


Рисунок 5.1 – Пример №5

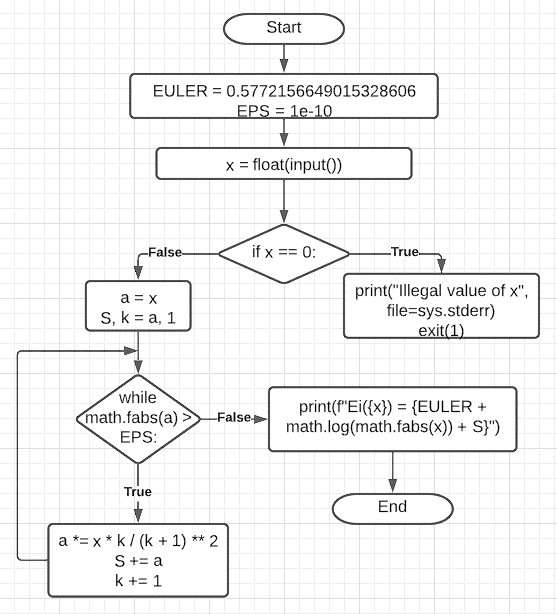


Рисунок 5.2 – UML диаграмма примера №5

Задание№1: Студенты убирают урожай помидоров. При сборе до 50 кг в день работа оплачивается из расчёта 30 коп. за 1 кг; при сборе от 50 до 75 кг в день - 50 коп. за 1 кг; при сборе от 75 до 90 кг в день - 65 коп. за 1 кг; при сборе свыше 90 кг в день - 70 коп. за 1 кг плюс 20 руб. премия. Студент собрал X кг за день. Определить его заработок.

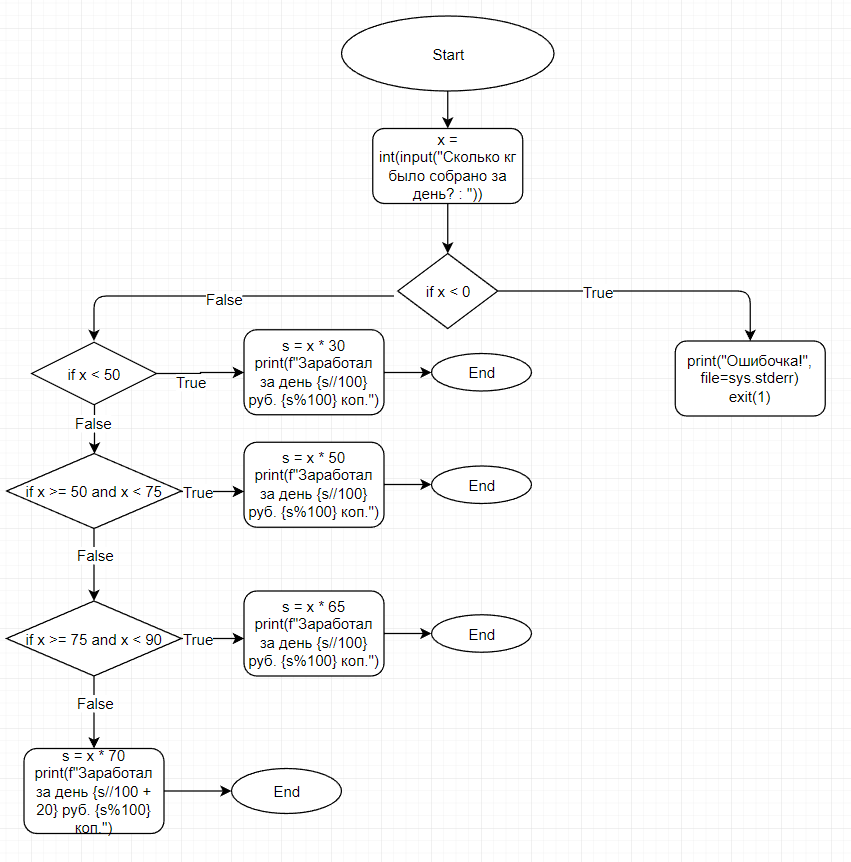


Рисунок 1 – UML диаграмма первого задания

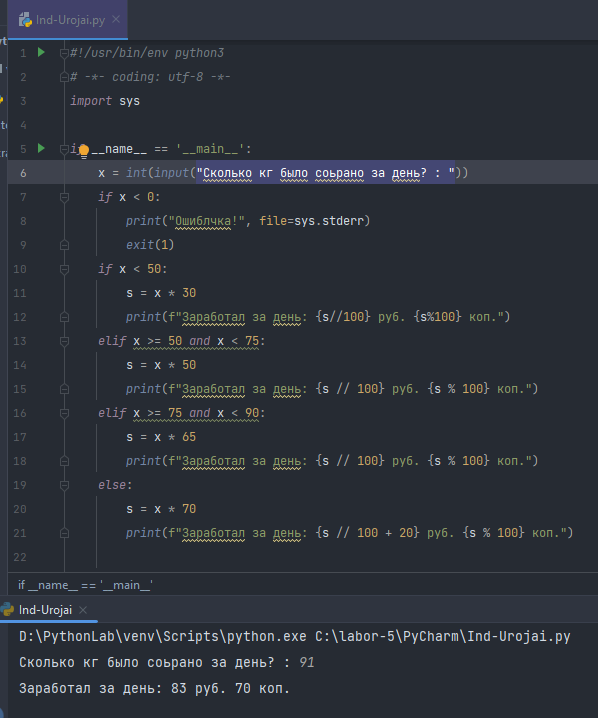


Рисунок 2 – Код и результат работы программы №1

Задание №2: Составить программу, выясняющую делится ли натуральное число x нацело на натуральное число y.

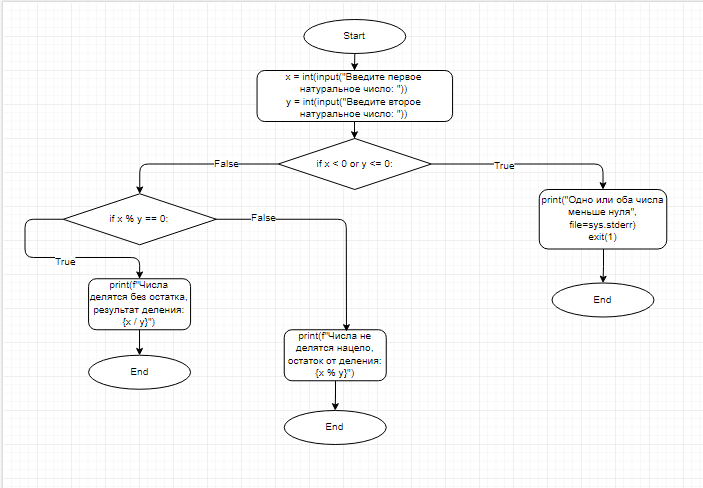


Рисунок 3 – UML диаграмма второго задания

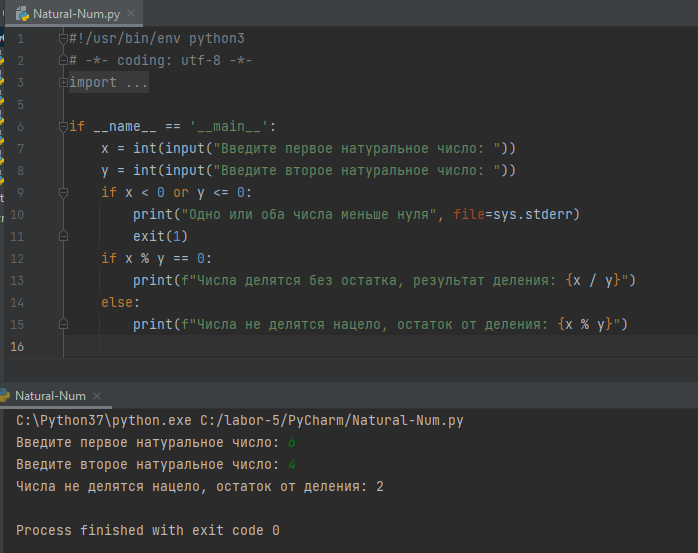


Рисунок 4 – Код и результат работы программы №2

Задание №3: Дано натуральное число . Получить все его натуральные делители.

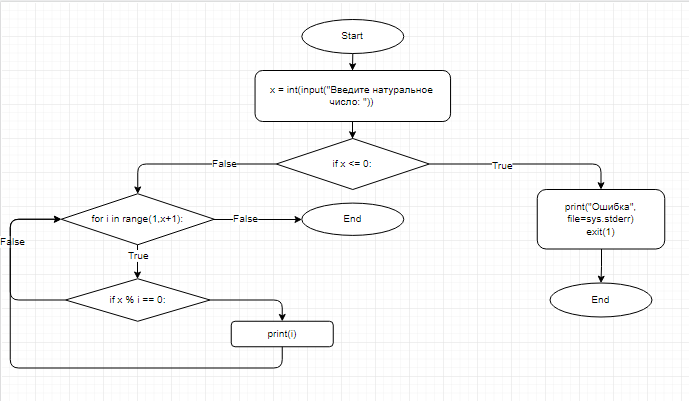


Рисунок 5 – UML диаграмма третьего задания

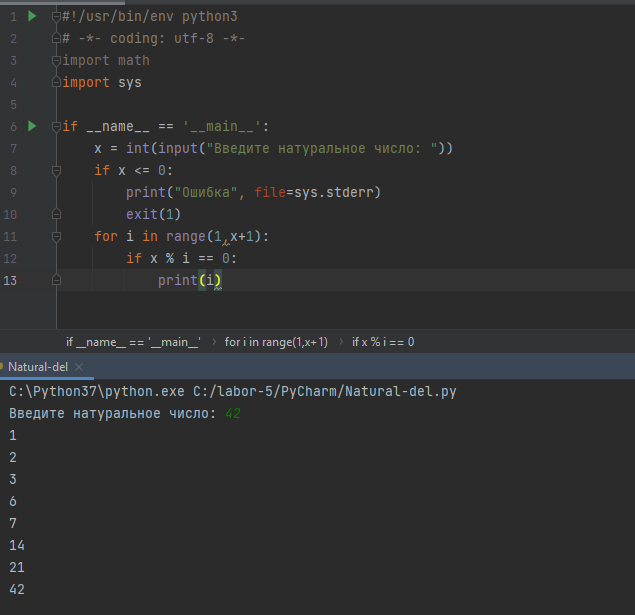


Рисунок 6 – Код программы №3

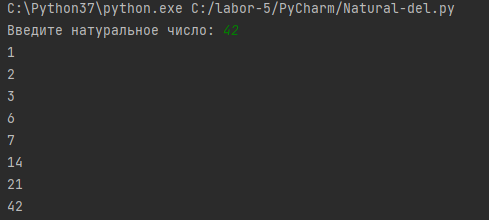


Рисунок 7.1 – Вывод программы №3

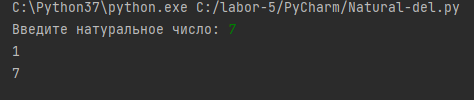


Рисунок 7.2 – Вывод программы №3

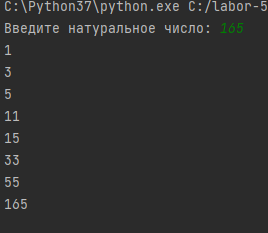


Рисунок 7.3 – Вывод программы №3

**Контрольные вопросы**

# Для чего нужны диаграммы деятельности UML?

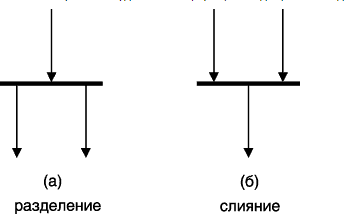
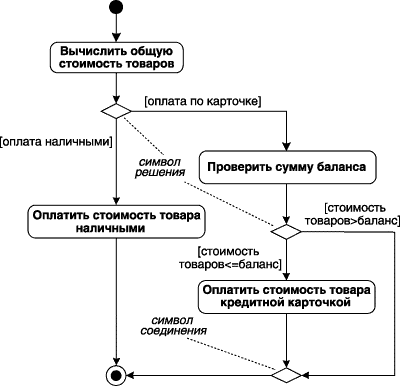
Диаграмма деятельности — это, по существу, блок-схема, которая показывает, как поток управления переходит от одной деятельности к другой, однако, по сравнению с последней, у ней есть явные преимущества: поддержка многопоточности и объектно-ориентированного проектирования.

# 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.

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



1. **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.

# Чем отличается разветвляющийся алгоритм от линейного?

Наличием условных операторов

# Что такое условный оператор? Какие существуют его формы?

Команда, которая выполняется только при каком-либо условии

# Какие операторы сравнения используются в Python?

оператор < , «меньше»;

оператор <= , «меньше или равно; оператор == , «равно»;

оператор != , «не равно»; оператор > , «больше»;

оператор >= , «больше или равно».

# 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

# Что такое составное условие? Приведите примеры.

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

# Какие логические операторы допускаются при составлении сложных условий?

Логическое И, логическое ИЛИ, логическое отрицание

# Может ли оператор ветвления содержать внутри себя другие ветвления?

Да

# 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.

# Типы циклов в языке Python.

В Python есть два вида циклов: for и while

# 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

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

range(0, 15, 2)

# Могул ли быть циклы вложенными?

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

Да могут

# 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

# Для чего нужен оператор break ?

Оператор break предназначен для досрочного прерывания работы цикла while.

# Где употребляется оператор continue и для чего он используется?

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

# 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.

# 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.

# 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.