* 1. Чем отличаются компилируемые и интерпретируемые языки программирования?

Компилируемые языки

Скорость исполнения, благодаря конвертации в машинный код. Управление памятью и контроль процессора. Программу исполняет операционная система

### Интерпретируемые языки

Программу построчно исполнят интерпретаторы, благодаря этому достигается кроссплатформенность. Реализация конструкций, позволяющие динамические изменения на этапе Runtime

* 1. Каковы особенности типизации в языке Python? Как интерпретатор Python работает с памятью?

В python тип данных может определяться на этапе runtime, но не выполняет автоматические неявные преобразования, например нельзя вычесть из строки множество.( Неявная сильная динамическая)

Управление памятью в Python включает в себя частную кучу, содержащую все Объекты и структуры данных Python. Диспетчер памяти имеет различные компоненты, такие как совместное использование, сегментация, preallocation или кеширование.

* 1. Каковы особенности преобразования типов в языке Python?

int() - преобразует строку в целое число

bin() - преобразует десятичное число в двоичный формат

hex() – преобразует в 16-ричный формат

list() преобразует в список

set() преобразует в множество

tuple() преобразует кортеж

str() преобразует строку

* 1. Что общего и отличного в языке Python имеют строки, списки, словари, кортежи, множества?

**Кортежи** (англ. tuple) используется для представления неизменяемой последовательности разнородных объектов.

**Списки** для хранения однородных (и не только) объектов.

**Словарь**  – изменяемая структура данных, предназначенная для хранения элементов вида ключ: значение**.**

**Множество** - "контейнер", содержащий не повторяющиеся элементы в случайном порядке

**Строки** - упорядоченные последовательности символов, используемые для хранения и представления текстовой информации, можно работать со всем, что может быть представлено в текстовой форме.

* 1. Как в Python обявляются и вызываются пользовательские функции?

**Функция** - объект, принимающий аргументы и возвращающий значение, определяется с помощью инструкции def.

* 1. Что такое область видимости функции и правило LEGB?

Область видимости указывает интерпретатору, когда и где вы можете использовать свои переменные, функции.

**L – Local.** Включает в себя имена (идентификатор / переменные), указанные в функции (с использованием def или lambda), а не объявляются с помощью ключевого слова global.  
**E – Enclosing**. Включает в себя имя из локальной области видимости объемлющих функций  (например, с использованием def или lambda).  
**G – Global**. Включает в себя имена, работающих на верхнем уровне модуля или определенных с помощью ключевого слова global.  
**B – Build-in** . Встроенные встроенные функции, такие как print, input, open и т.д

Для поиска имен в **Python** определен следующий порядок:

##### **LOCAL -> ENCLOSING-> GLOBAL -> BUILT-IN**

* 1. Что такое анонимные функции? Когда их удобно использовать?

Анонимная функция(лямбда выражение) в программировании — особый вид функций, которые объявляются в месте использования и не получают уникального идентификатора для доступа к ним.

* 1. Что делают функции map, filter, reduce, zip?

**map** принимает функцию и аргумент составного типа данных. map применяет к каждому элементу списка переданную функцию.

**filter** предлагает элегантный вариант фильтрации элементов последовательности. Принимает в качестве аргументов функцию и последовательность(принимаемая функция должна возвращать true и false)

**reduce** последовательно применяет функцию-аргумент к элементам списка, возвращает единичное значение.

**zip** объединяет в кортежи элементы из последовательностей переданных в качестве аргументов.

* 1. Каковы особенности обработки исключений в языке Python?

Исключения (exceptions) - ещё один тип данных в python. Исключения необходимы для того, чтобы сообщать программисту об ошибках.

В блоке try выполняем инструкцию, которая может породить исключение, а в блоке except перехватываем их.