МДК.05.02 Разработка кода информационных систем

Основы языка программирования Python

1.Что такое динамическая типизация в Python и чем она отличается от статической?

Динамическая типизация — это особенность языка программирования, при которой тип переменной определяется во время выполнения программы, а не заранее при компиляции. Другими словами, переменная может хранить значения разных типов в зависимости от контекста.

Статически типизированные языки (Java, C++) требуют обязательного объявления типа переменной при её создании, и тип остаётся фиксированным на протяжении всего жизненного цикла переменной. Изменять тип переменной нельзя без изменения самого программного кода

2.Какие правила именования переменных существуют в Python?

-Именование начинается с буквы или символа подчёркивания:

-Разрешается использование цифр в имени переменной, но НЕ в качестве первой буквы:

-Зарезервированные слова запрещены: Нельзя использовать названия ключевых слов Python (например, for, while, class, def, import и т.д.) в качестве имён переменных.

-Рекомендуемые соглашения стиля (PEP 8)

-Для обычных переменных рекомендуется использовать строчные буквы с подчёркиванием между словами для улучшения читабельности

-Семантически осмысленные имена

3.Что такое PEP 8 и зачем он нужен?

PEP 8 - это документ с рекомендациями по стилю написания кода на Python. Название расшифровывается как Python Enhancement Proposal (предложение по улучшению Python), а номер 8 обозначает конкретные рекомендации по оформлению кода.

PEP 8 предназначен для повышения читаемости и единообразия написанного кода среди разработчиков. Основные рекомендации включают:

-Стиль отступов (использование пробелов, а не табуляции);

-Ограничение длины строки (до 79 символов);

-Рекомендации по именованию функций, классов, методов и модулей;

-Порядок импорта библиотек и зависимостей;

-Организация пространства вокруг операторов и скобок

4.Как работает множественное присваивание в Python?

Многократное присваивание — это возможность одновременного назначения значений нескольким переменным в одном операторе. Оно существенно упрощает инициализацию переменных и обмен их содержимым

1. Обмен значениями переменных

2. Распаковка коллекций

numbers = [1, 2, 3]

a, b, c = numbers здесь a=1, b=2, c=3

3. Применение звездочки (\*) для сбора оставшихся элементов

Иногда полезно выделить лишь некоторые элементы, а остальные оставить как отдельный массив. Для этого используют звёздочку (\*):

values = [1, 2, 3, 4, 5]

first, second, \*rest = values

В данном примере:

Переменная first примет значение 1,

Переменная second примет значение 2,

Остальные элементы соберутся в списке rest: [3, 4, 5].

5.Чем отличается = от == в Python?

Оператор = - это оператор присваивания, который записывает значение справа в переменную слева

Оператор == - это оператор сравнения, который проверяет равенство значений выражений слева и справа и возвращает логическое значение (True или False)

6.Какие арифметические операторы поддерживает Python?

+ Сложение

- Вычитание

\* Умножение

/ Обычное деление

// Целочисленное деление

% Остаток от деления

\*\* Возведение в степень

7.В чем разница между / и // при делении чисел?

Оператор / выполняет обычное деление, возвращая результат типа float даже если оба операнда целые числа (3.33)

Оператор // выполняет целочисленное деление, отбрасывая дробную часть результата и возвращая целую часть частного (3)

8.Что возвращает оператор %? Приведите пример использования.

Оператор % называется оператором взятия остатка от деления (modulo operator). Он возвращает остаток от деления левого операнда на правый.

result = 9 % 4 # Результат будет 1, так как 9 ÷ 4 даёт частное 2 и остаток 1

9.Как работают логические операторы and, or, not?

Оператор and соединяет два условия и возвращает истину (True), только если оба условия выполняются. Если хотя бы одно из условий ложно, возвращается ложь (False).

Оператор or возвращает истину, если хотя бы одно из условий выполняется. Если оба условия ложны, возвращается ложь.

Оператор not инвертирует логическое значение условия. То есть, если условие было истинным, оно становится ложным, и наоборот.

10.Что такое операторы принадлежности in и not in?

Оператор in проверяет, содержится ли элемент в заданной последовательности. Если да, возвращает True, иначе — False.

Оператор not in инверсия оператора in. Проверяет отсутствие элемента в последовательности. Если элемент отсутствует, возвращает True, иначе — False

11.Какие числовые типы данных есть в Python?

1. Тип int (целые числа)

Представляют обычные целые числа без ограничений на размер

2. Тип float (вещественные числа)

Используются для представления чисел с плавающей точкой (десятичные дроби).

3. Тип complex (комплексные числа)

Представляются в виде суммы действительного и мнимого компонентов.

4. Тип bool

Хранят логические значения True и False, наследуют класс int, где True эквивалентно 1, а False — 0

12.Чем отличается int от float?

Тип int представляет целые числа, такие как 5, -3. Они не содержат десятичной точки и обладают неограниченной точностью (в разумных пределах памяти компьютера).

Тип float представляет вещественные числа с плавающей запятой, такие как 3.14, -0.5. Такие числа ограничены в точности вычислений, поскольку хранятся в двоичном формате IEEE 754.

13.Как записать очень большое число для удобства чтения?

В Python большие числа удобно читать благодаря возможности вставлять символ подчёркивания \_ между цифрами. Эти подчёркивания игнорируются интерпретатором, и число воспринимается как единое целое. (1\_000\_000\_000)

14.Что такое комплексные числа и как их создать в Python?

Комплексные числа состоят из действительной и мнимой частей. Они представлены встроенным типом данных complex.

Создание комплексного числа:

Через конструктор класса complex():

Использованием синтаксиса мнимых чисел (литерал j):

z = 3 + 4j # Тоже самое, создаст 3+4j

Здесь:

Действительная часть равна 3,

Мнимая часть равна 4.

15.Какие три способа создания строк вы знаете?

-Используя одинарные кавычки

-Используя двойные кавычки

-Используя тройные кавычки (для многострочных строк)

16.Что такое f-строки и чем они лучше других способов форматирования?

Фундаментальное отличие f-строк от старых методов форматирования заключается в следующем:

-Читаемость: F-строки делают код проще и легче для восприятия, так как шаблон непосредственно показывает, какое значение подставляется в какую позицию.

-Производительность: Они быстрее обрабатываются, поскольку выполняются почти напрямую, минуя лишние этапы обработки.

-Поддержка выражений: Внутри фигурных скобок можно помещать полноценные выражения Python, что открывает дополнительные возможности для динамического построения строк.

17.Какие методы для работы со строками вы помните?

len(): возвращает длину строки, то есть количество символов в ней.

upper() и lower(): преобразуют строку в верхний и нижний регистр соответственно.

strip(): удаляет пробелы в начале и конце строки.

split(): разбивает строку на список подстрок по заданному разделителю.

lower(): Переводит всю строку в нижний регистр.

upper(): Переводит всю строку в верхний регистр.

capitalize(): Первая буква строки становится заглавной, остальное — строчными.

18.Как разбить строку на список слов?

Метод split() разделяет строку на отдельные слова, используя заданный разделитель. По умолчанию разделителем служит пробел.

19.Что такое срезы строк и как они работают?

Срезы (slices) — это мощный инструмент Python для выделения фрагментов из последовательностей, таких как строки, списки, кортежи и другие подобные структуры данных. Они позволяют получать подмножество элементов последовательности удобным и эффективным способом.

Как работают: срез не изменяет исходный объект, а создаёт новый, содержащий выбранные элементы. При этом исходный объект остаётся неизменным. взятие одного символа строки, а именно, S[i] — это срез, состоящий из одного символа, который имеет номер i, при этом считая, что нумерация начинается с числа 0. То есть если S = ‘Hello’, то S[0]=='H', S[1]=='e', S[2]=='l', S[3]=='l', S[4]=='o'.

20.Что такое список и чем он отличается от строки?

Список — это упорядоченный изменяемый контейнер объектов произвольных типов. Элементы списка могут храниться в одной структуре и обращаться к ним по индексу

Ключевые отличия списка от строки:

-Список изменяется (можно добавлять, удалять, изменять элементы).

-Списки могут содержать элементы любых типов данных.

-Строка неизменна, а изменение строки фактически создаёт новую строку

21.Какие методы для добавления элементов в список вы знаете?

-append: добавляет элемент в конец списка.

-extend: расширяет список элементами другого итерируемого объекта.

-insert: вставляет элемент в указанное положение.

22.В чем разница между append() и extend()?

Основное различие состоит в том, что append() добавляет весь передаваемый объект целиком [1, 2, 3, [4, 5]], а extend() распаковывает этот объект и добавляет его элементы индивидуально [1, 2, 3, 4, 5]