Вопросы в экзамене будут по темам :

Алгоритм и программа:

Алгоритм — это точное и полное описание последовательности действий над заданными объектами, позволяющее получить конечный результат. Основные требования к алгоритмам: дискретность (последовательность простых этапов), определённость (шаги должны допускать однозначную трактовку), массовость (решение не только для конкретного набора значений, а для целого класса задач) и результативность (конкретный результат). 2

Программа — это запись алгоритма на языке программирования. 2 Основная задача на ЕГЭ по информатике — «научить» код забирать данные, обрабатывать их и выдавать число. 1

Некоторые советы, как превратить решение задачи в код для ЕГЭ по информатике:

Написать последовательные шаги. Нужно понять, что требуется для решения задачи, а затем перевести всё в алгоритм — последовательность действий. 1

Схематично изобразить решение. Сначала изображают абстракции, а потом преображают их в последовательность шагов. 1

Протестировать решение. После того как написан скелет программы, необходимо протестировать код, например, с помощью трассировки (пошаговое выполнение программы).

Компиляторы и интерпретаторы:

Компилятор — это компьютерная программа, которая переводит компьютерный код с одного языка программирования на другой. 2 Он читает всю программу целиком, делает её перевод и создаёт законченный вариант программы на машинном языке, который затем и выполняется. Результат работы компилятора — бинарный исполняемый файл. 1

Интерпретатор — это компьютерная программа, которая преобразует каждый программный оператор высокого уровня в машинный код. 2 Он переводит и выполняет программу строка за строкой. Программа, обрабатываемая интерпретатором, должна заново переводиться на машинный язык при каждом очередном запуске программы. 1

Откомпилированные программы работают быстрее, но интерпретируемые проще исправлять и изменять

Парадигмы программирования:

Парадигма программирования — это совокупность понятий, идей и правил, которые определяют подход к написанию кода. Она нужна для того, чтобы сделать текст программы структурированным и понятным как для других разработчиков, так и для компьютера. 2

Некоторые основные парадигмы программирования:

Императивная. 14 Предполагает запись программы в виде набора инструкций, которые нужно последовательно выполнить для получения результата. Отвечает на вопрос о том, как следует решать проблему. 1

Декларативная. 14 В ней указывается, что задано и каким должен быть результат. Отвечает на вопрос о том, что нужно сделать для решения проблемы. 1

Процедурная. В этой парадигме программа представляет собой набор процедур, одна из которых является главной и из неё производятся обращения к прочим. 1

Структурная. В её основе лежит представление программы как совокупности процедурных блоков, которая имеет чёткую иерархию. Это позволяет лучше видеть всю структуру связей между отдельными блоками. 1

Объектно-ориентированная. 14 Предполагает взгляд на программу как на совокупность отдельных объектов, определённым образом взаимодействующих друг с другом. 1

Функциональная. Трактует реализацию алгоритма как процесс нахождения значений некоторых математических функций.

Языки программирования и их применение:

Pascal. Универсальный язык со строгой структурой и простым синтаксисом. Подходит для изучения базовых алгоритмов и структур данных. 1

C++. Позволяет создавать программные обеспечения для разных целей — от компьютерных игр до целых операционных систем. Известен высокой производительностью и эффективностью, работает с большими объёмами данных. 1

C#. Язык создан для проектов Windows, но сейчас на нём пишут приложения, игры, нейросети. За счёт интеграции с .NET позволяет разрабатывать приложения под Windows, использовать встроенные библиотеки для упрощения задач. 1

Java. Используется для работы с анализом данных, написания приложений для Android, научных программ. На нём создаются корпоративные софты, различные встроенные системы. 1

Python. 1 Язык известен простотой и лаконичностью, что делает его идеальным выбором для начинающих программистов. 5 Используется во многих областях — от веб-разработки до машинного обучения.

Система контроля версий:

Система контроля версий позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение.

Типы данных в Python:

**boolean** — логическое значение (true или false); [1](https://prooge.ru/zagruzki/Python-min.pdf)

**int** — целое число, для хранения которого выделяется 4 байта памяти компьютера; [1](https://prooge.ru/zagruzki/Python-min.pdf)

**float** — число с плавающей точкой, для хранения которого выделяется 8 байт памяти компьютера; [1](https://prooge.ru/zagruzki/Python-min.pdf)

**str** — строки; [1](https://prooge.ru/zagruzki/Python-min.pdf)

**list** — списки. [1](https://prooge.ru/zagruzki/Python-min.pdf)

Также к основным типам данных в Python относятся:

**None** — экземпляр типа объекта NoneType и особая переменная, которая не имеет целевого значения; [4](https://habr.com/ru/articles/825806/)

**complex** — комплексные числа; [4](https://habr.com/ru/articles/825806/)

**tuple** — неизменяемые упорядоченные коллекции элементов (кортежи); [4](https://habr.com/ru/articles/825806/)

**bytes** — байтовые последовательности, которые используются для работы с бинарными файлами; [4](https://habr.com/ru/articles/825806/)

**frozenset** — функция, которая возвращает неизменяемый объект frozenset, инициализированный элементами из заданного итерируемого объекта; [4](https://habr.com/ru/articles/825806/)

**list** — изменяемые упорядоченные коллекции элементов (списки); [4](https://habr.com/ru/articles/825806/)

**dict** — ассоциативный массив, пары «ключ-значение», где каждый ключ является уникальным; [4](https://habr.com/ru/articles/825806/)

**set** — неупорядоченная и неиндексированная коллекция уникальных элементов.

Синтаксис инструкции if в языке программирования C: **if (условие) { // if body // Инструкции для выполнения, если условие истинно }**.

**Цикл while** в языке Си — это **цикл с предусловием**. [5](https://prog-cpp.ru/c-cycles/) Он сразу проверяет истинность некоторого условия, и если условие истинно, то есть не равно 0, то код цикла выполняется. [1](https://metanit.com/c/tutorial/2.12.php)

**Цикл for** — **параметрический цикл** (цикл с заданным числом повторений). [5](https://prog-cpp.ru/c-cycles/) Он используется для многократного выполнения набора инструкций до тех пор, пока не будет выполнено определённое условие. [2](https://tr-page.yandex.ru/translate?lang=en-ru&url=https%3A%2F%2Fwww.studytonight.com%2Fc%2Floops-in-c.php)

**Цикл for** состоит из трёх частей: [1](https://metanit.com/c/tutorial/2.12.php)

**Инициализация**. Выполняется один раз при начале выполнения цикла и представляет установку начальных условий. [1](https://metanit.com/c/tutorial/2.12.php)

**Условие**. Условие, при соблюдении которого выполняется цикл. Зачастую в качестве условия используется операция сравнения. [1](https://metanit.com/c/tutorial/2.12.php)

**Приращение**. Задает изменение параметров цикла. Обычно здесь происходит увеличение счётчиков цикла. [1](https://metanit.com/c/tutorial/2.12.php)

**Словарь (dict) в Python** — это структура, которая хранит данные в формате пар ключ-значение. Для создания словаря необходимо передать последовательность элементов внутри фигурных скобок {}, разделив их запятыми (,). Каждый элемент имеет ключ и значение, выраженное парой «ключ: значение». Значения могут быть представлять собой любые типы данных и повторяться, но ключи обязаны быть уникальными.

1. **Кортежи (tuple)**
2. **Индексы и срезы**
3. **Индексы**. [1](https://dzen.ru/a/Y_7WRoyZygYzdWAT)[3](https://pythoshka.com/p1386.html) Нумерация элементов начинается с нуля. В Python поддерживаются отрицательные индексы, при этом нумерация идёт с конца. [3](https://pythoshka.com/p1386.html) Например, чтобы получить последний элемент списка, в квадратных скобках указывают -1, индекс предпоследнего элемента — -2, третьего с конца элемента — -3 и так далее. [1](https://dzen.ru/a/Y_7WRoyZygYzdWAT)
4. **Срезы**. [1](https://dzen.ru/a/Y_7WRoyZygYzdWAT)[3](https://pythoshka.com/p1386.html) Позволяют получить не один, а сразу несколько идущих подряд элементов списка. Чтобы сделать срез, в квадратных скобках через знак двоеточия указывают начало и конец желаемого диапазона. Начало среза — это индекс первого входящего в желаемый диапазон элемента, а конец среза — индекс последнего элемента +1. Результатом выполнения среза является новый список. Если указанный конец среза превышает количество индексов в списке, ошибки не будет, лишние значения просто отбрасываются.

* **Неизменяемость** (нельзя добавить или удалить новые элементы). [5](https://smartiqa.ru/python-workbook/tuple)
* **Упорядоченность** (всегда известно, на какой позиции какой элемент находится). [5](https://smartiqa.ru/python-workbook/tuple)
* **Длина** однажды созданного кортежа не меняется никогда. [5](https://smartiqa.ru/python-workbook/tuple)
* **Чаще всего используются** для разнородных данных (представленных строками, числами, списками, множествами и т.п.). [5](https://smartiqa.ru/python-workbook/tuple)

**Некоторые преимущества кортежей**:

* **Защита от изменений** (кортеж защищён от намеренных и случайных изменений). [2](https://pythoshka.com/p1392.html)
* **Меньший размер** по сравнению со списками. [2](https://pythoshka.com/p1392.html)
* **Возможность использовать кортежи в качестве ключей словаря**.

Функции в Python

**Для описания функции в Python необходимо** **указать имя функции, описать её аргументы и программный код**. Описание начинается с зарезервированной команды

def

. После указывается название функции и перечисляются аргументы (после имени функции в круглых скобках). [1](https://prooge.ru/zagruzki/Python-min.pdf)

**Тело функции часто содержит инструкцию**

**return**

. Она завершает работу функции и передаёт результат вызывающей программе. Команда

return

 является необязательной, если её нет, то работа функции завершается, когда поток управления достигает конца тела функции. [1](https://prooge.ru/zagruzki/Python-min.pdf)

**Пример решения задачи на ЕГЭ с использованием функций в Python**: [4](https://dzen.ru/a/ZSmvAqP3eQVtt8y-)

def f(n):

if n == 1:

return 1

return f(n - 1) \* n

print(f(5))

**Рекурсивная функция** — это функция, которая в процессе выполнения вызывает саму себя.

**Конструкция try — except** используется для обработки исключений в Python. [2](https://pythonchik.ru/osnovy/python-try-except)[3](https://pythoshka.com/p1416.html)

**Принцип работы**: операция, которая может вызвать исключение, помещается внутрь блока try, а код, который должен быть выполнен при возникновении ошибки, находится внутри блока except. [2](https://pythonchik.ru/osnovy/python-try-except)

**Пример**: как можно обработать ошибку деления на ноль:

try:

a = 7 / 0

except:

print('Ошибка! Деление на 0')

**Tkinter** — это **кроссплатформенный графический интерфейс Python, позволяющий работать с библиотекой Tk**.

**Что такое JSON**? Это простой формат данных, используемый в качестве средства связи между различными системами

**Для взаимодействия с образовательным порталом «СДАМ ГИА» можно использовать Python-модуль SdamGIA Api**

* **itertools**. Содержит функции для комбинаторики. [2](https://300.ya.ru/v_w3zMeET8)
* **turtle**. Можно использовать для решения задач. [2](https://300.ya.ru/v_w3zMeET8)
* **math**. В библиотеке есть функции для округления чисел, вычисления факториала, нахождения наибольшего общего делителя и наименьшего общего кратного. [2](https://300.ya.ru/v_w3zMeET8)
* **functools**. Основные функции: product, permut, combinations. [2](https://300.ya.ru/v_w3zMeET8)
* **string**. Библиотека предназначена для работы со строками.

r = Режим только для чтения. Указатель стоит в начале файла. rb =Режим для чтения в двоичном формате. Указатель стоит в начале файла. r+ = Режим для чтения и записи. Указатель стоит в начале файла. rb+ =Режим для чтения и записи в двоичном формате. Указатель стоит в начале файла. w =Режим только для записи. Указатель стоит в начале файла. Создает файл с именем имя\_файла, если такового не существует. wb =Режим для записи в двоичном формате. Указатель стоит в начале файла. Создает файл с именем имя\_файла, если такового не существует. w+ =Режим для чтения и записи. Указатель стоит в начале файла. Создает файл с именем имя\_файла, есл такового не существует. wb+ =Режим для чтения и записи в двоичном формате. Указатель стоит в начале файла. Создает файл с имимя\_файла, если такового не существует. a Режим для добавления информации в файл. Указатель стоит в конце файла. Создает файл с именем имя\_файла, если такового не существует. ab =Режим для добавления в двоичном формате. Указатель стоит в конце файла. Создает файл с именемимя\_файла, если такового не существует. a+ =Режим для добавления и чтения. Указатель стоит в конце файла. Создает файл с именем имя\_файлатакового не существует. ab+ =Режим для добавления и чтения в двоичном формате. Указатель стоит в конце файла. Создает файл именем имя\_файла, если такового не существует.