Ключевые слова C# и их назначения

### 1. 1. ****Типы данных****

* **bool**: Логический тип (true/false).
* **byte**: Целое число без знака (0 до 255).
* **sbyte**: Целое число со знаком (-128 до 127).
* **short**: Целое число со знаком (-32,768 до 32,767).
* **ushort**: Целое число без знака (0 до 65,535).
* **int**: Целое число со знаком (-2,147,483,648 до 2,147,483,647).
* **uint**: Целое число без знака (0 до 4,294,967,295).
* **long**: Целое число со знаком (-9,223,372,036,854,775,808 до 9,223,372,036,854,775,807).
* **ulong**: Целое число без знака (0 до 18,446,744,073,709,551,615).
* **float**: Число с плавающей точкой одинарной точности.
* **double**: Число с плавающей точкой двойной точности.
* **decimal**: Число с высокой точностью для финансовых расчетов.
* **char**: Символ Unicode.
* **string**: Строка текста.
* **object**: Базовый тип для всех типов.
* **dynamic**: Тип, определяемый во время выполнения.
* **var**: Тип, выводимый компилятором.

### 2. ****Условные конструкции****

* **if**: Проверяет условие.
* **else**: Выполняется, если условие в if ложно.
* **else if**: Дополнительное условие.
* **switch**: Выбор на основе значения.
* **case**: Вариант в switch.
* **default**: Выполняется, если ни один case не подошел.
* **when**: Условие в switch для фильтрации.

### 3. ****Циклы****

* **for**: Цикл с заданным количеством итераций.
* **foreach**: Перебор элементов коллекции.
* **while**: Цикл с условием.
* **do**: Цикл с постусловием.

### 4. ****Управление потоком выполнения****

* **break**: Прерывает цикл или switch.
* **continue**: Переходит к следующей итерации цикла.
* **return**: Возвращает значение из метода.
* **goto**: Переход к метке.
* **throw**: Выбрасывает исключение.
* **try**: Блок для обработки исключений.
* **catch**: Обрабатывает исключение.
* **finally**: Блок, выполняющийся после try и catch.

### 5. ****Классы, структуры и интерфейсы****

* **class**: Определяет класс.
* **struct**: Определяет структуру.
* **interface**: Определяет интерфейс.
* **enum**: Определяет перечисление.
* **partial**: Разделяет определение класса или метода на несколько файлов.
* **abstract**: Определяет абстрактный класс или метод.
* **sealed**: Запрещает наследование.
* **static**: Определяет статический член.
* **virtual**: Позволяет переопределение метода.
* **override**: Переопределяет метод базового класса.
* **new**: Создает экземпляр или скрывает член базового класса.
* **this**: Ссылка на текущий экземпляр.
* **base**: Ссылка на базовый класс.
* **operator**: Определяет пользовательский оператор.

### 6. ****Модификаторы доступа****

* **public**: Доступно везде.
* **private**: Доступно только внутри класса.
* **protected**: Доступно в классе и производных.
* **internal**: Доступно в пределах сборки.
* **protected internal**: Доступно в сборке и производных классах.

### 7. ****Асинхронность****

* **async**: Определяет асинхронный метод.
* **await**: Ожидает завершения асинхронной операции.

### 8. ****Делегаты и события****

* **delegate**: Определяет тип делегата.
* **event**: Определяет событие.

### 9. ****Преобразования типов****

* **as**: Безопасное приведение типов.
* **is**: Проверяет тип объекта.
* **explicit**: Явное преобразование типов.
* **implicit**: Неявное преобразование типов.

### 10. ****LINQ (Language Integrated Query)****

* **from**: Начинает запрос.
* **where**: Фильтрует данные.
* **select**: Выбирает данные.
* **group**: Группирует данные.
* **orderby**: Сортирует данные.
* **join**: Объединяет данные.
* **let**: Создает переменную в запросе.
* **into**: Продолжает запрос после группировки.
* **ascending**: Сортировка по возрастанию.
* **descending**: Сортировка по убыванию.

### 11. ****Управление памятью и небезопасный код****

* **unsafe**: Разрешает небезопасный код.
* **fixed**: Фиксирует объект в памяти.
* **stackalloc**: Выделяет память в стеке.
* **checked**: Включает проверку переполнения.
* **unchecked**: Отключает проверку переполнения.

### 12. ****Директивы препроцессора****

* **#define**: Определяет символ.
* **#undef**: Удаляет символ.
* **#if**: Условная компиляция.
* **#else**: Альтернатива в условной компиляции.
* **#elif**: Дополнительное условие.
* **#endif**: Завершает условную компиляцию.

### 13. ****Другие ключевые слова****

* **const**: Определяет константу.
* **readonly**: Определяет поле, которое можно задать только один раз.
* **volatile**: Указывает, что поле может изменяться в многопоточной среде.
* **extern**: Указывает, что метод или переменная определены вне программы.
* **namespace**: Определяет пространство имен.
* **using**: Упрощает использование пространств имен или управляет ресурсами.
* **typeof**: Возвращает тип объекта.
* **nameof**: Возвращает имя переменной или типа.
* **yield**: Используется в итераторах.
* **params**: Позволяет передавать переменное количество аргументов.
* **ref**: Передача параметра по ссылке.
* **out**: Возврат значения через параметр.
* **sizeof**: Возвращает размер типа в байтах.
* **true**: Логическое значение "истина".
* **false**: Логическое значение "ложь".

### Пропущенные ключевые слова

* **value**: Используется в сеттерах свойств для доступа к передаваемому значению.
* **global**: Обращение к глобальному пространству имен.
* **init**: Определяет инициализатор свойства (C# 9.0+).
* **record**: Определяет тип записи (C# 9.0+).
* **with**: Используется для копирования записей с изменениями (C# 9.0+).
* **and**, **or**, **not**: Используются в шаблонах (C# 9.0+).