1. **Расписать в текстовом документе зачем нужны интерфейсы и их отличие от классов**

Отличие класса от интерфейса состоит в том, что класс может наследоваться лишь от одного класса, но от множества интерфейсов, дабы расширить свой функционал.

Интерфейсы описывают какие-то общие наборы – методы, свойства.. А в классе уже прописываются конкретные моменты. Если очень грубо говоря, мы создаем интерфейс Iживотное.

interface IЖивотное

{

string Type { get;set;};

string Sex { get;set;};

int Age { get;set;};

void Передвижение();

}

Это обобщенная модель, которая содержит основные параметры животного

Далее мы создаем класс, который уже реализует эту модель, указывая конкретные параметры

class кот : IЖивотное

{

public string Type {}

public string Sex {}

public int Age {}

void Передвижение()

{

ходит на 4х лапах

Debug.Log("Мяу!");

}

}

Мы также можем создать class корова : IЖивотное. И он будет содержать такие же основные параметры, но тип, возраст и т.д будут иными. Также мы можем наследовать и другие интерфейсы.

class корова : IЖивотное, IЛюблюЛуг, IЖвачкаlivesMatter и тд..

И еще в классе мы помимо параметров интерфейсов можем добавлять свои уникальные параметры. Например к классу корова мы можем добавить метод

void ПрезирающийВзгляд()

{

} и т.д.

В интерфейсе можно прописать реализацию чего-либо, которая должна быть по умолчанию. Соответственно в классе мы уже будем реализовывать все методы и свойства интерфейсов, кроме тех, которые идут по умолчанию. И, как правило, то что идет по умолчанию имеет модификатор доступа privat (но это не точно)

Класс же может наследоваться лишь от одного класса. Еще, если мы добавляем в интерфейсе что-то (тот же метод), то это нужно обязательно реализовать в классе.

В общем, интерфейсы классная штука для наследования, когда нужно сделать много похожих предметов, которые с виду имеют одинаковый фантик и шоколадную изнанку¸ а внутри начинка разная.

1. **Расписать в текстовом документе зачем нужны обобщение и привести несколько примеров их использования**

Обобщения позволяют создавать класс, методы и тд. не с конкретными данными, а с параметром, вместо которого можно выбрать тот тип данных, который нам нужен. То есть какой-то универсальный класс/метод и пр., который подходит для разных типов данных.

Обобщения позволяют не использовать приведение к типу и проверке типов на соответствие вручную. Обобщенный класс один раз создают, и далее от него создают экземпляры различных типов. С помощью обобщения не придется дублировать код, чтобы создать новый тип, что существенно сокращает код.