Cтруктуры в **C#** практически ничем не отличаются от структур в любом другом языке. Отличия наблюдаются лишь на более низком уровне. В основном это касается того, что для структур в **C#** не существует базового класса. Но в то же время структуры являются производными от типа **ValueType**.  
  
Зачем же нужны структуры? В принципе и без них можно легко обходиться, но использование структур делает код более понятным и наглядным. Структура - это набор зависимых друг от друга переменных. Зависимость здесь исключительно логическая и определяется условиями задачи. Чтобы стало понятно, рассмотрим простой пример. Допустим, мы пишем программу, печатающую справки для студентов. Все справки имеют один и тот же вид и текст, кроме следующих полей: ФИО, форма обучения, курс, факультет. Это зависимые данные и их можно представить в виде структуры, например так:

struct STUDENT

{

public string fio;

public string FormOfEducation;

public int course;

public string faculty;

}

Наша структура называется **STUDENT** и имеет четыре поля. После того как структура объявлена, мы можем с ней работать. Вот небольшой пример:

static void Main(string[] args)

{

STUDENT stud;

stud.fio = "Нгуен Павел Зунгович";

stud.FormOfEducation = "очного";

stud.course = 3;

stud.faculty = "электроэнергетического";

Console.WriteLine("СПРАВКА подтверждает, что "+stud.fio

+" является студентом "+stud.FormOfEducation+" отделения ВоГТУ "

+stud.course+" курса "+stud.faculty+" факультета");

Console.Read();

}

Передавать такие данные удобно, т.к. они являются одним объектом. В частности можно передавать структуры в качестве параметров функциям. Усовершенствуем нашу программу, написав отдельную функцию для печатания справок:

static void print(STUDENT stud)

{

Console.WriteLine("СПРАВКА подтверждает, что "+stud.fio

+" является студентом "+stud.FormOfEducation+" отделения ВоГТУ "

+stud.course+" курса "+stud.faculty+" факультета");

}

В качестве параметра мы передаем структуру типа **STUDENT**. Уверен, что вы скажете: "Зачем нужны структуры, когда есть классы?". Дело в том, что структуры являются более простым языковым средством. В частности имеют ограничения, которые обезопасят программиста от ошибок.  
  
Кроме переменных структуры могут иметь конструкторы и методы. Правда для структур нельзя переопределить конструктор по умолчанию, т.е. тот, который не получает никаких параметров. Конструкторы структур ничем не отличаются от конструкторов классов, поэтому не будем разжевывать этот аспект, а лишь рассмотрим простой пример:

struct STUDENT

{

public string fio;

public string FormOfEducation;

public int course;

public string faculty;

public STUDENT (string name)

{

this.fio = name;

this.FormOfEducation = "Неопределено";

this.course = 0;

this.faculty = "Неопределено";

}

}

static void Main(string[] args)

{

STUDENT stud = new STUDENT("Нгуен Павел Зунгович");

print(stud);

Console.Read();

}