CsharpDay06Assignment

Name: Mohamed Waleed Elkady

PART01

1) Why can't a struct inherit from another struct or class in C#?

A struct cannot inherit from another struct or class because it is a value type.
Structs are designed to be simple and lightweight.
They automatically inherit from `System.ValueType` and cannot inherit from any other type.
This keeps structs simple and avoids complexity.

---

2) How do access modifiers impact the scope and visibility of a class member?

Access modifiers control where a member can be accessed from.

- `private` → Accessible only inside the same class.
- `internal` → Accessible inside the same project.
- `public` → Accessible from anywhere.
- `protected` → Accessible inside the class and derived classes.

They help protect data and organize the program correctly.

---

3) Why is encapsulation critical in software design?

Encapsulation protects the data inside a class or struct.
It prevents direct access to fields and allows controlled access using methods or properties.
It helps prevent wrong data, improves security, and makes the program easier to maintain.

---

4) What is constructors in structs?

A constructor in a struct is a special method used to initialize its fields when creating an object.
It assigns initial values to the variables inside the struct.
Structs can have parameterized constructors, and they always have a default constructor that sets values to default.

---

5) How does overriding methods like ToString() improve code readability?

Overriding `ToString()` makes the object print meaningful information instead of just the type name.
It makes the output clearer and easier to understand.
It also helps during debugging and displaying data.

PART02

1) What is a Copy Constructor?

A copy constructor is a constructor that creates a new object by copying the values of another object from the same class.

In simple words, it makes a new object that has the same data as an existing object.

C# does not provide a built-in copy constructor automatically like C++, but we can create one ourselves by passing an object to the constructor and copying its fields manually.

We use a copy constructor when we want:

- To duplicate an object.
- To create a new object without affecting the original one.
- To work on a copy while keeping the original data safe.

It helps when we don't want changes in one object to affect another.

---

2) What is an Indexer?

An indexer is a feature in C# that allows us to access an object like an array using square brackets [].

For example:

objectName[0]

Instead of calling a method like GetItem(0), we can directly use the index inside brackets.

This makes the code look cleaner and easier to understand.

When do we use an Indexer?

We use an indexer when a class contains a collection of data (like a list or array), and we want to access elements by their position in a simple way.

Business cases where we use an Indexer:

- In a school system, if a class contains a list of students, we can access a student like this:
  school[0]
- In a company system, if we have a list of employees, we can access an employee like this:
  company[2]
- In an inventory system, we can access products by index like this:
  inventory[5]
- In a library system, we can access books using:
  library[3]

Using an indexer makes the class easier to use and makes the code more readable and organized.

- Linkedin article:-

Mohamed Waleed Elkady • You
Undergraduate Electronics and Communication Engineer / Full stack .Net dev...
now •

يعني إيه Constructor في C#؟
وأنت بتتعلم OOP في C#، هتلاقي حاجة اسمها Constructor بتقابلك في كل كلاس.
بس ناس كتير مش فاهمينه صح وبيستخدموه من غير ما يعرفوا دوره الحقيقي 👀

💡 تعريف بسيط
الـ Constructor هو method خاصة جوه الكلاس.
بتتنفذ automatic أول ما تعمل object جديد.
يعني أول ما تعمل:

Student s =new Student();

الmethod اللي اسمها زي اسم الكلاس هي اللي بتشتغل.
وظيفته الأساسية:
>>>يجهز الـ object
>>>يدي قيم ابتدائية للمتغيرات
>>>يضمن إن الكائن يبدأ بحالة صحيحة
باختصار، الـ Constructor بيخلي كل object يبدأ "صح" ومايبقاش ناقص أو خاطئ.

🔹 الخصائص العامة للـ Constructor
>>>اسمه لازم يكون نفس اسم الكلاس
ملوش return type (ولا حتى void)
بيتنادي بشكل automatic مع new

🔥 أنواع الـ Constructor في C#
في C# عندنا 4 أنواع رئيسية:
1️⃣ Default Constructor
2️⃣ Parameterized Constructor
3️⃣ Copy Constructor
4️⃣ Static Constructor
📸 الأمثلة العملية لكل نوع موجودة في الصورة تحت، عشان تشوف الكود مباشرة وتفهم الفرق بسهولة.
🎯 ليه الـ Constructor مهم؟
عشان الobject يتبني بحالة صحيحة من البداية
عشان مبحصلش أخطاء لما نشتغل عليه
عشان نقدر نهيئ كل object بطريقة مختلفة لو احتجنا
بخلي الكود أنضف وأسهل للقراءة والصيانة
🏅 نصيحة
لو عايز تكون مطور شاطر، لازم تعرف:
إمتى تستخدم Default
إمتى تستخدم Parameterized
إمتى تعمل Copy
وإمتى Static
كل نوع له استخدامه الصح، وفهمك لكل نوع هيسهل عليك الشغل في المشاريع الكبيرة.

Show translation

أنواع الـ Constructor في C#؟

1 Default Constructor

```
public Student()
{
    Name = "Unknown";
}
```

2 Parameterized Constructor

```
public Student(string name, int age)
{
    Name = name;
    Age = age;
}
```