TOP 3

.NET

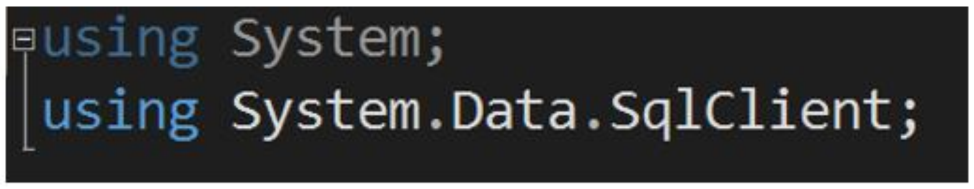
Interview Questions
(for 2- 5 Yrs Exp)

Q . What is the purpose of "using" keyword in C#?

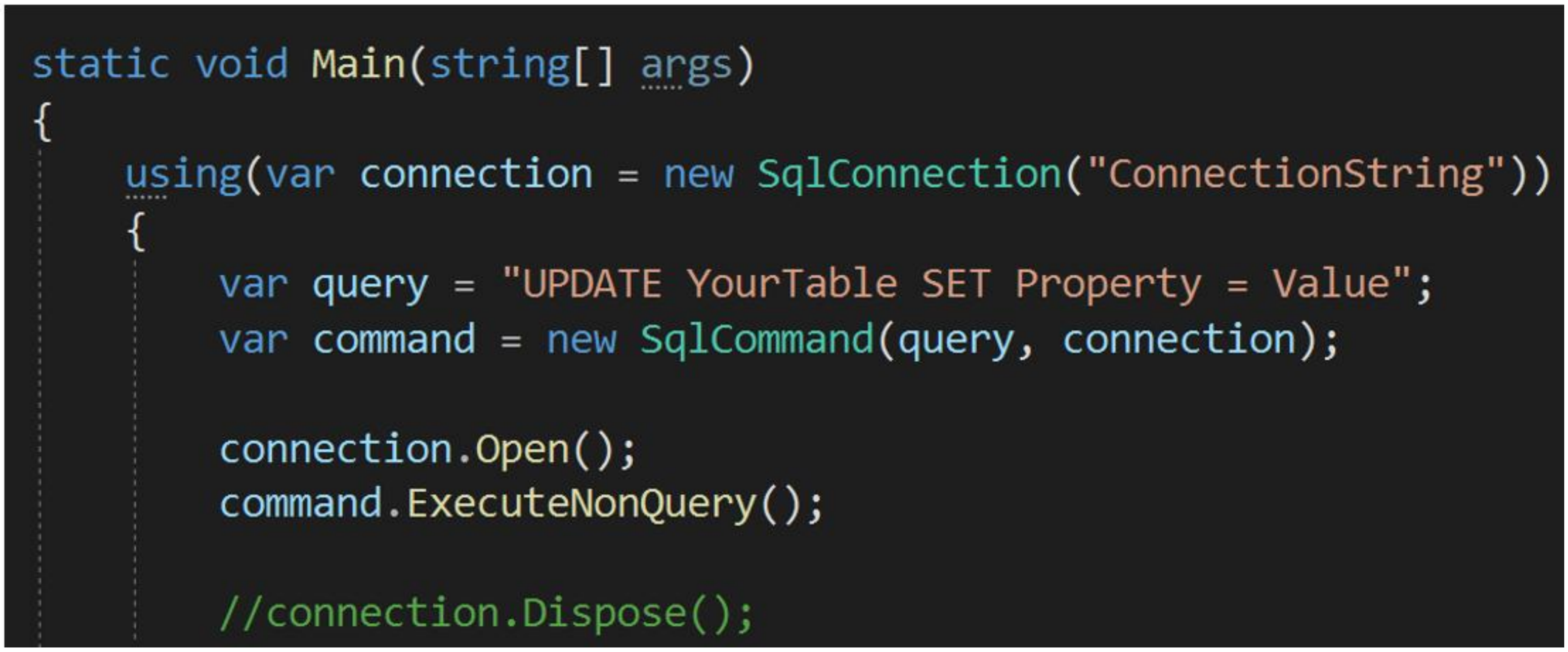
- There are two purpose of using keyword in C#:

  1. USING DIRECTIVE

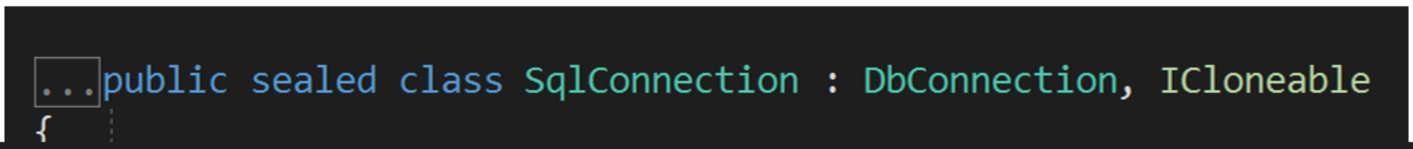
2. USING STATEMENT - The using statement ensures that DISPOSE() method of the class object is called even if an exception occurs.



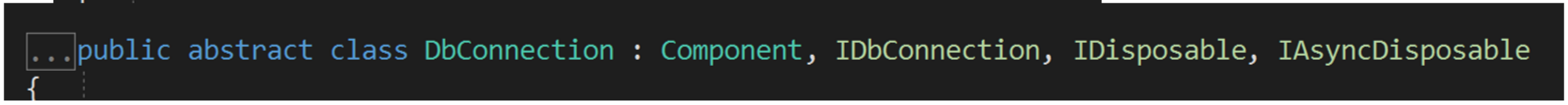
```
using System;
using System.Data.SqlClient;
```



```
static void Main(string[] args)
{
    using(var connection = new SqlConnection("ConnectionString"))
    {
        var query = "UPDATE YourTable SET Property = Value";
        var command = new SqlCommand(query, connection);

        connection.Open();
        command.ExecuteNonQuery();

        //connection.Dispose();
```



```
...public sealed class SqlConnection : DbConnection, ICloneable
{
```

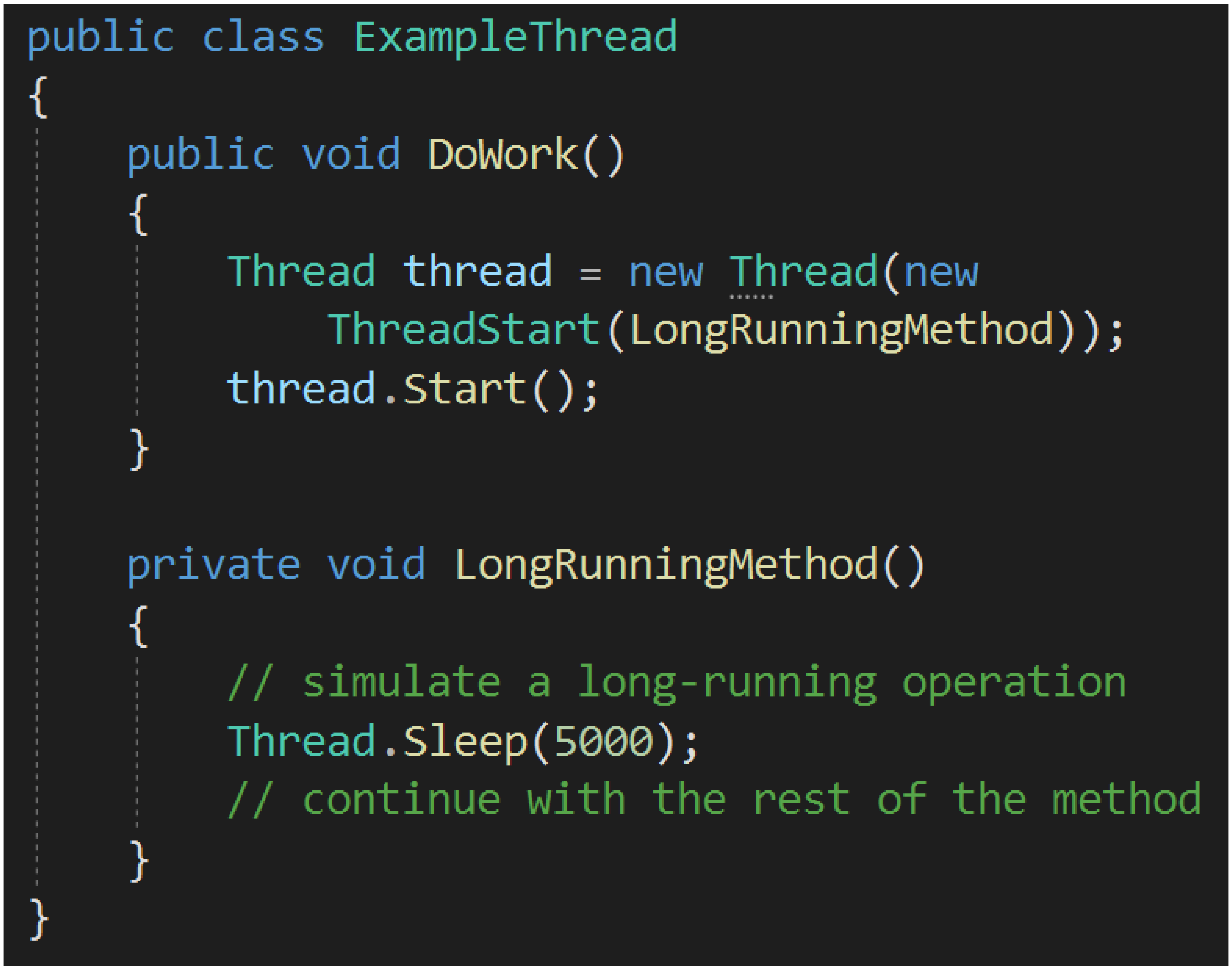


```
...public abstract class DbConnection : Component, IDbConnection, IDisposable, IAsyncDisposable
{
```

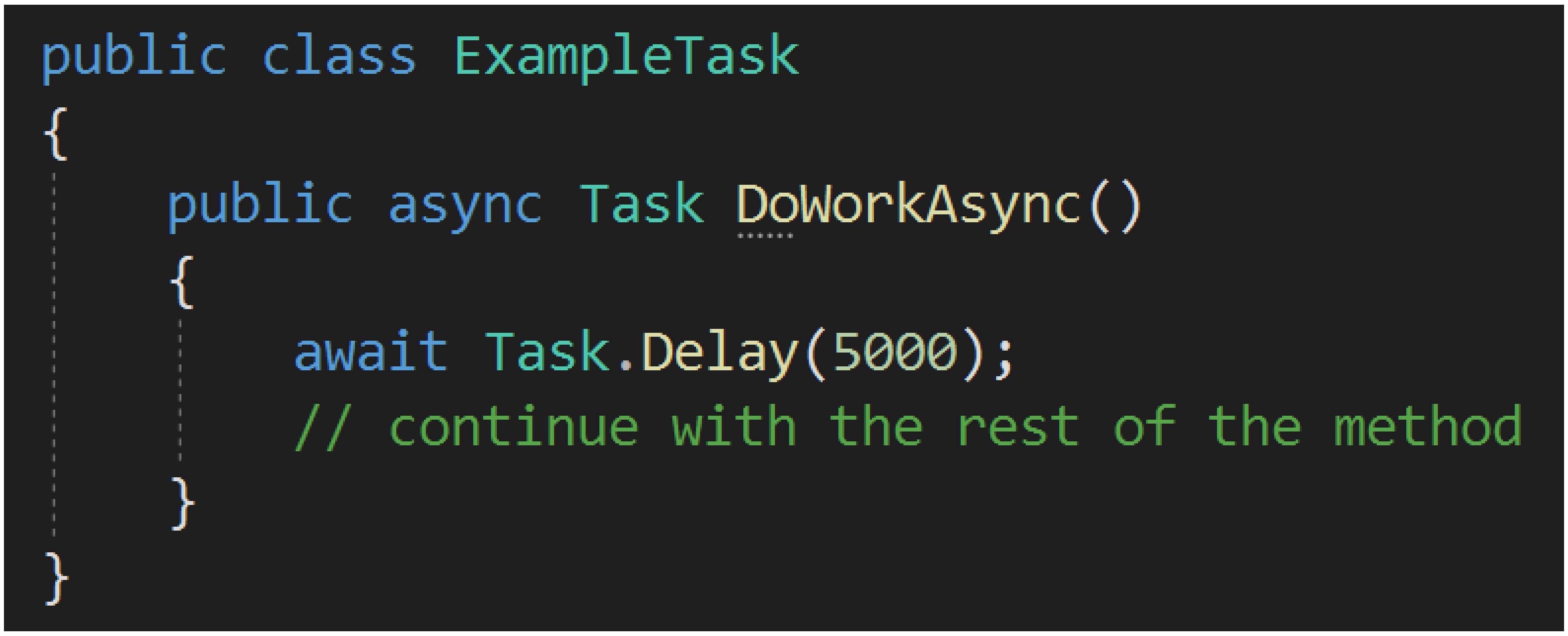
Q. What is the difference between Threads and Tasks? What are the advantages of Tasks over Threads?

- In .NET, threads and tasks are two different ways for doing **multithreading**.
- Thread is a general programming concept. On the other hand, Microsoft created Task in .NET to **simplify** the use of Threads.
- Tasks are like a **wrapper** over Threads.
- Tasks **internally** uses threads only.

Q. What is the difference between Threads and Task
What are the advantages of Tasks over Threads?



```
public class ExampleThread
{
    public void DoWork()
    {
        Thread thread = new Thread(new
            ThreadStart(LongRunningMethod));
        thread.Start();
    }

    private void LongRunningMethod()
    {
        // simulate a long-running operation
        Thread.Sleep(5000);
        // continue with the rest of the method
    }
}
```



```
public class ExampleTask
{
    public async Task DoWorkAsync()
    {
        await Task.Delay(5000);
        // continue with the rest of the method
    }
}
```

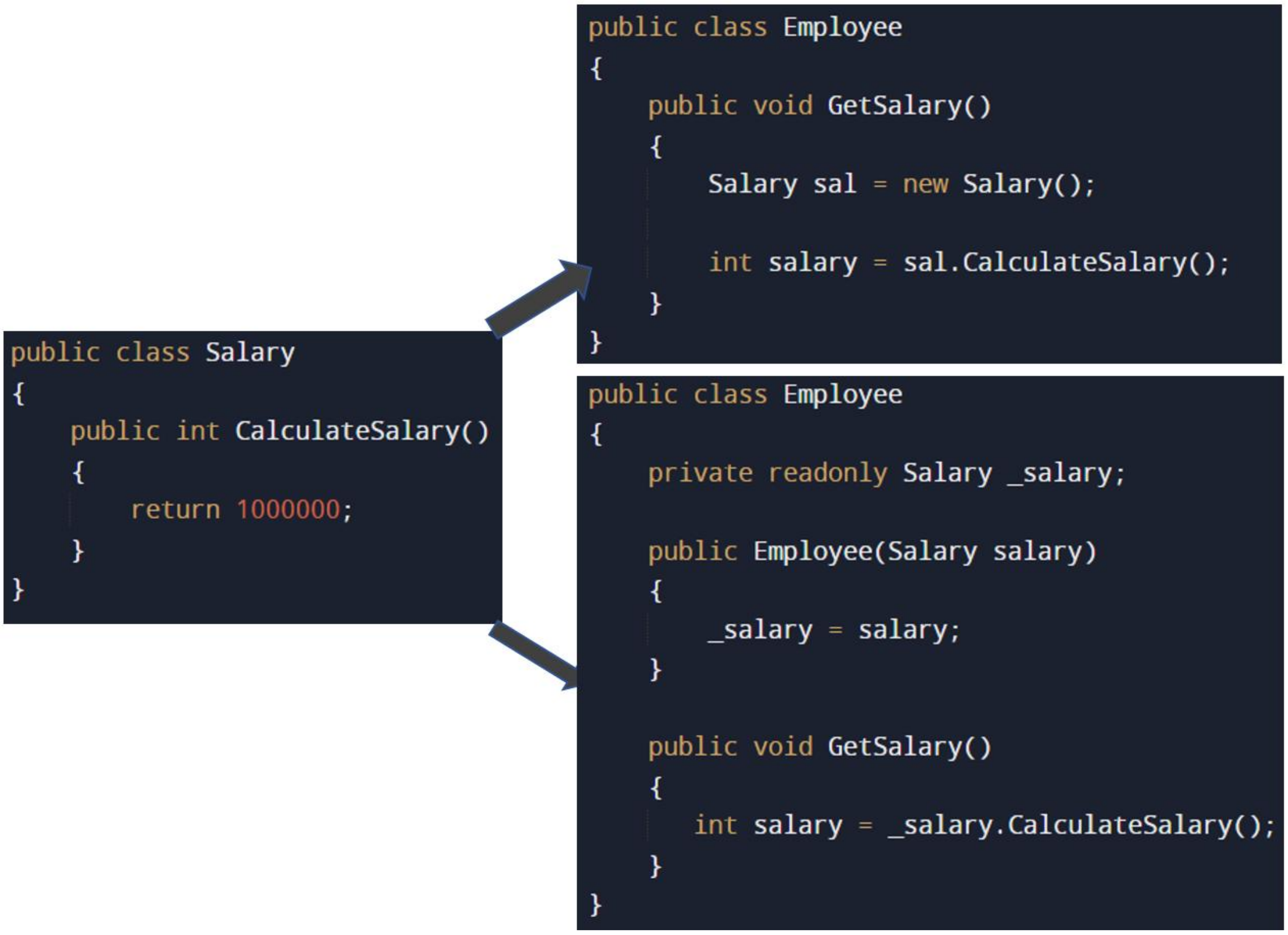
Q. What is the difference between Threads and Task What are the advantages of Tasks over Threads?

❖ Advantages of Tasks over Threads:

1. Simplified Code.
2. Exception handling.
3. A Task can **return a result**, but there is no proper way to return a result from Thread.
4. We can apply **chaining** and **parent/ child** on multiple tasks, but it can be very difficult in threads.

Q. What is Dependency Injection?

- Dependency Injection (DI) is a software **design pattern** in which we inject the dependency object of a class into another class.



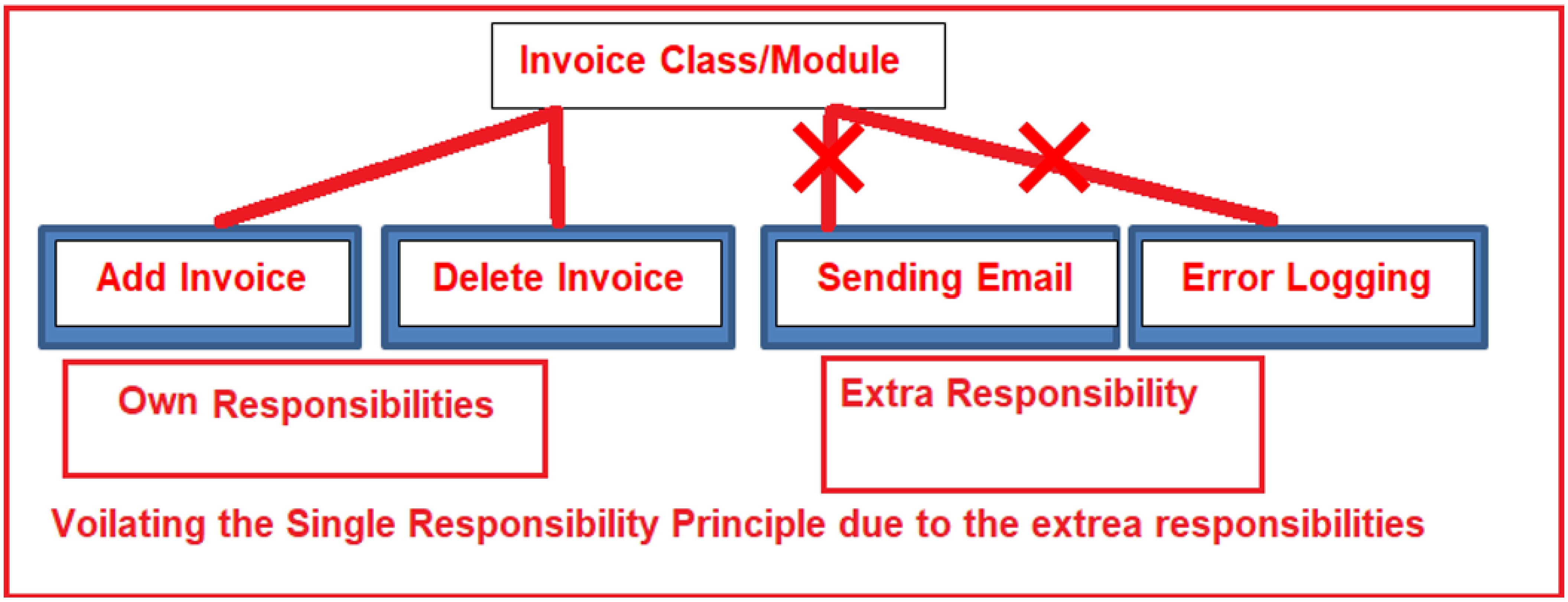
```
public class Salary
{
    public int CalculateSalary()
    {
        return 1000000;
    }
}
```

```
public class Employee
{
    public void GetSalary()
    {
        Salary sal = new Salary();

        int salary = sal.CalculateSalary();
    }
}
```

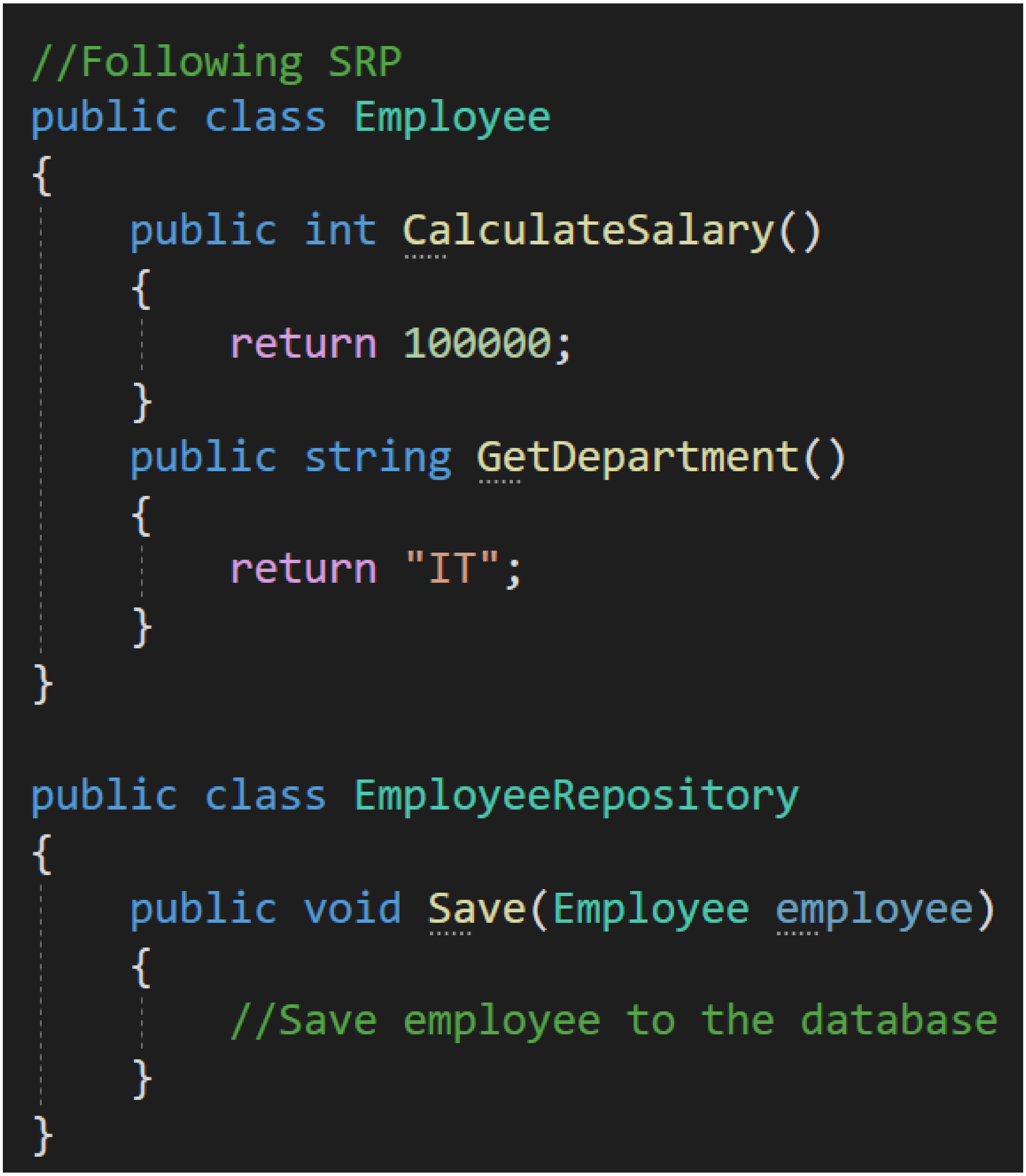
```
public class Employee
{
    private readonly Salary _salary;

    public Employee(Salary salary)
    {
        _salary = salary;
    }

    public void GetSalary()
    {
       int salary = _salary.CalculateSalary();
    }
}
```

Q. What is Single Responsibility Principle?

- Single Responsibility Principle (SRP) states that a class should have only **one responsibility**.
- Or a class should have only **one reason** to change.
- When a class has only one responsibility, it becomes easier to change and test. If a class has multiple responsibilities, changing one responsibility may impact others and more testing efforts will be required then.



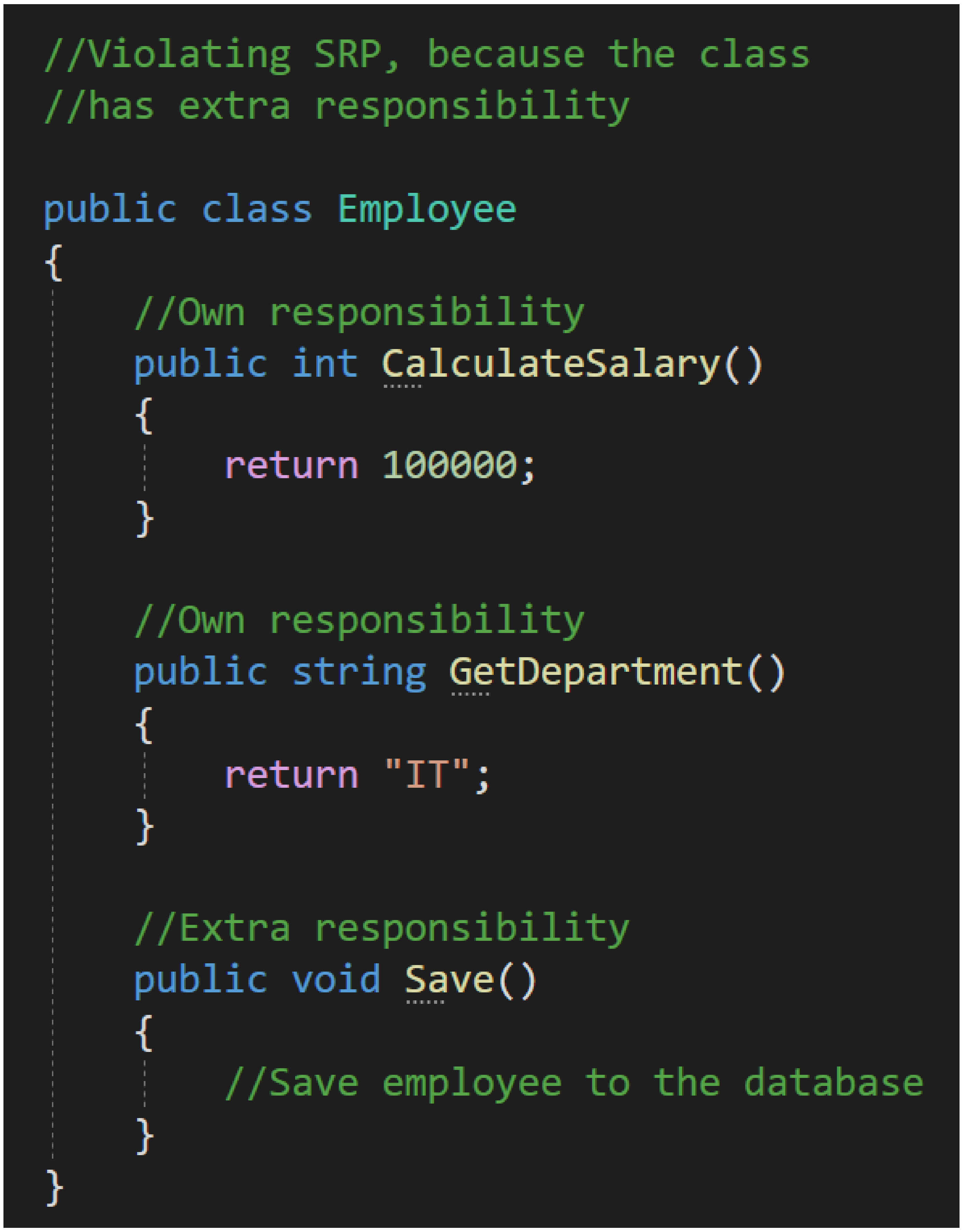
Q. What is Single Responsibility Principle?



```
//Following SRP
public class Employee
{
    public int CalculateSalary()
    {
        return 100000;
    }
    public string GetDepartment()
    {
        return "IT";
    }
}

public class EmployeeRepository
{
    public void Save(Employee employee)
    {
        //Save employee to the database
    }
}
```

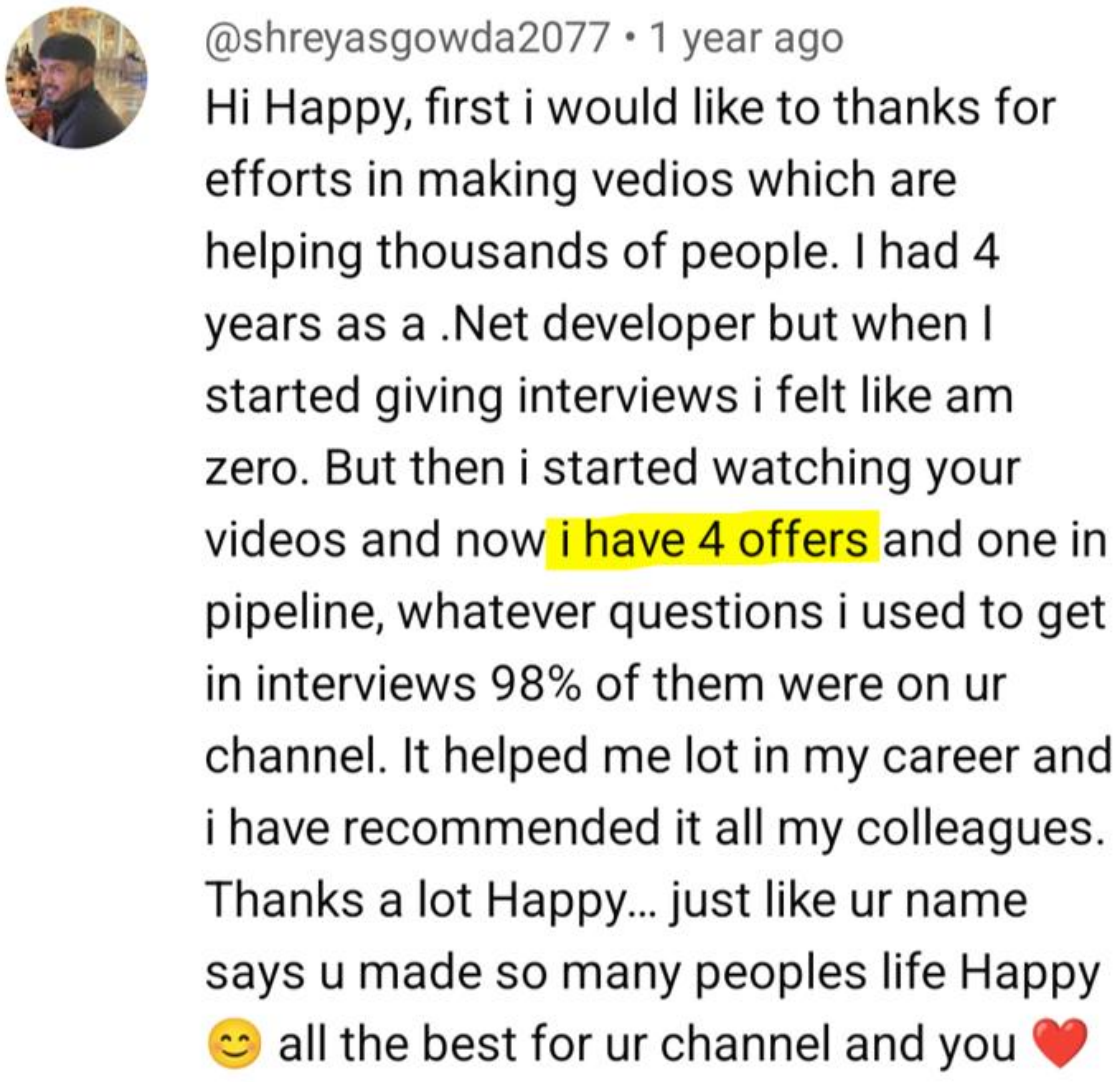
Q. What is Single Responsibility Principle?

- Single Responsibility Principle (SRP) states that a class should have only **one responsibility**.
- Or a class should have only **one reason** to change.



```
//Violating SRP, because the class
//has extra responsibility

public class Employee
{
    //Own responsibility
    public int CalculateSalary()
    {
        return 100000;
    }

    //Own responsibility
    public string GetDepartment()
    {
        return "IT";
    }

    //Extra responsibility
    public void Save()
    {
        //Save employee to the database
    }
}
```

Never Give Up Stories



@shreyasgowda2077 • 1 year ago

Hi Happy, first i would like to thanks for efforts in making vedios which are helping thousands of people. I had 4 years as a .Net developer but when I started giving interviews i felt like am zero. But then i started watching your videos and now i have 4 offers and one in pipeline, whatever questions i used to get in interviews 98% of them were on ur channel. It helped me lot in my career and i have recommended it all my colleagues. Thanks a lot Happy... just like ur name says u made so many peoples life Happy 😊 all the best for ur channel and you ❤️