OOP LAB 6 (25 Oct 22)

C# Object Oriented Programming

By: Miss Tia

Access Modifier

Why Access Modifiers? To control the visibility of class members (the security level of each individual class and class member). To achieve "Encapsulation" - which is the process of making sure that "sensitive" data is hidden from users.

```
class Program
{
    //public field that can be accessed from anywhere
    public string name;

    //private field can only be accessed within the Program class
    private int num;

}
```

In C#, there are 4 basic types of access modifiers.

- public
- private
- protected
- internal

Modifier	Description
`public`	The code is accessible for all classes
`private`	The code is only accessible within the same class
`protected`	The code is accessible within the same class, or in a class that is inherited from that class. You will learn more about inheritance in a later chapter
`internal`	The code is only accessible within its own assembly, but not from another assembly. You will learn more about this in a later chapter

Public Access Modifier

When we declare a type or type member public, it can be accessed from anywhere. For example:

- created a class name 'Student'
- created method 'print()'

Output:

```
Name: Loo Pei Xin
This is from a public print method, inside class Student
```

Since the field and method are public, we are able to access them from the Program class.

```
class Student
{
    public string name = "Loo Pei Xin";

    1 reference
    public void print()
    {
        Console.WriteLine("This output is from a public print method, inside class Student");
    }
}

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {

        // creating object of Student class
        Student student1 = new Student();

        // accessing name field and printing it
        Console.WriteLine("Name: " + student1.name);

        // accessing print method from Student
        student1.print();
        Console.ReadLine();
    }
}
```

Private Access Modifier

When we declare a type member with the private access modifier, it can only be accessed within the same class or struct. For example:

- created a class name 'Student'
- created method 'print()'

```
class Student
{
    private string name = "Loo Pei Xin";

    1 reference
    private void print()
    {
        Console.WriteLine("This output is from a public print method, inside class Student");
    }
}

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {

        // creating object of Student class
        Student student1 = new Student();

        // accessing name field and printing it
        Console.WriteLine("Name: " + student1.name);

        // accessing print method from Student
        student1.print();
        Console.ReadLine();
    }
}
```

Output:

```
prog.cs(20,45): error CS0122: `MyApplication.Student.name' is inaccessible due to its protection level
prog.cs(23,16): error CS0122: `MyApplication.Student.print()' is inaccessible due to its protection level
prog.cs(8,18): (Location of the symbol related to previous error)
Compilation failed: 2 error(s), 0 warnings
```

Since the field and method are private, we cannot access them from the Program class.

Note: By default, all members of a class are private if you don't specify an access modifier:

Protected Access Modifier

When we declare a type member as protected, it can only be accessed from the same class and its derived classes. For example,:

- created a class name 'Student'
- created method 'print()'

```
class Student
{
    protected string name = "Loo Pei Xin";

    1 reference
    protected void print()
    {
        Console.WriteLine("This output is from a public print method, inside class Student");
    }
}

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {

        // creating object of Student class
        Student student1 = new Student();

        // accessing name field and printing it
        Console.WriteLine("Name: " + student1.name);

        // accessing print method from Student
        student1.print();
        Console.ReadLine();
    }
}
```

Output:

```
prog.cs(21,45): error CS0122: `MyApplication.Student.name' is inaccessible due to its protection level
prog.cs(21,45): error CS1540: Cannot access protected member `MyApplication.Student.name' via a qualifier of type `
prog.cs(6,22): (Location of the symbol related to previous error)
prog.cs(24,16): error CS1540: Cannot access protected member `MyApplication.Student.print()' via a qualifier of typ
prog.cs(8,24): (Location of the symbol related to previous error)
prog.cs(24,16): error CS0122: `MyApplication.Student.print()' is inaccessible due to its protection level
prog.cs(8,24): (Location of the symbol related to previous error)
Compilation failed: 4 error(s), 0 warnings
```

Protected Access Modifier

How to access it? inheriting the Program class from the Student class.

Output:

```
Name: Loo Pei Xin
This lalala output is from a public print method, inside class Student
```

```
class Student
{
    protected string name = "Loo Pei Xin";

    1 reference
    protected void print()
    {
        Console.WriteLine("This lalala output is from a public print method, inside class Student");
    }
}

2 references
class Program : Student
{
    0 references
    static void Main(string[] args)
    {

        // creating object of Program class
        Program student1 = new Program();

        // accessing name field and printing it
        Console.WriteLine("Name: " + student1.name);

        // accessing print method from Student
        student1.print();
        Console.ReadLine();
    }
}
```

Let the debugging begins

Exercise: create class animal and cat (or any other animal you choose). Utilize the protected access modifier and inheritance.

Output:

```
I am a cat
My name is lola
```