Swinburne University of Technology

Object Oriented Programming (2020 S2)

Doubtfire Submission

Semester Test

Submitted By:
Kevin Pham
102683681
2020/10/02 14:56

Tutor:
Charlotte Pierce

October 2, 2020

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SemesterTest
{
    class Program
    {
        static void Main(string[] args)
        {

            Library library = new Library("Brimbank Library");


            Book book1 = new Book("Apocrypha Now", "Mark Russell", "
            ↪  978-1-60309-369-9");

            book1.OnLoan = false;


            Book book2 = new Book("Bottled", "Chris Gooch", "978-1-60309-420-7");

            book2.OnLoan = true;



            Game game1 = new Game("DOOM", "Bethesda", "R18+");

            game1.OnLoan = false;

            Game game2 = new Game("Super Mario Odyssey", "Nintendo", "G");

            game2.OnLoan = true;



            library.AddResource(book1);

            library.AddResource(book2);

            library.AddResource(game1);

            library.AddResource(game2);


            Console.WriteLine("DOOM is available?: " + library.HasResource("DOOM"));
            Console.WriteLine("Bottled is available?: " +
            ↪  library.HasResource("Bottled"));

            Console.ReadLine();
        }
```

```
52      }
53  }
```

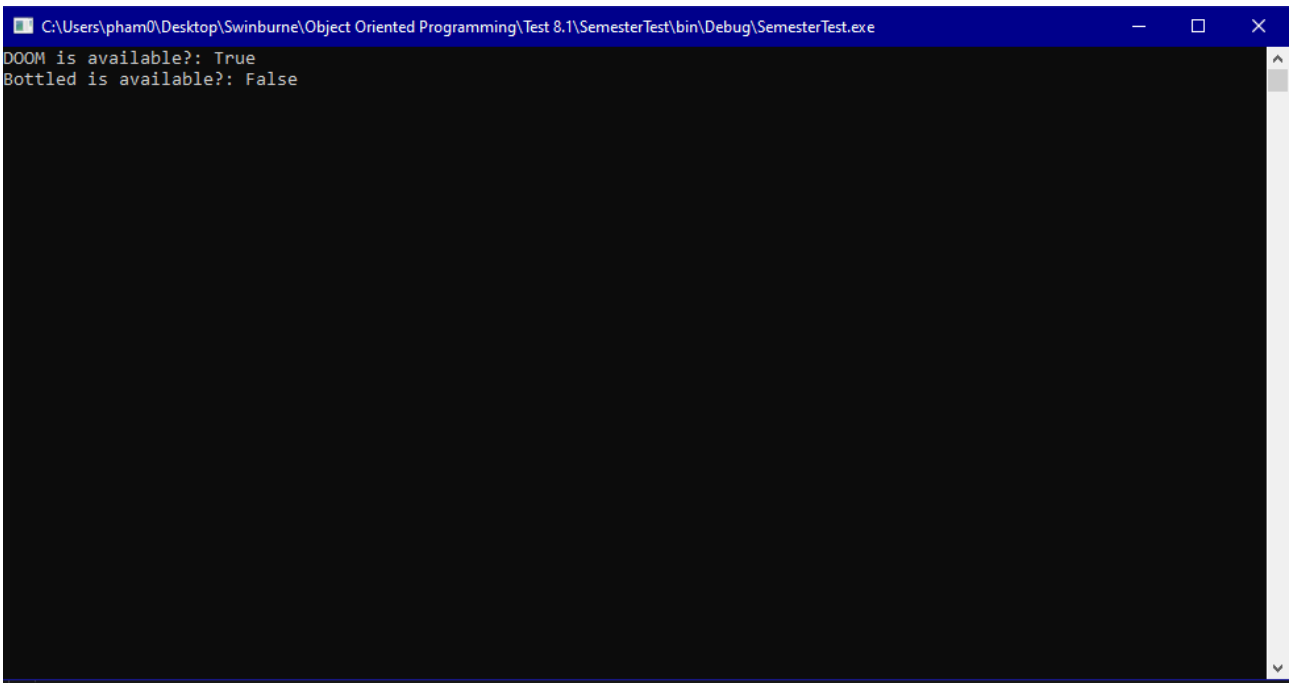
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SemesterTest
{
    class Library
    {
        private string _name;
        private List<LibraryResource> _resources;

        public Library(string name)

        {

            _resources = new List<LibraryResource>();
            this._name = name;

        }

        public void AddResource(LibraryResource resource)

        {

            if (resource != null)

            {

                _resources.Add(resource);

            }

        }

        public bool HasResource(string name)

        {

            foreach (LibraryResource resource in _resources)

            {

                if (resource.Name.ToLower() == name.ToLower() && resource.OnLoan ==
                ↪  false)

                {

                    return true;

                }

```

```
53                }
54
55                return false;
56
57            }
58
59        }
60
61    }
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SemesterTest
{
    abstract class LibraryResource
    {
        private string _name;
        private bool _onLoan;

        public LibraryResource(string name)

        {

            _name = name;

            _onLoan = false;

        }

        public string Name
        {
            get
            {
                return _name;
            }
        }

        public bool OnLoan
        {
            get
            {
                return _onLoan;
            }
            set
            {
                _onLoan = value;
            }
        }

    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SemesterTest
{
    class Game: LibraryResource
    {
        private string _developer;
        private string _contentRating;

        public Game(string name, string developer, string contentRating) :
        ↪ base(name)

        {
            _developer = developer;
            _contentRating = contentRating;
        }

        public string Developer
        {
            get
            {
                return _developer;
            }
        }
        public string ContentRating
        {
            get
            {
                return _contentRating;
            }
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SemesterTest
{
    class Book: LibraryResource
    {
        private string _author;
        private string _isbn;

        public Book(string name, string author, string isbn) : base(name)

        {

            _author = author;

            _isbn = isbn;

        }

        public string Author
        {
            get
            {
                return _author;
            }
        }

        public string ISBN
        {
            get
            {
                return _isbn;
            }
        }
    }
}
```


C:\Users\pham0\Desktop\Swinburne\Object Oriented Programming\Test 8.1\SemesterTest\bin\Debug\SemesterTest.exe
DOOM is available?: True
Bottled is available?: False

Task 2: Polymorphism and Abstraction

In object-oriented principle terms, polymorphism is defined by a classes ability to perform multiple different methods with identical names that are initialised with different parameters. In the case of the previous task, an example of this is demonstrated in lines 38 to 44 of the Program class shown in Figure 1 below:

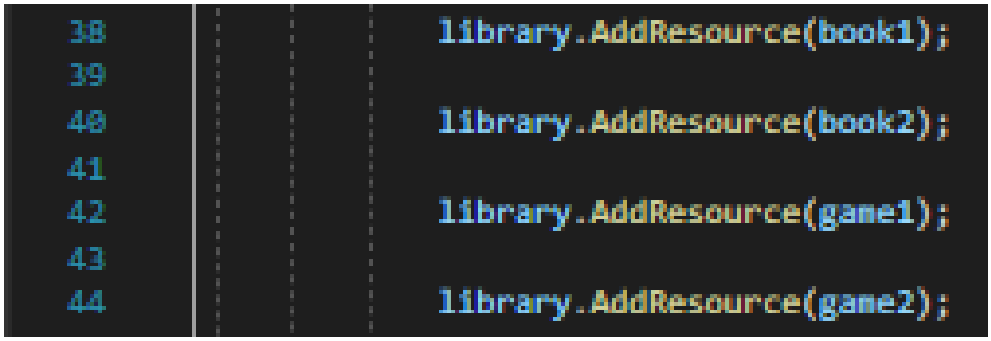


Figure 1: Polymorphism Example

This example demonstrates polymorphism as there are four AddResource() methods each with unique and unrelated parameters such as book1 and game2. Polymorphism is used here as it provides the benefit of reusing the same code for each parameter rather than typing it repeatedly.

On the other hand, abstraction generally refers to concealing unnecessary details of code and stating only the required characteristics of an object. In the case of the previous task, an example of abstraction is demonstrated in lines 47 to 48 of the Program class shown in Figure 2 below:

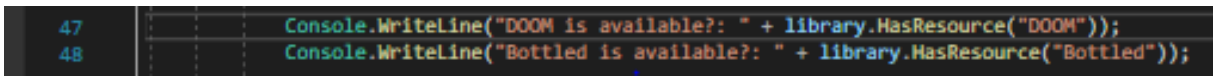
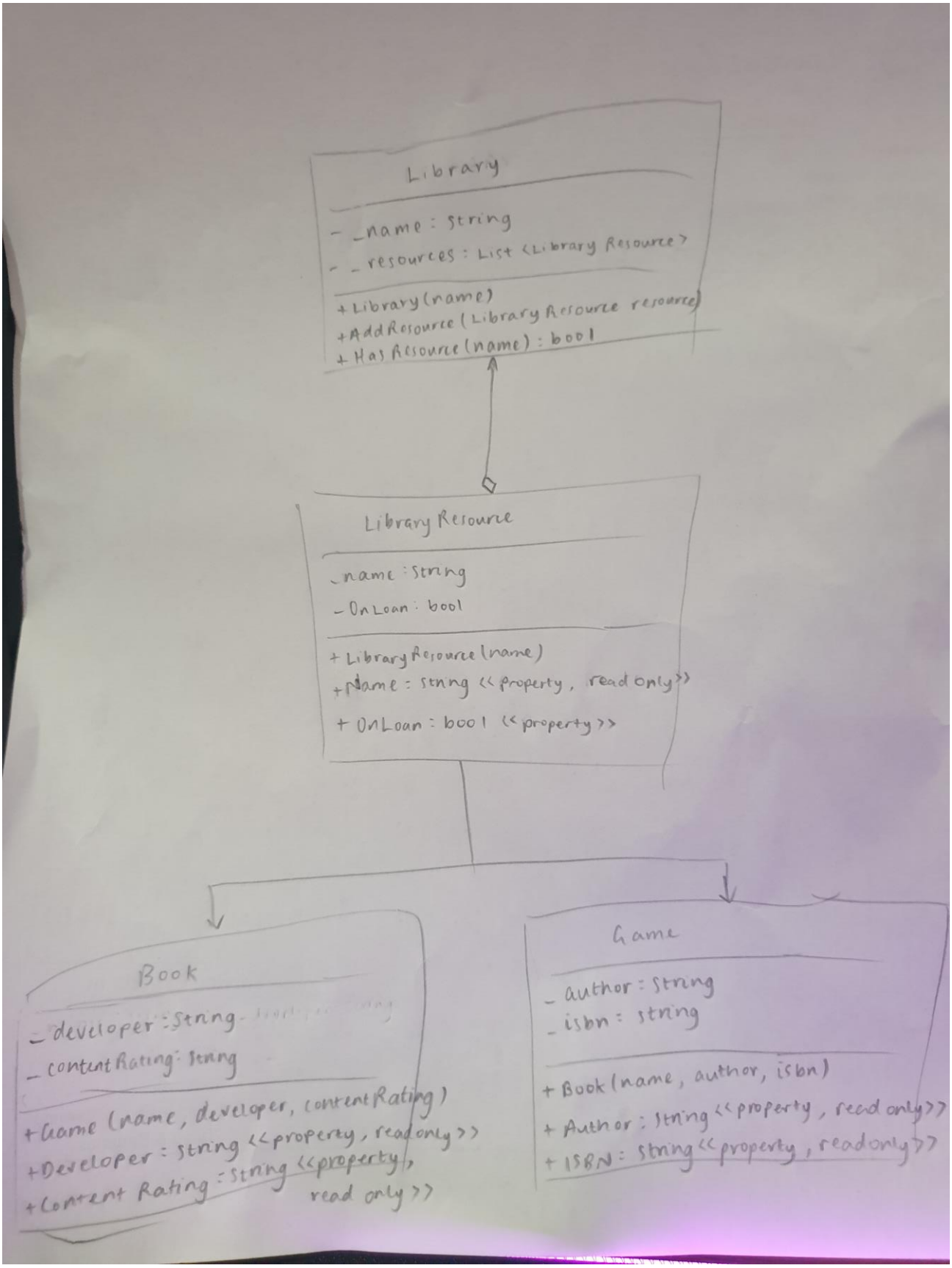


Figure 2: Abstraction Example

This example demonstrates abstraction is it will simply show the answer to the user's request, abstracting the search code that underlies the HasResource() method. Conclusively, an advantage of implementing abstraction is that the user does not need to see how the code looks for whether what they're looking for is in the library, they just need to know if it is available.


Library
- _name : string
- _resources : List <Library Resource>
+ Library(name)
+ AddResource(LibraryResource resource)
+ HasResource(name) : bool
Library Resource
_name : String
- OnLoan : bool
+ LibraryResource(name)
+ Name : string <<property, read only>>
+ OnLoan : bool <<property>>
Book
- developer : String
- contentRating : String
+ Game(name, developer, contentRating)
+ Developer : string <<property, read only>>
+ Content Rating : String <<property, read only>>
Game
- author : String
- isbn : string
+ Book(name, author, isbn)
+ Author : String <<property, read only>>
+ ISBN : String <<property, read only>>