SIT771

Object Oriented Development

Learning Summary Report

Achmad Mustafa Kemal
219374683

Self-Assessment Details

The following checklists provide an overview of my self-assessment for this unit.

	Pass (D)	Credit (C)	Distinction (B)	High Distinction (A)
Self-Assessment				✓

Self-Assessment Statement

	Included
Learning Summary Report	✓
Pass tasks complete	✓

Minimum Pass Checklist

	Included
All Credit Tasks are Complete on Doubtfire	✓

Minimum Credit Checklist (in addition to Pass Checklist)

	Included
Distinction tasks (other than Custom Program) are Complete	✓
Custom program meets Distinction criteria	✓

Minimum Distinction Checklist (in addition to Credit Checklist)

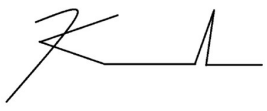
	Included
Something Awesome included	✓
Custom project meets HD requirements	✓

Minimum High Distinction Checklist (in addition to Distinction Checklist)

Declaration

I declare that this portfolio is my individual work. I have not copied from any other student's work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part of this submission been written for me by another person.

Signature: **Achmad Mustafa Kemal**



Portfolio Overview

This portfolio includes work that demonstrates that I have achieve all Unit Learning Outcomes for SIT771 Unit Title to a **High Distinction** level.

I believe that I should get a High Distinction grade because I finish all task with a High distinction grade target. Also, I believe that I achieved with all unit learning outcomes for SIT771. I can draw concept visualization for C# programming language. I can build a custom program for my project in the SIT771 on track. I can know which principle of object-oriented programming including abstraction, encapsulation, inheritance and polymorphism that I can used to my code program. I can identify which problem that my friend faced in their code. I suggest the concept to make easy understanding. Then, my friend can figure out their problem. For High distinction candidate, I believe that I'm a high distinction candidate because I have a lot of knowledge about C# programming language after I learn in this subject. I can explain about the concept of C# to my classmates.

To become High Distinction candidate, I believe that I cover all aspects including intended learning outcomes. There are evaluate code, principles, build program, design and justify.

In the evaluate code, i must evaluate my code or code hat given in the task. So, I debugging the code to check if the code has some issue. Evaluate code is all about behavior to us for teach us to not go straight to run a code. We must evaluate code before run it. There are tasks that with this learning outcome. Such as, Task 1.3p (How many objects) and Task 4.2c (Messy Code). For Task 1.3p, we must evaluate code that given in the task and then I must evaluate that how many objects inside the code. I must evaluate the code very carefully in this task. For Task 4.2 c, I must use code refactoring and code formatting to evaluate the code in this task because the code is looks messy and we must evaluate and give reflection to this task.

In the principle learning outcome, I must apply and explain the principles of object-oriented programming including abstraction, encapsulation, inheritance and polymorphism. This learning outcome relate to Task 7.1p (Abstract transaction), Task 7.2c (Different Robot), Task 4.4c (Concept Visualization 1) and Task 8.2c (Concept Visualization 2). For task 7.1p, the task must apply abstract principle to execute transaction. In the Task 7.2c, I must abstract and polymorphism to make robot appear different type when the program has been run it. For Task 8.2c and Task 4.4c, I draw own concept visualization including class, object, construction, method, field, property, control flows, inheritance, polymorphism, delegate and lambdas. All of those tasks make me to easier to understanding to know the principle of object-oriented programming.

In the build program section, I think almost task make us build program. Such as, Task 7.3d (Custom Program Code), Task 7.1p (Abstract Transaction), Task 5.3p(Warehouse), Task 5.4c(Many robot), task 9.1c (Another Language) and Task 7.2c (Different Robot). I must implement and test the program.

In the design section, I must design, communicate and evaluate with using diagram. Task 6.1p (Document Design) is make me to design a UML Class diagram for Warehouse. With using UML Class diagram, I can see how the warehouse flow when the code program run it. Another

task is 7.3d (custom program code). Before implement a code in the custom program, I must make design of my custom program and then send to the lecture for approval.

The last learning outcome is aligning justify in this subject. I'm must give relevant proof from the code. I must justify the principle of object-oriented programming with code evidence. The task 8.1p (Code Interview – explanation) is related to justify learning outcome. So, I must explain the key of principle including abstraction, encapsulation, inheritance and polymorphism. Also, the task 7.3d (custom program code) also need to justify about what concept that I'm going to use for my custom program.

In my conclusion, I should get a High Distinction grade because I'm doing better in all tasks. I also give to this subject a best result in this tri semester.

Reflection

The most important things I learnt:

I learnt about how to create a language C# with splash kit. I learn a lot about C# programming language. In this subject, I realized that C# have many concepts to use it in the program. Such as, inheritance, polymorphism, delegates and lambdas. In this subject, I learn how to make code indentation and code refactoring. Code indentation is all about code formatting. I believe that code formatting is also important in the software development. Code Refactoring is a method that fix a problem without changing present method in the code. In the end, I learn what I expected about C# programming language.

The things that helped me most were:

Lecture class is a thing that helped me a lot because we can know the fundamental of C# programming language. Having a discussion with the lecture is helped me a lot because my unit chair gives good explanation and clear feedback. She always gives me respond when I need a help on my ontrack task. She always patience what problem that I have it on my code. She always has best solution to solve the problem. Furthermore, discussing with classmate is another thing that helped me most because we can have different idea and talk about it. Different idea is making me have more knowledge about concept of C#. In addition, cloud Deakin resource is also helped me a lot because when I late a class and the unit chair is busy. The cloud Deakin resource is helping me to fix the problem.

I found the following topics particularly challenging:

The challenging topics are all robot dodge task in this subject because the final result of the task is we can have developed C# game in this subject. There are a lot of time to write a code about this task and I must be carefully write a code in this task because we must sure player have bullet to shot robot, robot have different type, the user also can move player in the game window.

I found the following topics particularly interesting:

Messy code is a topic that I interested to do it because I must fix code that given by the task without changing present method that I already given. This task want me to do code formatting and share the reflection about this task. Concept visualization 1 and Concept visualization 2 are very interesting. So, we must explain about concept of C# programming with our own idea. For example, I draw mind map to discuss about class, object, constructor, method, field, property and control flows in the concept visualization 1 which is I found interesting that I can understanding what is C# concept. I also found better understanding in concept visualization 2.

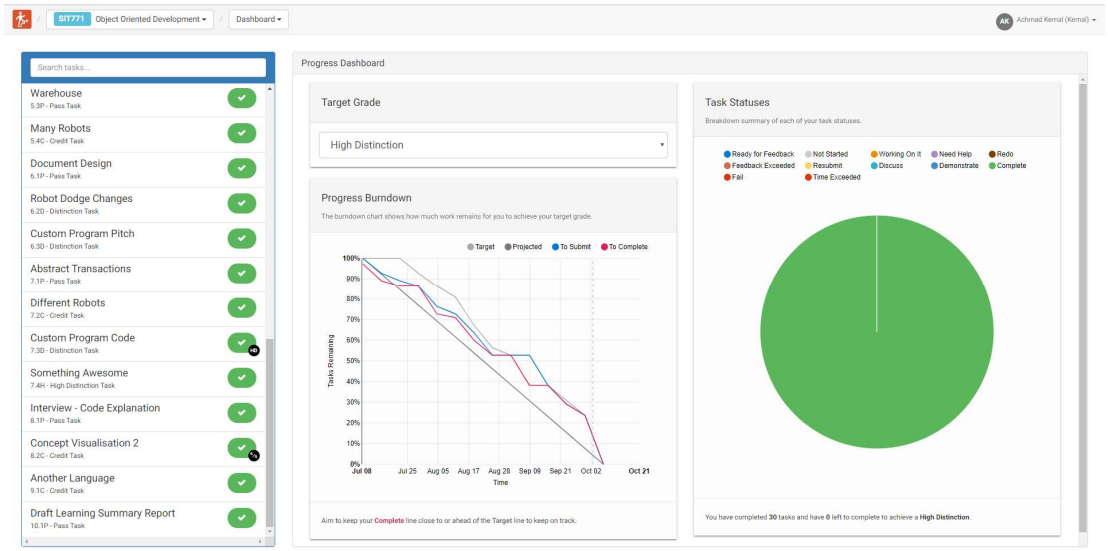
I feel I learnt these topics, concepts, and/or tools really well:

With UML class diagram, I learnt really well about C# programing. I know which one is public and private. UML class diagram is like guide map to develop a C# program code. I can identify about method, class and constructor.

I still need to work on the following areas:

I still need to work about how to have a code formatting because sometime I found that my code is very hard to read and understand. Sometime, I write unnecessary code that make confuses because I think too hard about my program. Choose simple variable name is important thing that I need to work on because I choose long variable that can cause problem.

My progress in this unit was …:



My progress in this unit is almost every pass, credit, distinction and high distinction task. Already done with all task of SIT771. After I finish all task, I hope I can get HD result.

This unit will help me in the future:

This unit will help me in my career because I found that many IT company use C# programming language to develop project. C# is a most common programming language. I'm very grateful that learn C# programming language.

If I did this unit again I would do the following things differently:

In the future, I would like to manage time about learn C# more in-depth because C# is new experience for me. Because I want C# become my ability of programming skill. So, I need to learn carefully if I did this unit again.

Other…:

My reflection is there are a lot concept that I should learn more in depth for better understanding. Because C# is not only IF logical statement but C# is all about inheritance, polymorphism, delegate, lambda and many more.

Deakin University

Object Oriented Development

Achmad Mustafa Kemal

Portfolio Submission

Submitted By:
Achmad Mustafa Kemal
akemal

Tutor:
Mengmeng Ge

October 3, 2019

Contents

2 Overall Task Status

Task	Status	Times Assessed
Hello World	Complete	1
Shape Drawing	Complete	1
How Many Objects?	Complete	4
Making A Scene	Complete	1
Help Others	Complete	2
Hello User	Complete	1
The Stock class	Complete	1
The Player Class	Complete	2
Name Tester	Complete	3
Validating Stock Actions	Complete	3
Moving The Player	Complete	4
Transactions	Complete	4
Messy Code	Complete	2
Robot Dodge	Complete	1
Concept Visualisation 1	Complete	2
Arrays	Complete	2
Lists	Complete	2
Warehouse	Complete	2
Many Robots	Complete	2
Document Design	Complete	3
Robot Dodge Changes	Complete	4
Custom Program Pitch	Complete	1
Abstract Transactions	Complete	3
Different Robots	Complete	1
Custom Program Code	Complete	2
Something Awesome	Complete	2
Interview - Code Explanation	Complete	1
Concept Visualisation 2	Complete	3
Another Language	Complete	1
Draft Learning Summary Report	Complete	2

3 Learning Outcomes

3.1 Evaluate Code

Evaluate simple program code for correct use of coding conventions, and use code tracing and debugging techniques to identify and correct issues.

Task	Rating	Status	Times Assessed
How Many Objects?	♦♦♦♦◊	Complete	4
Messy Code	♦♦♦♦◊	Complete	2
Custom Program Code	♦♦♦◊◊	Complete	2
Something Awesome	♦♦♦♦♦	Complete	2

3.2 Principles

Apply and explain the principles of object oriented programming including abstraction, encapsulation, inheritance and polymorphism.

Task	Rating	Status	Times Assessed
The Stock class	♦♦♦♦◊	Complete	1
Name Tester	♦♦♦♦◊	Complete	3
Validating Stock Actions	♦♦♦♦◊	Complete	3
Moving The Player	♦♦♦♦◊	Complete	4
Transactions	♦♦♦◊◊	Complete	4
Messy Code	♦♦♦♦◊	Complete	2
Concept Visualisation 1	♦♦♦◊◊	Complete	2
Different Robots	♦♦♦♦◊	Complete	1
Custom Program Code	♦♦♦◊◊	Complete	2
Something Awesome	♦♦♦♦♦	Complete	2
Concept Visualisation 2	♦♦♦♦◊	Complete	3
Another Language	♦♦♦♦◊	Complete	1

3.3 Build Programs

Implement, and test small object oriented programs that conform to planned system structures and requirements

Task	Rating	Status	Times Assessed
Hello World	♦♦♦♦♦	Complete	1
Making A Scene	♦♦♦♦◊	Complete	1
Hello User	♦♦♦♦◊	Complete	1
The Stock class	♦♦♦◊◊	Complete	1
The Player Class	♦♦♦♦◊	Complete	2
Validating Stock Actions	♦♦♦♦◊	Complete	3
Moving The Player	♦♦♦♦◊	Complete	4
Transactions	♦♦♦♦◊	Complete	4
Robot Dodge	♦♦♦◊◊	Complete	1
Arrays	♦♦♦◊◊	Complete	2
Lists	♦♦♦♦◊	Complete	2
Many Robots	♦♦♦◊◊	Complete	2
Robot Dodge Changes	♦♦♦♦◊	Complete	4
Abstract Transactions	♦♦♦♦◊	Complete	3
Different Robots	♦♦♦♦◊	Complete	1
Custom Program Code	♦♦♦◊◊	Complete	2
Something Awesome	♦♦♦♦♦	Complete	2
Interview - Code Explanation	♦♦♦♦◊	Complete	1
Another Language	♦♦♦♦◊	Complete	1

3.4 Design

Design, communicate, and evaluate solution structures using appropriate diagrams and textual descriptions.

Task	Rating	Status	Times Assessed
Shape Drawing	♦♦♦♦♦	Complete	1
Making A Scene	♦♦♦♦◊	Complete	1
The Player Class	♦♦♦♦◊	Complete	2
Validating Stock Actions	♦♦♦♦◊	Complete	3
Moving The Player	♦♦♦♦◊	Complete	4
Transactions	♦♦♦◊◊	Complete	4
Robot Dodge	♦◊◊◊◊	Complete	1
Concept Visualisation 1	♦♦♦◊◊	Complete	2
Warehouse	♦♦♦♦◊	Complete	2
Many Robots	♦♦♦◊◊	Complete	2
Document Design	♦♦♦♦◊	Complete	3
Robot Dodge Changes	♦♦♦♦◊	Complete	4
Custom Program Pitch	♦♦♦♦◊	Complete	1
Abstract Transactions	♦♦♦♦◊	Complete	3
Custom Program Code	♦♦♦◊◊	Complete	2
Something Awesome	♦♦♦♦♦	Complete	2
Another Language	♦♦♦♦◊	Complete	1

3.5 Justify

Justify meeting specified outcomes through providing relevant evidence and critiquing the quality of that evidence against given criteria.

Task	Rating	Status	Times Assessed
Help Others	♦♦♦♦◊	Complete	2
Validating Stock Actions	♦♦♦♦◊	Complete	3
Messy Code	♦♦♦♦◊	Complete	2
Robot Dodge	♦◊◊◊◊	Complete	1
Concept Visualisation 1	♦♦♦◊◊	Complete	2
Arrays	♦♦◊◊◊	Complete	2
Lists	♦♦◊◊◊	Complete	2
Warehouse	♦◊◊◊◊	Complete	2
Many Robots	♦♦♦◊◊	Complete	2
Document Design	♦♦♦◊◊	Complete	3
Robot Dodge Changes	♦♦♦♦◊	Complete	4
Custom Program Pitch	♦♦♦♦◊	Complete	1
Abstract Transactions	♦♦♦♦◊	Complete	3
Different Robots	♦♦♦♦◊	Complete	1
Custom Program Code	♦♦♦◊◊	Complete	2
Something Awesome	♦♦♦♦♦	Complete	2
Concept Visualisation 2	♦♦♦♦◊	Complete	3
Draft Learning Summary Report	♦♦♦♦♦	Complete	2

4 Hello World

Get started by testing the tools you have installed.

Outcome	Weight
Build Programs	♦♦♦♦♦

HelloWorld is a basic code to build programs

Date	Author	Comment
2019/07/08 16:05	Achmad Kemal	This task makes me learn about C# programming language

Deakin University

Object Oriented Development

OnTrack Submission

Hello World

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/08 16:05

Tutor:
Mengmeng Ge

Outcome	Weight
Build Programs	♦♦♦♦♦

HelloWorld is a basic code to build programs

July 8, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
    Console.WriteLine ("Hello World");

        Window w = new Window(" My First Program", 200, 100);
        w.DrawText("Hello World", Color.Black, 10, 45);
        w.Refresh(60);
        SplashKit.Delay(5000);
    }
}
```


Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet run
Hello World
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 26.55 ms for D:\Users\Kemal\Documents\Code\KemalHelloWorld\KemalHelloWorld.csproj.
KemalHelloWorld -> D:\Users\Kemal\Documents\Code\KemalHelloWorld\bin\Debug\netcoreapp2.0\KemalHelloWorld.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:01.40
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet run
Hello World
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ ls
bin KemalHelloWorld.csproj lib myeasylog.log obj Program.cs
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ pwd
/d/Users/Kemal/Documents/Code/KemalHelloWorld
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 27.35 ms for D:\Users\Kemal\Documents\Code\KemalHelloWorld\KemalHelloWorld.csproj.
KemalHelloWorld -> D:\Users\Kemal\Documents\Code\KemalHelloWorld\bin\Debug\netcoreapp2.0\KemalHelloWorld.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:00.82
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet run
Hello World
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet run
Hello World
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 26.65 ms for D:\Users\Kemal\Documents\Code\KemalHelloWorld\KemalHelloWorld.csproj.
KemalHelloWorld -> D:\Users\Kemal\Documents\Code\KemalHelloWorld\bin\Debug\netcoreapp2.0\KemalHelloWorld.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:00.84
Kemal@MSI MINGW64 /d/Users/Kemal/Documents/Code/KemalHelloWorld
$ skm dotnet run
My ...
Hello World

5 Shape Drawing

Create and draw shapes to the screen using SplashKit!

Outcome	Weight
Design	♦♦♦♦♦

This task makes design a thing with a c# programming language. And very useful

Date	Author	Comment
2019/07/08 17:00	Achmad Kemal	Good Task for the student who learns c# programming language

Deakin University

Object Oriented Development

OnTrack Submission

Shape Drawing

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/08 17:00

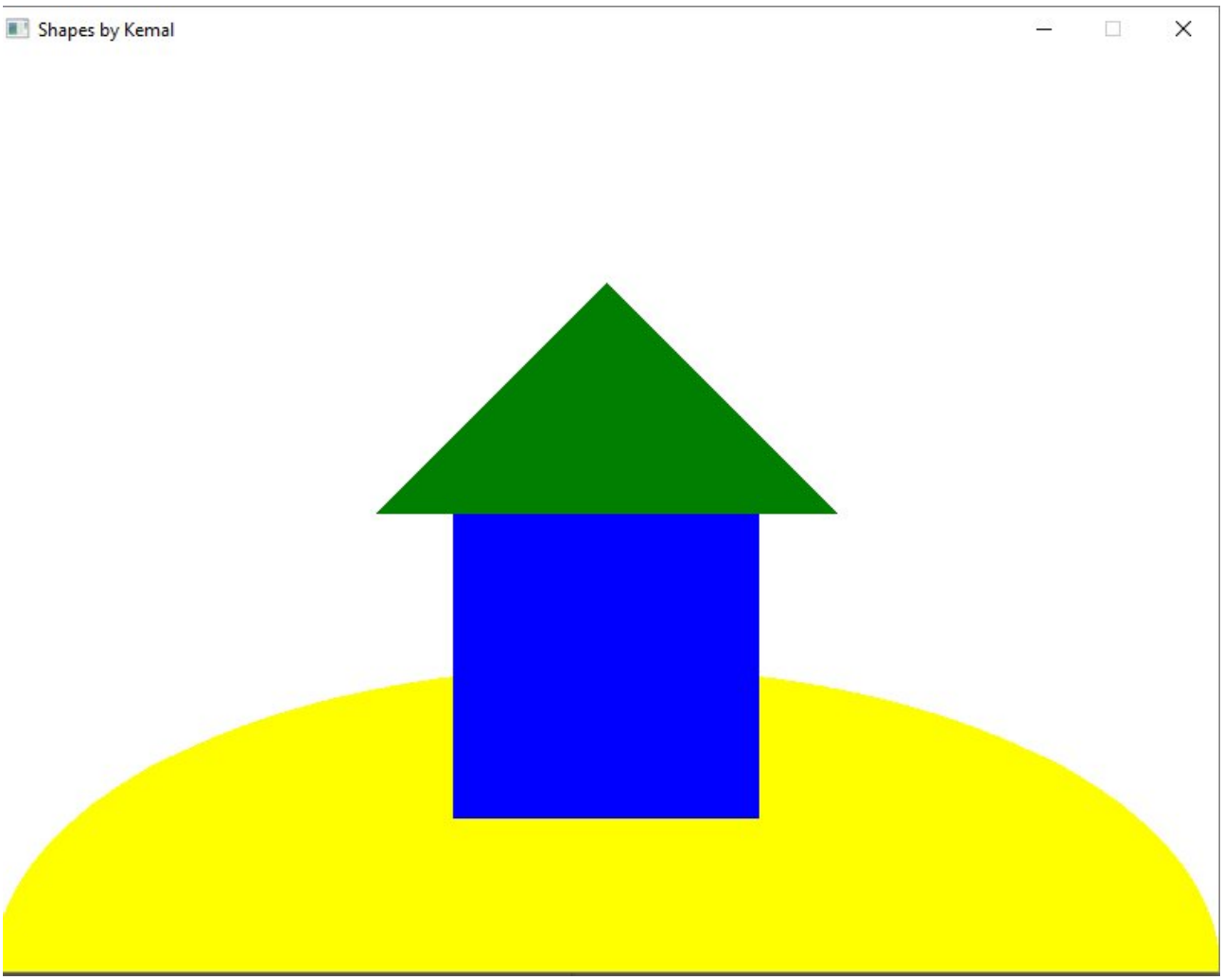
Tutor:
Mengmeng Ge

Outcome	Weight
Design	♦♦♦♦♦

This task makes design a thing with a c# programming language. And very useful

July 8, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        Window shapesWindow;
        shapesWindow = new Window("Shapes by Kemal", 800,600);

        shapesWindow.Clear(Color.White);
        shapesWindow.FillEllipse(Color.BrightGreen, 0,400,800,400);
        shapesWindow.FillRectangle(Color.Gray, 300, 300, 200, 200);
        shapesWindow.FillTriangle(Color.Red, 250, 300, 400, 150, 550, 300);

        shapesWindow.Refresh();

        SplashKit.Delay (5000);


        shapesWindow = new Window("Shapes by Kemal", 800,600);
        shapesWindow.Clear(Color.White);
        shapesWindow.FillEllipse(Color.Yellow, 0,400,800,400);
        shapesWindow.FillRectangle(Color.Blue, 300, 300, 200, 200);
        shapesWindow.FillTriangle(Color.Green, 250, 300, 400, 150, 550, 300);


        shapesWindow.Refresh();

        SplashKit.Delay (5000);

    }
}
```


Shapes by Kemal

6 How Many Objects?

Read the supplied code samples and let us know how many objects are created in each.

Outcome	Weight
Evaluate Code	♦♦♦♦◊

Well this task all about understanding C# programming language

Date	Author	Comment
2019/07/09 14:25	Achmad Kemal	This task is very useful for me to analyst
2019/07/09 14:26	Achmad Kemal	hey Dr Mengmeng Ge
2019/07/09 14:26	Achmad Kemal	This is Cam
2019/07/09 14:26	Achmad Kemal	I'm your new student
2019/07/09 14:27	Achmad Kemal	If you don't mind, can you check my word
2019/07/09 14:27	Achmad Kemal	i need your feedback
2019/07/09 14:27	Achmad Kemal	thank you
2019/07/09 16:12	Mengmeng Ge	Please check the following comments for your questions with issues:Question 2: Clear method is used to change the color of the window. You need to fill in the color after the window is cleared.Question 3: you have included Bitmap and SoundEffect. Both of them are Class type.Question 4: yetAnotherWindow is a variable referring to the "Hello World" Window object as well. So anything code including yetAnotherWindow should also be included.Question 5: the second and third lines of code are used to draw the bitmap on the window thus should be deleted.Question 6: the first line is to create a new Bitmap object. You could refer to the lines of "declare a variable" and "copy reference" to see how to complete this question.
2019/07/10 13:32	Achmad Kemal	Hey Dr Mengmeng Ge
2019/07/10 13:32	Achmad Kemal	i already resubmit my work
2019/07/10 13:33	Achmad Kemal	i hope this one is correct
2019/07/10 19:54	Mengmeng Ge	Question 6 is still not correct. I didn't mean copying those lines because they are used to declare variables of Window type and copy the reference for Window object. You will need to do similar thing for the Bitmap.
2019/07/12 10:39	Achmad Kemal	Dr Mengmeng ge
2019/07/12 10:40	Achmad Kemal	i resubmit my work again
2019/07/12 10:40	Achmad Kemal	can you check my work?
2019/07/12 10:40	Achmad Kemal	thanks
2019/07/12 11:24	Mengmeng Ge	You need to swap second line with third line of the code because you need to declare hello variable, create the Bitmap object, assign it to hello variable first and then copy the reference. I suggest you to type the code in the Visual Studio Code to see what will happen with different sequences of the code to have a thorough understanding.
2019/07/12 13:28	Achmad Kemal	Thank you for your explanation
2019/07/12 13:29	Achmad Kemal	I try code in the visual studio
2019/07/12 13:29	Achmad Kemal	Now i understand this task
2019/07/12 13:29	Achmad Kemal	i hope my task 1.3 work is done
2019/07/12 14:39	Mengmeng Ge	Great! Keep up with the good work!

Deakin University

Object Oriented Development

OnTrack Submission

How Many Objects?

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/12 13:28

Tutor:
Mengmeng Ge

Outcome	**Weight**
Evaluate Code	♦♦♦♦◊

Well this task all about understanding C# programming language

July 12, 2019

Answers for 1.3P How Many Objects?

Student Name: Achmad Mustafa Kemal
Student ID: 219374683

Question 1: How many of each kind of objects are created in this code?

Class	Number of Objects
Window	2
Bitmap	1
Sound Effect	1
Font	0

Question 2: What are the details of the different windows? Complete the following table. For Color Shown, indicate the color that the window was cleared to.

Window Title	Width	Height	Color Shown
“Hello World”	800	600	Blue
“Another Window”	300	300	Green

Question 3: How are the variables and Window objects connected? Which variables refer to which objects?

Window Title	Number of Variables that Refer to this Object?	Variable Names (comma separate if multiple)
“Hello World”	2	helloWindow, yetAnotherWindow
“Another Window”	1	anotherWindow

Question 4: How many times is the Window object with the title “Hello World” told to do something? Copy in the lines of code that get this Window object to do something.

```
helloWindow = new Window(“Hello World”,  800, 600);

yetAnotherWindow = helloWindow;

helloWindow.MoveTo(0, 0);


yetAnotherWindow.Clear(Color.Blue);
```

```
yetAnotherWindow.Refresh(60);

helloWindow.DrawBitmap(pegasi, 10, 50);
helloWindow.Refresh(60);
```

Question 5: How could you create another Bitmap object? One that loads a “Hello.png” image?

```
Bitmap hello = new Bitmap(“Hello”, “Hello.png”);
```

Question 6: How could you create another variable that will also refer to the “Hello.png” image you loaded in Question 5?

```
//Declare variable

Bitmap anotherHello;

// Work with image

Bitmap hello = new Bitmap(“hello”, “hello.png”);

//reference

anotherHello = hello;
```

7 Hello User

Create your own version of the Hello User program with some additional requirements.

Outcome	Weight
Build Programs	♦♦♦♦◊

Deakin University

Object Oriented Development

OnTrack Submission

Hello User

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/15 16:26

Tutor:
Mengmeng Ge

July 15, 2019

```
using System;
using SplashKitSDK;

public class HelloUser
{
    public static void Main()
    {
        string name;
        string inputText;
        int heightInCM;
        double weightInKG;
        double heightInMeters;
        double bmi;

        Console.Write("Enter your name: ");
        name = Console.ReadLine();
        Console.Write("Hello " + name);

        Console.Write("Enter your Height:");
        inputText = Console.ReadLine();
        heightInCM = Convert.ToInt32(inputText);
        heightInMeters = heightInCM / 100.0 ;
        Console.Write("Your Heigth is" + heightInMeters);


        Console.Write("Enter your Weight:");
        inputText = Console.ReadLine();
        weightInKG = Convert.ToDouble(inputText);
        Console.Write("Your Weigth is" + weightInKG);

        bmi = weightInKG/(heightInMeters*heightInMeters);
        Console.WriteLine("Your BMI is" + bmi);









    }
}
```

```
HelloUser.cs(10,16): warning CS0168: The variable 'weightInKG' is declared but n
ever used [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
HelloUser.cs(11,16): warning CS0168: The variable 'heightInMeters' is declared b
ut never used [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
HelloUser.cs(12,16): warning CS0168: The variable 'bmi' is declared but never us
ed [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
Enter your name:Kemal
HelloKemalEnter your Height:123
Your Heigth is123
Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
HelloUser.cs(10,16): warning CS0168: The variable 'weightInKG' is declared but n
ever used [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
HelloUser.cs(11,16): warning CS0168: The variable 'heightInMeters' is declared b
ut never used [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
HelloUser.cs(12,16): warning CS0168: The variable 'bmi' is declared but never us
ed [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
Enter your name: Kemal
Hello KemalEnter your Height:123
Your Heigth is123
Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
HelloUser.cs(10,16): warning CS0168: The variable 'weightInKG' is declared but n
ever used [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
HelloUser.cs(12,16): warning CS0168: The variable 'bmi' is declared but never us
ed [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
Enter your name: Kemal
Hello KemalEnter your Height:123
Your Heigth is1.23
Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
HelloUser.cs(13,16): warning CS0168: The variable 'bmi' is declared but never us
ed [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
Enter your name: Kemal
Hello KemalEnter your Height:123
Your Heigth is1.23Enter your Weight:76
Your Weigth is76
Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
Enter your name: Kemal
Hello KemalEnter your Height:183
Your Heigth is1.83Enter your Weight:77
Your Weigth is770.0237662337662338

Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
HelloUser.cs(13,16): warning CS0168: The variable 'bmi' is declared but never us
ed [D:\users\kemal\documents\code\HelloUser\HelloUser.csproj]
Enter your name: Kemal
Hello KemalEnter your Height:183
Your Heigth is1.83Enter your Weight:77
Your Weigth is77
Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
Enter your name: Kemal
Hello KemalEnter your Height:183
Your Heigth is1.83Enter your Weight:77
Your Weigth is77Your BMI is0.0237662337662338

Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
Enter your name: Kemal
Hello KemalEnter your Height:123
Your Heigth is1.23Enter your Weight:55
Your Weigth is55Your BMI is36.3540220768061

Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$ skm dotnet run
Enter your name: Kemal
Hello KemalEnter your Height:183
Your Heigth is1.83Enter your Weight:78
Your Weigth is78Your BMI is23.291229956105

Kemal@MSI MINGW64 /d/users/kemal/documents/code/HelloUser
$
```

8 The Stock class

Create a custom class to represent a product in a stock management system.

Outcome	Weight
Principles	♦♦♦♦◊

This explain about Private, Method, Property in the C#

Outcome	Weight
Build Programs	♦♦♦◊◊

This explain about Private, Method, Property in the C#

Date	Author	Comment
2019/07/15 18:26	Achmad Kemal	Give me some feedback
2019/07/15 20:36	Mengmeng Ge	The code looks good!
2019/07/17 08:50	Mengmeng Ge	I double checked this task. You will need to add RemoveStock method by yourself. Please add this method within Stock class and re-submit the task.
2019/07/17 16:30	Achmad Kemal	Thank you for your reminder
2019/07/17 16:31	Achmad Kemal	i already resubmit the code
2019/07/17 16:31	Achmad Kemal	i hope this one is correct
2019/07/18 11:24	Mengmeng Ge	Excellent!
2019/07/18 15:27	Achmad Kemal	thanks

Deakin University

Object Oriented Development

OnTrack Submission

The Stock class

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/17 16:30

Tutor:
Mengmeng Ge

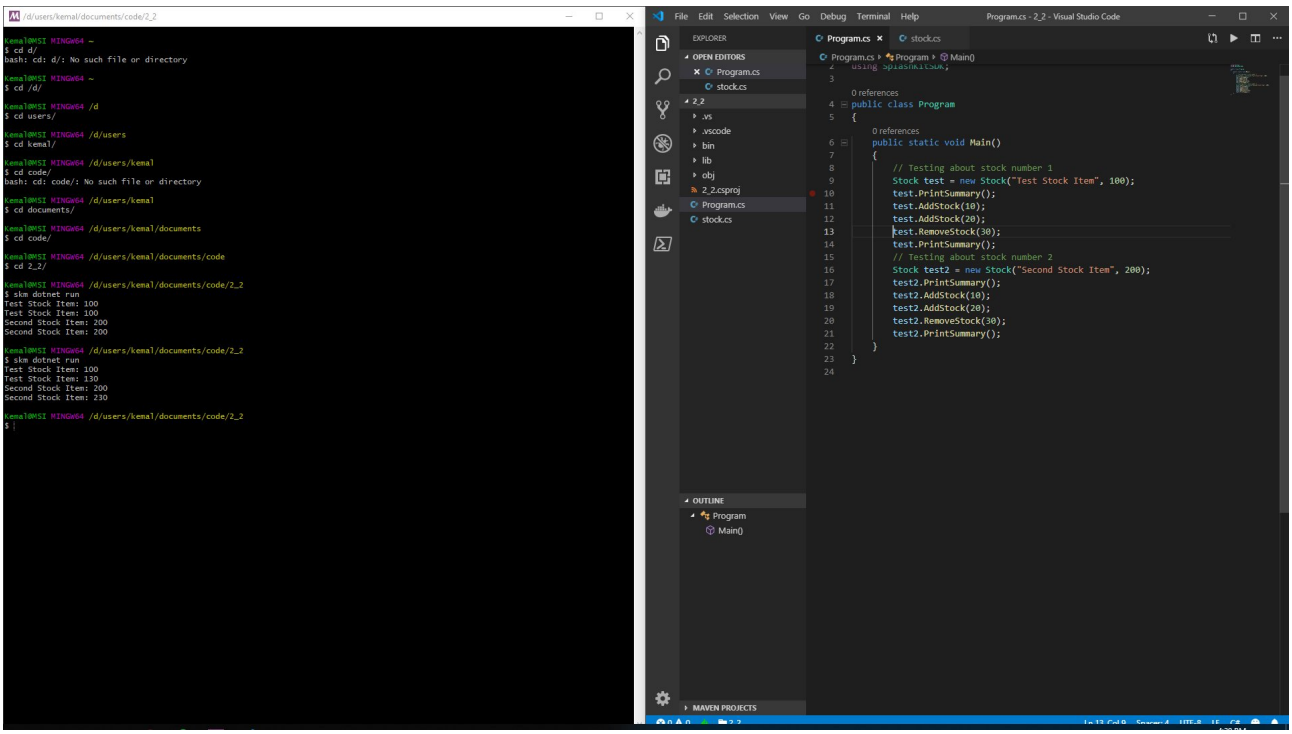
Outcome	**Weight**
Principles	♦♦♦♦◊
Build Programs	♦♦♦◊◊

This explain about Private, Method, Property in the C#

July 17, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        // Testing about stock number 1
        Stock test = new Stock("Test Stock Item", 100);
        test.PrintSummary();
        test.AddStock(10);
        test.AddStock(20);
        test.RemoveStock(30);
        test.PrintSummary();
        // Testing about stock number 2
        Stock test2 = new Stock("Second Stock Item", 200);
        test2.PrintSummary();
        test2.AddStock(10);
        test2.AddStock(20);
        test2.RemoveStock(30);
        test2.PrintSummary();
    }
}
```

```
using System;


public class Stock
{
    private int _quantityOnHand;
    private string _name;
    // Constructor
    public Stock(string name, int initialLevel)
    {
        _name = name;
        _quantityOnHand = initialLevel;
    }

    //Method to add stock
    public void AddStock(int quantityAdded)
    {
        _quantityOnHand = _quantityOnHand + quantityAdded;
    }
    public void RemoveStock(int quantityAdded)
    {
        _quantityOnHand = _quantityOnHand - quantityAdded;
    }
    //Property
    public string Name
    {
        get {return _name;} // "Get" is for read
        set{_name = value;} // "Set" is for write
    }

    //Property
    public int QuantityOnHand
    {
        get { return _quantityOnHand;}
        private set {_quantityOnHand = value;}
    }
   public void PrintSummary()
   {
       Console.WriteLine($"{Name}: {QuantityOnHand}");
   }


}
```


/d/users/kemal/documents/code/2_2
Kemal@MSI MINGW64 ~
$ cd d/
bash: cd: d/: No such file or directory
Kemal@MSI MINGW64 ~
$ cd /d/
Kemal@MSI MINGW64 /d
$ cd users/
Kemal@MSI MINGW64 /d/users
$ cd kemal/
Kemal@MSI MINGW64 /d/users/kemal
$ cd code/
bash: cd: code/: No such file or directory
Kemal@MSI MINGW64 /d/users/kemal
$ cd documents/
Kemal@MSI MINGW64 /d/users/kemal/documents
$ cd code/
Kemal@MSI MINGW64 /d/users/kemal/documents/code
$ cd 2_2/
Kemal@MSI MINGW64 /d/users/kemal/documents/code/2_2
$ skm dotnet run
Test Stock Item: 100
Test Stock Item: 100
Second Stock Item: 200
Second Stock Item: 200
Kemal@MSI MINGW64 /d/users/kemal/documents/code/2_2
$ skm dotnet run
Test Stock Item: 100
Test Stock Item: 130
Second Stock Item: 200
Second Stock Item: 230
Kemal@MSI MINGW64 /d/users/kemal/documents/code/2_2
$
File Edit Selection View Go Debug Terminal Help
Program.cs - 2_2 - Visual Studio Code
EXPLORER
OPEN EDITORS
Program.cs
stock.cs
2_2
.vs
.vscode
bin
lib
obj
2_2.csproj
Program.cs
stock.cs
OUTLINE
Program
Main()
MAVEN PROJECTS
Program.cs
stock.cs
Program.cs > Program > Main()
using SplashKitSDK;
0 references
public class Program
{
0 references
public static void Main()
{
// Testing about stock number 1
Stock test = new Stock("Test Stock Item", 100);
test.PrintSummary();
test.AddStock(10);
test.AddStock(20);
test.RemoveStock(30);
test.PrintSummary();
// Testing about stock number 2
Stock test2 = new Stock("Second Stock Item", 200);
test2.PrintSummary();
test2.AddStock(10);
test2.AddStock(20);
test2.RemoveStock(30);
test2.PrintSummary();
}
}

9 Making A Scene

Using all you have learnt so far, create a scene/comic/story using shape drawing in SplashKit.

Outcome	Weight
Build Programs	♦♦♦♦◊

Task 1.4 is all about design and code building

Outcome	Weight
Design	♦♦♦♦◊

Task 1.4 is all about design and code building

Date	Author	Comment
2019/07/09 14:33	Achmad Kemal	Good Task
2019/07/09 16:13	Mengmeng Ge	You could record your program and upload it onto YouTube and provide the link of YouTube.
2019/07/09 16:14	Mengmeng Ge	Or you could demonstrate it in next Monday's practical.
2019/07/09 18:26	Achmad Kemal	where I should put youtube link ?
2019/07/09 18:26	Achmad Kemal	in this comment?
2019/07/10 08:35	Mengmeng Ge	Yes. You could copy the link here.
2019/07/11 11:43	Achmad Kemal	https://youtu.be/We1PscrEQA8
2019/07/11 12:03	Achmad Kemal	Here is my youtube link for task 1.4
2019/07/27 14:06	Achmad Kemal	Hey i just got similarities
2019/07/27 14:06	Achmad Kemal	i don't know get this one
2019/07/28 17:30	Mengmeng Ge	I will need to discuss this with you on Monday.
2019/08/01 09:03	Achmad Kemal	i just wondering why i still get similarities detected
2019/08/01 11:25	Mengmeng Ge	Because another student who had similar code with you didn't change his submission.
2019/08/01 11:38	Achmad Kemal	okay thank you

Deakin University

Object Oriented Development

OnTrack Submission

Making A Scene

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/09 14:33

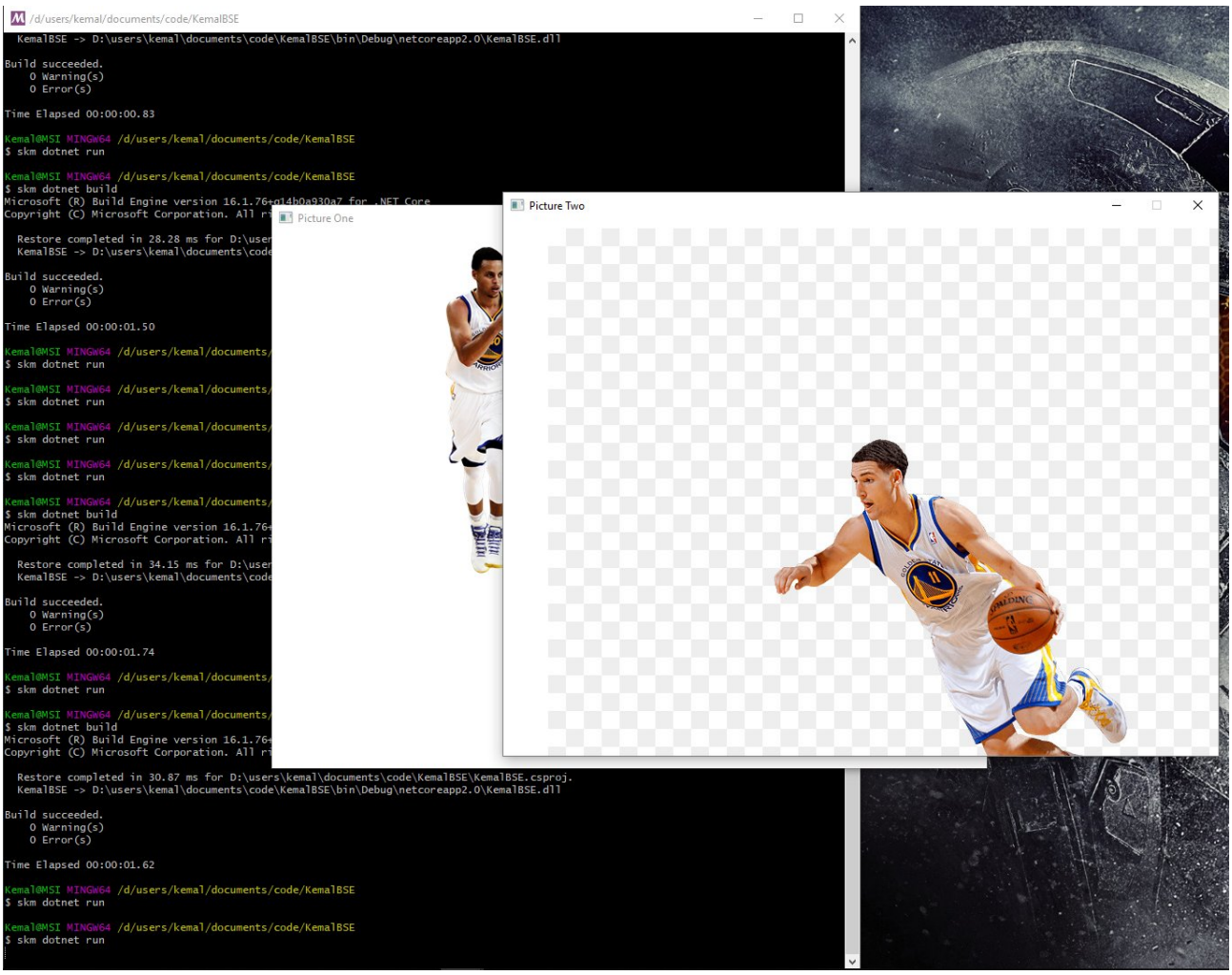
Tutor:
Mengmeng Ge

Outcome	**Weight**
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊

Task 1.4 is all about design and code building

July 9, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        Window FirstWindow;
        Window SecondWindow;

        FirstWindow = new Window("Picture One", 800, 600);

        Bitmap curry = new Bitmap("Curry", "curry.png");
        FirstWindow.DrawBitmap(curry, 50, 10);

        FirstWindow.Refresh();

        SoundEffect cs = new SoundEffect("cs", "cs.wav");
        cs.Play();


        SplashKit.Delay(5000);

        SecondWindow = new Window("Picture Two", 800, 600);

        Bitmap klay = new Bitmap("Klay", "klay.png");
        SecondWindow.DrawBitmap(klay, 50, 10);

        SecondWindow.Refresh();

        SplashKit.Delay(5000);









    }
}
```


/d/users/kemal/documents/code/KemalBSE
KemalBSE -> D:\users\kemal\documents\code\KemalBSE\bin\Debug\netcoreapp2.0\KemalBSE.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:00.83
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 28.28 ms for D:\users\kemal\documents\code\KemalBSE\KemalBSE.csproj.
KemalBSE -> D:\users\kemal\documents\code\KemalBSE\bin\Debug\netcoreapp2.0\KemalBSE.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:01.50
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 34.15 ms for D:\users\kemal\documents\code\KemalBSE\KemalBSE.csproj.
KemalBSE -> D:\users\kemal\documents\code\KemalBSE\bin\Debug\netcoreapp2.0\KemalBSE.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:01.74
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 30.87 ms for D:\users\kemal\documents\code\KemalBSE\KemalBSE.csproj.
KemalBSE -> D:\users\kemal\documents\code\KemalBSE\bin\Debug\netcoreapp2.0\KemalBSE.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:01.62
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/KemalBSE
$ skm dotnet run
Picture One
Picture Two

10 Name Tester

Create a simple name testing program.

Outcome	Weight
Principles	♦♦♦♦◊

This task about whille loop and try catch loop function

Date	Author	Comment
2019/07/29 16:21	Achmad Kemal	Ready to Mark
2019/07/29 16:21	Achmad Kemal	Good taskThank you for teaching me about while loop function
2019/07/29 16:48	Mengmeng Ge	Complete
2019/07/30 18:59	Mengmeng Ge	In main of Program class, your Testname() method should be put under case MenuOption.testname. Please resubmit and take the screenshot.
2019/07/30 18:59	Mengmeng Ge	Fix and Resubmit
2019/07/30 20:23	Achmad Kemal	Ready to Mark
2019/07/31 11:08	Mengmeng Ge	The try-catch block in ReadUserOption is incorrect. You will need to put the code of reading and converting user input inside try. Please correct and resubmit it.
2019/07/31 11:08	Mengmeng Ge	Fix and Resubmit
2019/07/31 18:47	Achmad Kemal	Ready to Mark
2019/08/01 11:26	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Name Tester

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/31 18:47

Tutor:
Mengmeng Ge

Outcome	**Weight**
Principles	♦♦♦♦◊

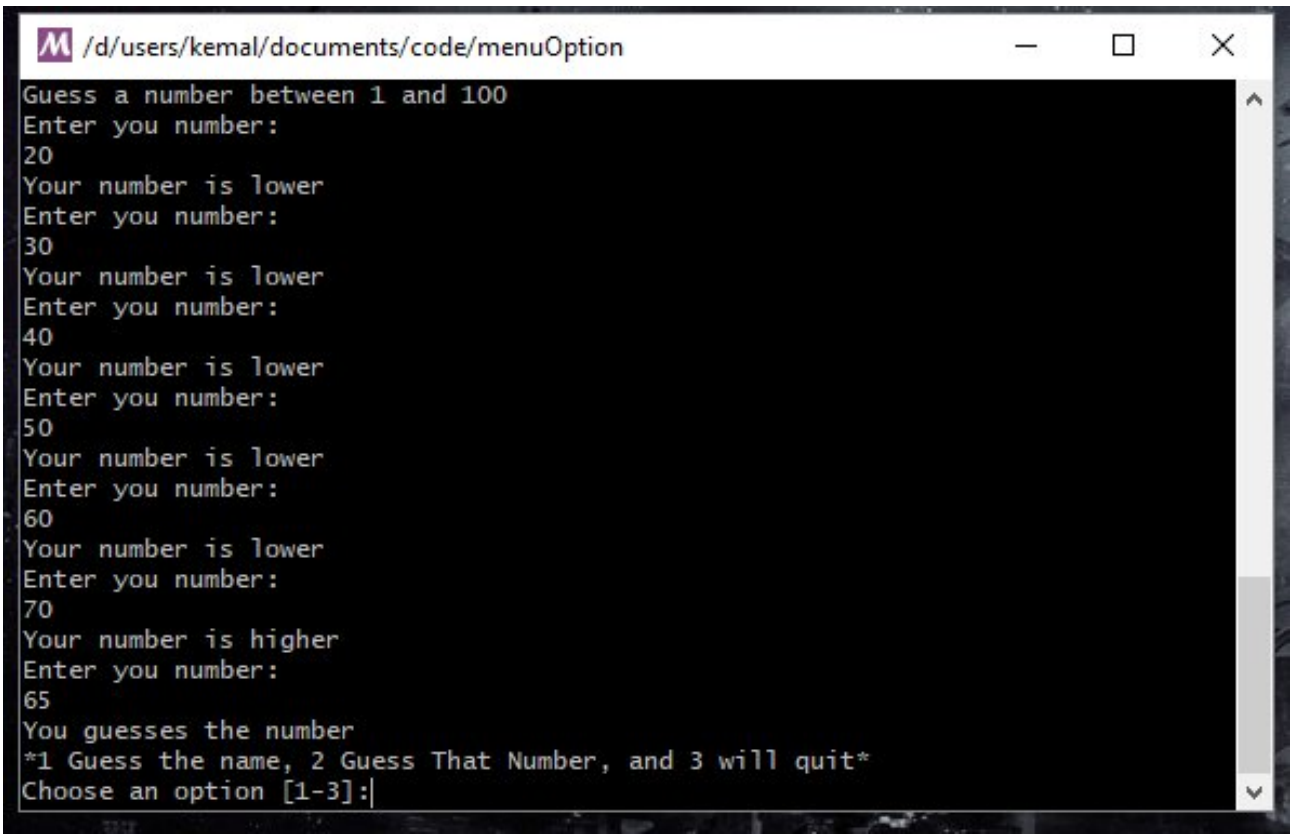
This task about whille loop and try catch loop function

July 31, 2019

```
using System;
using SplashKitSDK;

public class Program
{
     private static MenuOption ReadUserOption()
    {
        Console.WriteLine("*1 Guess the name, 2 Guess That Number, and 3 will
        ↪  quit*");
        int option;
            do
            {
            string inputText;
            try
            {
            Console.Write("Choose an option [1-3]:");
            inputText = Console.ReadLine();
            option = Convert.ToInt32(inputText);
            }
            catch
            {
            // output message
            option = -1;
            }
            }while(option < 1 || option >3);
            return (MenuOption)(option - 1);



    }


    public static void Main()
    {
        MenuOption userSelection;
        userSelection = ReadUserOption();
        Console.WriteLine(userSelection);

            do
            {
                userSelection = ReadUserOption();
                switch(userSelection)
                {
                    case MenuOption.Testname:
                    Testname();
                    break;

                    case MenuOption.GuessThatNumber:
                    GuessThatNumber();
                    break;
                }
            }while (userSelection != MenuOption.Quit);

```

```
53
54      }
55      public static void Testname()
56
57      {
58          string name;
59          Console.WriteLine("Hello");
60          name = Console.ReadLine();
61          Console.WriteLine("Hello" + name);
62
63          if (name == "Kemal")
64          {
65              Console.WriteLine ("Welcome my creator");
66          }
67          else if (name.ToLower() == "Cam")
68          {
69              Console.WriteLine("Don't worry you got this");
70          }
71          else
72          {
73              Console.WriteLine("Sorry bro try again");
74          }
75
76      }
77          private static int ReadGuess (int min, int max)
78          {
79
80              int number;
81              do
82              {
83              string inputText1;
84              try{
85              Console.WriteLine("Enter you number:");
86              inputText1 = Console.ReadLine();
87              number = Convert.ToInt32(inputText1);}
88              catch{
89                      number = -1;
90              }
91              }while(number < min || number > max);
92
93             return number ;
94
95          }
96
97          public static void GuessThatNumber()
98          {
99             int guess;
100            int min = 1;
101            int max  = 100;
102
103
104            int target = new Random().Next(100)+1;
105            Console.WriteLine("Guess a number between 1 and 100");
```

```
106             guess = ReadGuess(min,max);
107             while (guess != target)
108             {
109 
110                 if (guess < target)
111                 {
112 
113                     Console.WriteLine("Your number is lower" );
114                     min = guess;
115                 }
116                 else if (guess > target)
117                 {
118                     Console.WriteLine("Your number is higher");
119                     max = guess;
120                 }
121                 guess = ReadGuess(min, max);
122             }
123 
124                 Console.WriteLine("You guesses the number");
125 
126         }
127 }
```


/d/users/kemal/documents/code/menuOption
Guess a number between 1 and 100
Enter you number:
20
Your number is lower
Enter you number:
30
Your number is lower
Enter you number:
40
Your number is lower
Enter you number:
50
Your number is lower
Enter you number:
60
Your number is lower
Enter you number:
70
Your number is higher
Enter you number:
65
You guesses the number
1 Guess the name, 2 Guess That Number, and 3 will quit
Choose an option [1-3]:

11 Validating Stock Actions

Add validations to the program to make sure things are used correctly.

Outcome	Weight
Principles	♦♦♦♦◊

This task about validation function

Outcome	Weight
Build Programs	♦♦♦♦◊

This task about validation function

Outcome	Weight
Design	♦♦♦♦◊

This task about validation function

Outcome	Weight
Justify	♦♦♦♦◊

This task about validation function

Date	Author	Comment
2019/07/29 21:13	Achmad Kemal	Ready to Mark
2019/07/29 21:13	Achmad Kemal	very challenging task
2019/07/29 21:36	Mengmeng Ge	Try-catch block in ReadUserOption is not correct. Please refer to a similar code in NameTester program for where to add this.
2019/07/29 21:36	Mengmeng Ge	Fix and Resubmit
2019/07/29 22:00	Achmad Kemal	Ready to Mark
2019/07/30 08:26	Mengmeng Ge	You didn't fix the problem. Please check ReadUserOption carefully and put try-catch block in the correct place.
2019/07/30 08:27	Mengmeng Ge	Fix and Resubmit
2019/07/30 08:28	Mengmeng Ge	Reading user input and converting to integer should be put inside try because that's where your code may go wrong. You need to look at lecture notes to fully understand exception handling.
2019/07/30 09:32	Achmad Kemal	I will take a look lecture note and then resubmit
2019/07/30 11:09	Achmad Kemal	Ready to Mark
2019/07/30 18:55	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Validating Stock Actions

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/30 11:09

Tutor:
Mengmeng Ge

Outcome	Weight
Principles	♦♦♦♦◊
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊
Justify	♦♦♦♦◊

This task about validation function

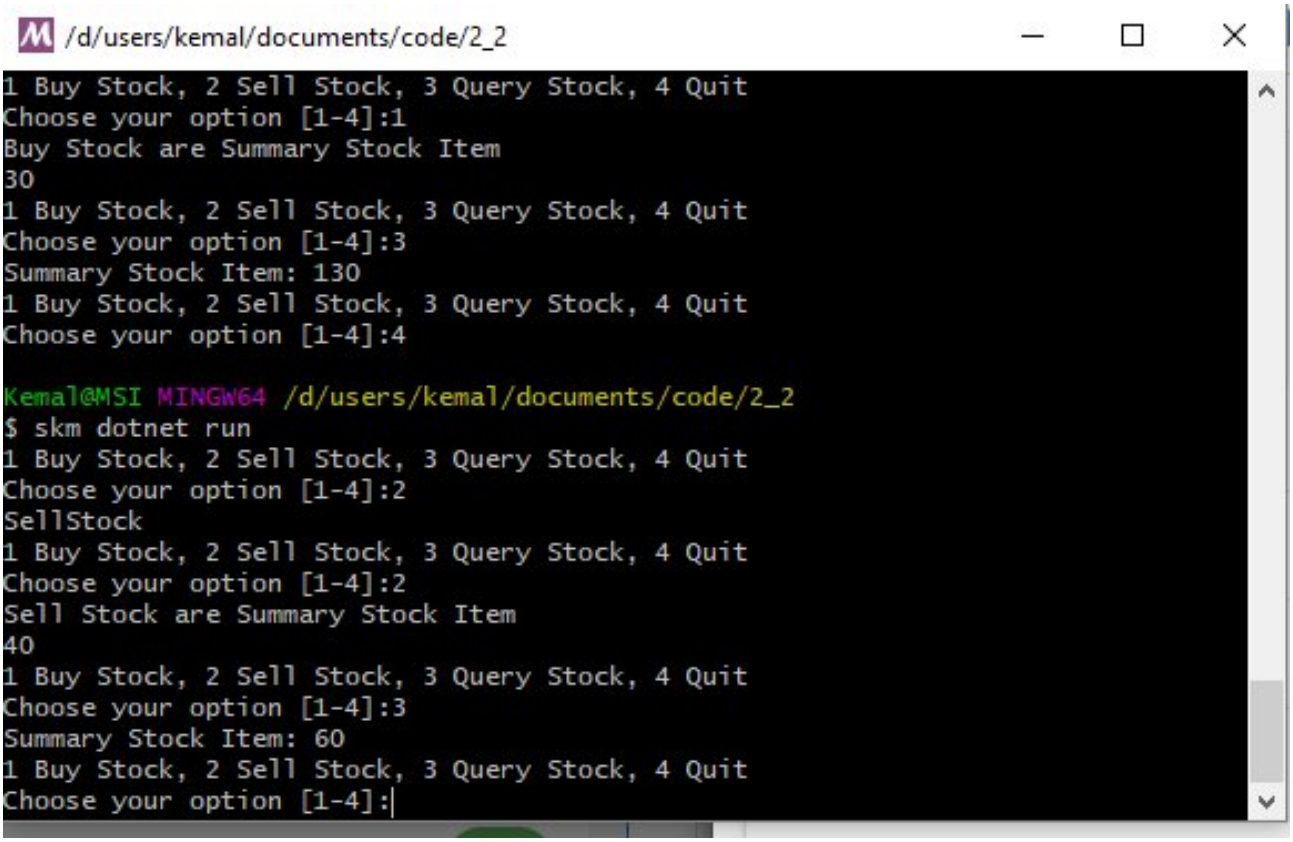
July 30, 2019

```
using System;
using SplashKitSDK;

public class Program
{

    private static MenuOption ReadUserOption()
    {
        Console.WriteLine("1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit  ");
        int option;
        do
        {
            try
        {
           string inputText;
            Console.Write("Choose your option [1-4]:");
            inputText = Console.ReadLine();
            option = Convert.ToInt32(inputText);
        }
            catch
        {
            //output message
            option = -1;
        }

        }while(option < 1 || option >4);
        return (MenuOption)(option -1);
    }
    public static void Main()
    {
        MenuOption userSelection;
        userSelection = ReadUserOption();
        Console.WriteLine(userSelection);
         Stock test = new Stock("Summary Stock Item", 100);
        do
        {
            userSelection = ReadUserOption();
            switch(userSelection)
            {
                case MenuOption.BuyStock:
                    PerformBuyStock(test);
                    break;
                case MenuOption.SellStock:
                    PerformSellStock(test);
                    break;
                case MenuOption.QueryStock:
                    PerformQueryStock(test);
                    break;
            }
        } while (userSelection != MenuOption.Quit);


```

```
54
55      }
56          private static void PerformBuyStock(Stock stock)
57          {
58
59               int quantity;
60
61              Console.WriteLine($"Enter your number {stock.Name}");
62              quantity = Convert.ToInt32(Console.ReadLine());
63
64              stock.AddStock(quantity);
65
66          }
67
68          private static void PerformSellStock(Stock stock)
69          {
70              int quantity;
71
72              Console.WriteLine($"Enter your number {stock.Name}");
73              quantity = Convert.ToInt32(Console.ReadLine());
74
75              stock.RemoveStock(quantity);
76
77          }
78          private static void PerformQueryStock(Stock stock)
79          {
80              stock.PrintSummary();
81          }
82  }
```

```
using System;


public class Stock
{
    private int _quantityOnHand;
    private string _name;
    // Constructor
    public Stock(string name, int initialLevel)
    {
        _name = name;
        _quantityOnHand = initialLevel;
    }

    //Method to add stock
   public bool AddStock(int quantityAdded)
   {
       if (quantityAdded > 0)
       {
           _quantityOnHand = _quantityOnHand +quantityAdded;
           return true;
       }
       else
       {
           return false;
       }
   }
      public bool RemoveStock(int quantityAdded)
   {
       if (quantityAdded > 0 && quantityAdded < QuantityOnHand)
       {
           _quantityOnHand = _quantityOnHand - quantityAdded;
           return true;
       }
       else
       {
           return false;
       }
   }
    //Property
    public string Name
    {
        get {return _name;} // "Get" is for read
        set{_name = value;} // "Set" is for write
    }

    //Property
    public int QuantityOnHand
    {
        get { return _quantityOnHand;}
        private set {_quantityOnHand = value;}
    }
   public void PrintSummary()
```

```
54      {
55          Console.WriteLine($"{Name}: {QuantityOnHand}");
56      }
57
58  }
```


/d/users/kemal/documents/code/2_2
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:1
Buy Stock are Summary Stock Item
30
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:3
Summary Stock Item: 130
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:4
Kemal@MSI MINGW64 /d/users/kemal/documents/code/2_2
$ skm dotnet run
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:2
SellStock
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:2
Sell Stock are Summary Stock Item
40
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:3
Summary Stock Item: 60
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:

12 Moving The Player

Get your robot character moving

Outcome	Weight
Principles	♦♦♦♦◊

Good task

Outcome	Weight
Build Programs	♦♦♦♦◊

Good task

Outcome	Weight
Design	♦♦♦♦◊

Good task

Date	Author	Comment
2019/07/30 13:55	Achmad Kemal	image comment
2019/07/30 13:58	Achmad Kemal	I have question about move player. I already write code about movement but the player don't want to move. Do you have any comment about my code?
2019/07/30 19:07	Mengmeng Ge	Your HandleInput method should be inside Player class as it is related to the player (move the player based on user input). Then you do not need to pass reference to the Player object.
2019/07/30 19:11	Mengmeng Ge	Besides, for the while loop in Program class, you should put draw and refresh method inside the loop. Then every time, you move the player, it will clear the window and redraw the player to the new position.
2019/08/01 13:34	Achmad Kemal	So i just put my handle input into player class?
2019/08/01 13:34	Achmad Kemal	so i dont need to edit my handle input
2019/08/01 13:46	Mengmeng Ge	Please read the instructions carefully on page 2 regarding HandleInput: you need to include four arrow keys, not just up key; you also need to check Escaped Key and set Quit to true.
2019/08/01 13:59	Achmad Kemal	Yeah i will do it that also
2019/08/01 13:59	Achmad Kemal	thank you
2019/08/01 19:33	Achmad Kemal	image comment
2019/08/01 19:33	Achmad Kemal	is normal to have shadow when i move the object?
2019/08/01 19:34	Achmad Kemal	this is screenshot when i move to the object
2019/08/02 00:01	Achmad Kemal	image comment
2019/08/02 00:01	Achmad Kemal	i'm not sure about this
2019/08/02 00:02	Achmad Kemal	are we using booelan splashkit.KeyTyped?
2019/08/02 00:03	Achmad Kemal	for quit section
2019/08/02 00:04	Achmad Kemal	but i can't put public void
2019/08/02 09:05	Mengmeng Ge	It is asked to create a Quit property. You could check the definition of property. public void is used to declare a method (void indicates the return type).
2019/08/02 09:36	Achmad Kemal	Thank you for your respond
2019/08/02 16:35	Achmad Kemal	image comment
2019/08/02 16:35	Achmad Kemal	this is my StayOnWindow method
2019/08/02 16:36	Achmad Kemal	it works fine but the object will stay on window when i just move up and left
2019/08/02 16:37	Achmad Kemal	do you have any comment about my code
2019/08/02 16:37	Achmad Kemal	thank you
2019/08/02 17:13	Achmad Kemal	I need your help with my StayOnWindow method because I'm not sure if I doing exactly the same with the instructor
2019/08/02 17:32	Mengmeng Ge	You could use height and width of the window to restrict the player's movement. Besides, you will need to clear the window everytime you move the player otherwise you can see your player is drawn on the window everytime you move it.
2019/08/02 17:32	Mengmeng Ge	Fix and Resubmit
2019/08/02 21:30	Achmad Kemal	Ready to Mark
2019/08/03 19:49	Mengmeng Ge	In HandleInput method, you could use X-=speed instead of declaring speed1 = -5. In StayOnWindow method, you do not need to declare a local variable wwidth. You could just use gameWindow.Width in the condition. Besides, you will need to consider player bitmap's width as well. When X = gameWindow.Width - Width, the player bitmap is ready at the edge of the window. Besides, please also check whether player will be at the bottom of the window (this is re-

Deakin University

Object Oriented Development

OnTrack Submission

Moving The Player

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/04 18:50

Tutor:
Mengmeng Ge

Outcome	Weight
Principles	♦♦♦♦◊
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊

Good task

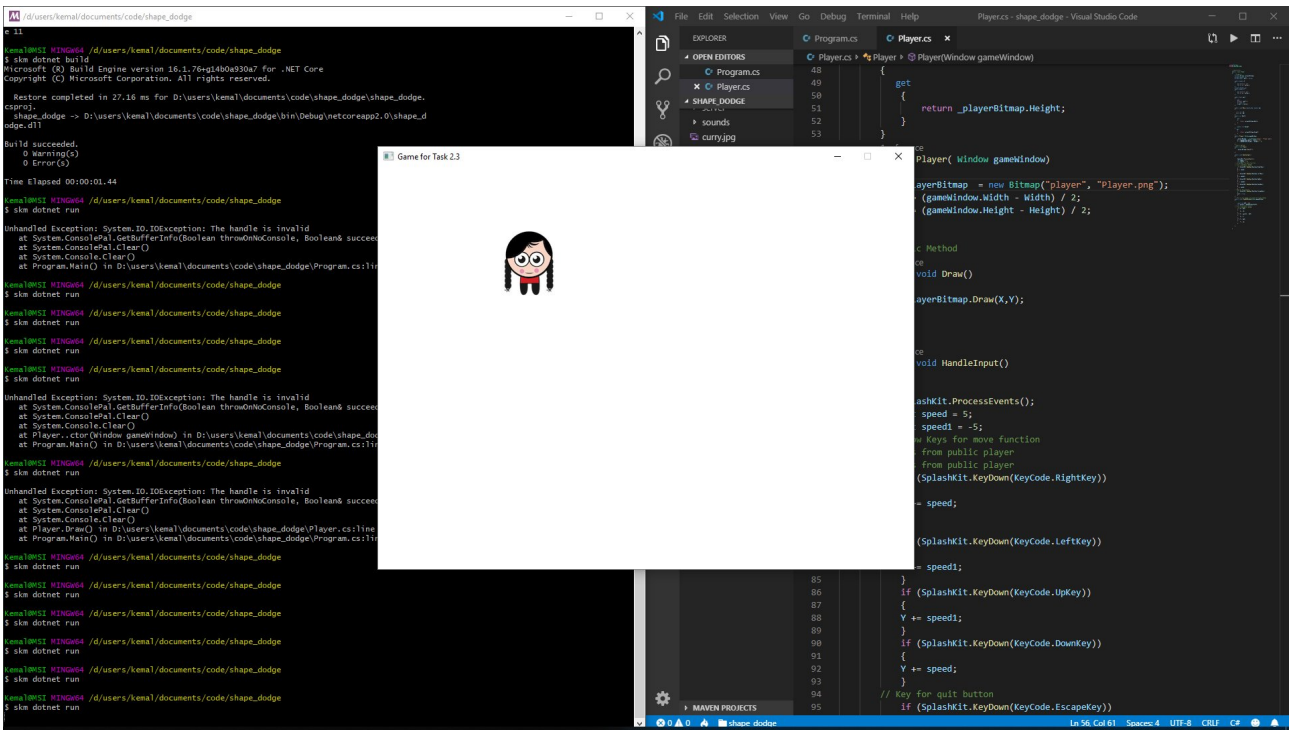
August 4, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        Window gameWindow;

        gameWindow = new Window("Game for Task 2.3", 800, 600);
        Player Player = new Player (gameWindow);
        do
        {
            Player.HandleInput();
            Player.StayOnWindow(gameWindow);
            Player.Draw();
            gameWindow.Refresh(60);
            gameWindow.Clear(Color.White);
        }while(!Player.quit);
    }
}
```

```
using System;
using SplashKitSDK;


    //Constructor
    public class Player
    {
        //Property
        private Bitmap _playerBitmap;
        private double _x;
        private bool _quit = false;

        public double X
        {
            get {return _x;}
            set { _x = value;}
        }
        private double _y;
        public double Y
        {
            get {return _y;}
            set { _y = value;}
        }

        public bool quit
        {
            get
            {return _quit;}
            private set
            {_quit = value;}
        }

        public void Move (double dx, double dy)
        {

          _x = _x + dx;
          _y = _y + dy;
        }
        public int Width
        {
            get
            {
                return _playerBitmap.Width;
            }
        }

            public int Height
        {
           get
            {
                return _playerBitmap.Height;
            }
        }
```

```
54          public Player( Window gameWindow)
55          {
56              _playerBitmap  = new Bitmap("player", "Player.png");
57              X = (gameWindow.Width - Width) / 2;
58              Y = (gameWindow.Height - Height) / 2;
59
60          }
61          //Public Method
62          public void Draw()
63          {
64              _playerBitmap.Draw(X,Y);
65
66          }
67
68          public void HandleInput()
69          {
70
71              SplashKit.ProcessEvents();
72              int speed = 5;
73          // Arrow Keys for move function
74          // X is from public player
75          // Y is from public player
76              if (SplashKit.KeyDown(KeyCode.RightKey))
77              {
78              X += speed;
79              }
80
81              if (SplashKit.KeyDown(KeyCode.LeftKey))
82              {
83              X -= speed;
84              }
85              if (SplashKit.KeyDown(KeyCode.UpKey))
86              {
87              Y -= speed;
88              }
89              if (SplashKit.KeyDown(KeyCode.DownKey))
90              {
91              Y += speed;
92              }
93          // Key for quit button
94              if (SplashKit.KeyDown(KeyCode.EscapeKey))
95              {
96              quit = true;
97              }
98          }
99          // This is for player no to go out from windows
100         public void StayOnWindow(Window gameWindow)
101         {
102
103             const int GAP = 10;
104         // X is from public player
105         // Y is from public player
106             if (X < GAP)
```

```
107             {
108                 X = GAP;
109             }
110             if (X > gameWindow.Width - Width - GAP)
111             {
112                 X = gameWindow.Width - Width - GAP;
113             }
114             if (Y < GAP)
115             {
116                 Y = GAP;
117             }
118             if (Y > gameWindow.Height - Height - GAP)
119             {
120                 Y = gameWindow.Height - Height - GAP;
121             }
122 
123 
124         }
125 
126     }
```


Game for Task 2.3
Player.cs - shape_dodge - Visual Studio Code
Program.cs
Player.cs
OPEN EDITORS
SHAPE_DODGE
sounds
curry.jpg
MAVEN PROJECTS

13 The Player Class

Create a player class

Outcome	Weight
Build Programs	♦♦♦♦◊

This task to give me knowledge about constructor and method in the c# programming

Outcome	Weight
Design	♦♦♦♦◊

This task to give me knowledge about constructor and method in the c# programming

Date	Author	Comment
2019/07/16 19:28	Achmad Kemal	Ready to Mark
2019/07/16 19:28	Achmad Kemal	Give me some feedback
2019/07/17 08:33	Mengmeng Ge	1. You will need to delete line 8 within Player class. gameWindow is not a field for Player.
2019/07/17 08:34	Mengmeng Ge	2. Player() is the constructor. You pass the Window gameWindow as a parameter in the constructor method: publich Payer(Window gameWindow)
2019/07/17 08:35	Mengmeng Ge	3. Comment for Draw method is not correct. It is NOT as constructor. It is a public method to define one behavior of the player. Please check the lecture notes for definition of constructor.
2019/07/17 08:41	Mengmeng Ge	4. In Draw method, you will need to delete line 50-54. You do not need to declare local variables. You also need to delete the parameter _playerBitmap. You have already declared the variable within the Player class. In line 57, you draw a rectangle. It is not what the task asks. You could replace that with: _playerBitmap.Draw(X, Y). Besides, in Program class, you need to delete line 12 and replace it with: player.Draw().
2019/07/17 08:42	Mengmeng Ge	The major problem you have is you do not use the fields you declare within the class. You need to understand the difference between local variables and instance variables. Please refer to the lecture notes or weekly resources.
2019/07/17 08:43	Mengmeng Ge	Please try to understand the code before you type them based on the instructions.
2019/07/17 08:43	Mengmeng Ge	Fix and Resubmit
2019/07/22 15:37	Achmad Kemal	Ready to Mark
2019/07/22 15:37	Achmad Kemal	pls give some feedback
2019/07/22 17:16	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

The Player Class

Submitted By:
Achmad Mustafa Kemal
akemal
2019/07/22 15:37

Tutor:
Mengmeng Ge

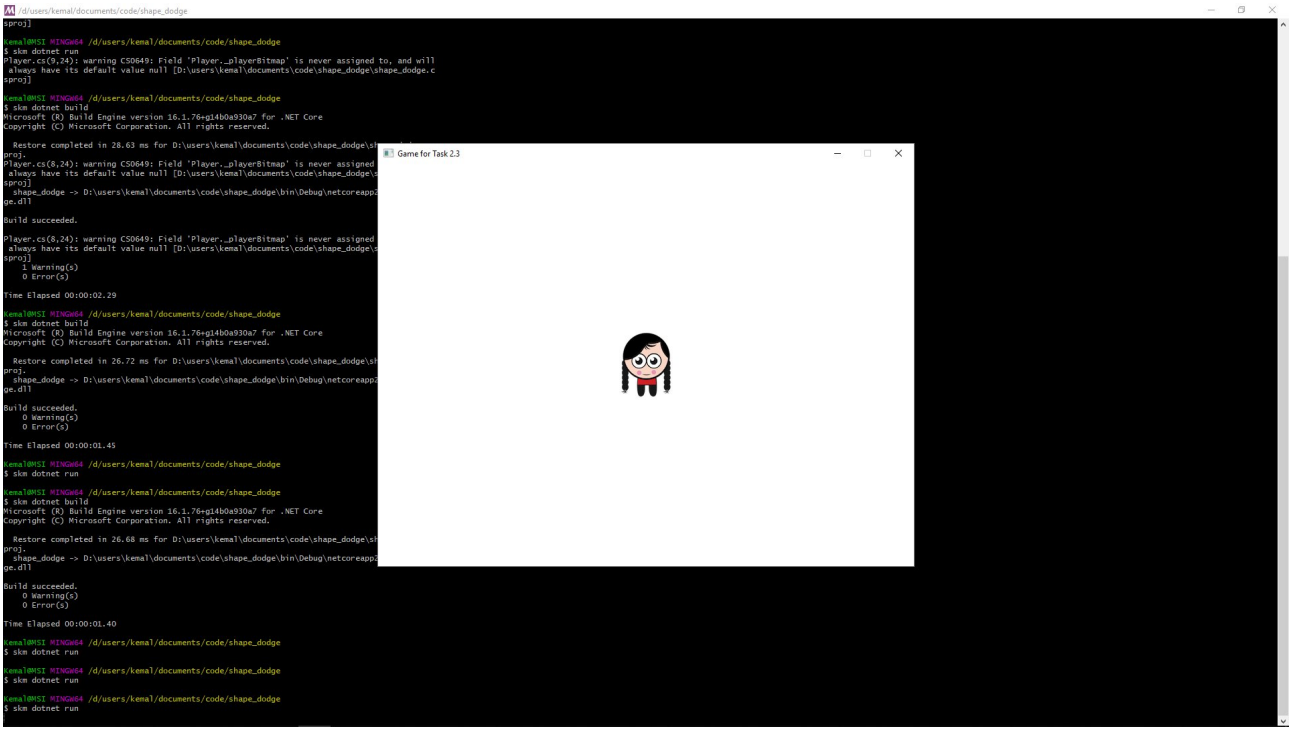
Outcome	Weight
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊

This task to give me knowledge about constructor and method in the c# programming

July 22, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        Window gameWindow;

        gameWindow = new Window("Game for Task 2.3", 800, 600);
        Player player = new Player (gameWindow);
        player.Draw();
        gameWindow.Refresh(60);
        SplashKit.Delay(5000);
    }
}
```

```
using System;
using SplashKitSDK;


    //Constructor
    public class Player
    {
        private Bitmap _playerBitmap;
        private double _x;
        public double X {get; private set;}
         private double _y;
        public double Y {get; private set;}


        public void Move (double dx, double dy)
        {
            _x = _x + dx;
            _y = _y + dy;
        }

        public int Width
        {
            get
            {
                return _playerBitmap.Width;
            }
        }

            public int Height
        {
           get
            {
                return _playerBitmap.Height;
            }
        }


        public Player( Window gameWindow)
        {
            _playerBitmap  = new Bitmap("player", "Player.png");
            X = (gameWindow.Width - Width) / 2;
            Y = (gameWindow.Height - Height) / 2;
        }


        //Public Method
        public void Draw()
        {
        _playerBitmap.Draw(X,Y);
        }

    }
```


/d/users/kemal/documents/code/shape_dodge
sproj]
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet run
Player.cs(9,24): warning CS0649: Field 'Player._playerBitmap' is never assigned to, and will
always have its default value null [D:\users\kemal\documents\code\shape_dodge\shape_dodge.c
sproj]
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 28.63 ms for D:\users\kemal\documents\code\shape_dodge\sh
proj.
Player.cs(8,24): warning CS0649: Field 'Player._playerBitmap' is never assigned
always have its default value null [D:\users\kemal\documents\code\shape_dodge\s
sproj]
shape_dodge -> D:\users\kemal\documents\code\shape_dodge\bin\Debug\netcoreapp2
ge.dll
Build succeeded.
Player.cs(8,24): warning CS0649: Field 'Player._playerBitmap' is never assigned
always have its default value null [D:\users\kemal\documents\code\shape_dodge\s
sproj]
1 Warning(s)
0 Error(s)
Time Elapsed 00:00:02.29
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 26.72 ms for D:\users\kemal\documents\code\shape_dodge\sh
proj.
shape_dodge -> D:\users\kemal\documents\code\shape_dodge\bin\Debug\netcoreapp2
ge.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:01.45
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet build
Microsoft (R) Build Engine version 16.1.76+g14b0a930a7 for .NET Core
Copyright (C) Microsoft Corporation. All rights reserved.
Restore completed in 26.68 ms for D:\users\kemal\documents\code\shape_dodge\sh
proj.
shape_dodge -> D:\users\kemal\documents\code\shape_dodge\bin\Debug\netcoreapp2
ge.dll
Build succeeded.
0 Warning(s)
0 Error(s)
Time Elapsed 00:00:01.40
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet run
Game for Task 2.3

14 Transactions

Add transaction classes to your program

Outcome	Weight
Principles	♦♦♦◊◊

Good task

Outcome	Weight
Build Programs	♦♦♦♦◊

Good task

Outcome	Weight
Design	♦♦♦◊◊

Good task

Date	Author	Comment
2019/08/03 19:17	Achmad Kemal	Ready to Mark
2019/08/03 19:17	Achmad Kemal	Good task
2019/08/03 20:26	Mengmeng Ge	In line 44 of StockPurchaseTransaction, it is to purchase stock, so you should use AddStock method. In line 43 of StockAdjustmentTransaction, it could be either remove or add based on the quantity.
2019/08/03 20:26	Mengmeng Ge	Fix and Resubmit
2019/08/04 01:03	Achmad Kemal	Ready to Mark
2019/08/04 01:14	Achmad Kemal	Ready to Mark
2019/08/04 12:42	Achmad Kemal	Ready to Mark
2019/08/04 13:16	Achmad Kemal	I'm not sure about my StockAdjustment Transaction in the progam.cs . Can you give code about my StockAdjustment code
2019/08/04 14:00	Mengmeng Ge	You could have if-else to check whether _quantity is larger than 0 or not. If larger than 0, add stock; otherwise, remove stock.
2019/08/04 14:00	Mengmeng Ge	Fix and Resubmit
2019/08/04 22:24	Achmad Kemal	I'm just done my code but I don't feel if I'm doing right or not. I already follow your comment that you gave me yesterday.
2019/08/05 08:51	Mengmeng Ge	We will discuss this during today's practical.
2019/08/05 08:51	Mengmeng Ge	Fix and Resubmit
2019/08/05 16:13	Achmad Kemal	Ready to Mark
2019/08/05 16:27	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Transactions

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/05 16:13

Tutor:
Mengmeng Ge

Outcome	**Weight**
Principles	♦♦♦◊◊
Build Programs	♦♦♦♦◊
Design	♦♦♦◊◊

Good task

August 5, 2019

```
using System;
using SplashKitSDK;

public class Program
{

    private static MenuOption ReadUserOption()
    {
        Console.WriteLine("1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit  ");
        int option;
        do
        {
            try
        {
            string inputText;
            Console.Write("Choose your option [1-4]:");
            inputText = Console.ReadLine();
            option = Convert.ToInt32(inputText);
        }
            catch
        {
            //output message
            option = -1;
        }

        }while(option < 1 || option >4);
        return (MenuOption)(option -1);
    }
    public static void Main()
    {
        MenuOption userSelection;
        userSelection = ReadUserOption();
        Console.WriteLine(userSelection);
        Stock test = new Stock("Summary Stock Item", 100);
        do
        {
            userSelection = ReadUserOption();
            switch(userSelection)
            {
                case MenuOption.BuyStock:
                    PerformBuyStock(test);
                    break;
                case MenuOption.SellStock:
                    PerformSellStock(test);
                    break;
                case MenuOption.QueryStock:
                    PerformQueryStock(test);
                    break;
            }
        } while (userSelection != MenuOption.Quit);
    }
        private static void PerformBuyStock(Stock stock)
        {
```

```
54
55              int quantity;
56              decimal price;
57
58
59
60              Console.WriteLine($"Enter your number {stock.Name}");
61              quantity = Convert.ToInt32(Console.ReadLine());
62
63              Console.WriteLine($"Sale price per item bought:");
64              price = Convert.ToDecimal(Console.ReadLine());
65
66              stock.AddStock(quantity);
67
68              //Transaction
69              StockPurchaseTransaction sale;
70              sale = new StockPurchaseTransaction(stock, price, quantity);
71
72              sale.Execute();
73              sale.PrintSummary();
74
75              stock.AddStock( quantity);
76
77          }
78
79          private static void PerformSellStock(Stock stock)
80          {
81              int quantity;
82              decimal price;
83
84              Console.WriteLine($"Enter your number {stock.Name}");
85              quantity = Convert.ToInt32(Console.ReadLine());
86
87              Console.WriteLine($"Sale price per item sold:");
88              price = Convert.ToDecimal(Console.ReadLine());
89
90              //Transaction
91              StockSaleTransaction sale;
92              sale = new StockSaleTransaction(stock, price, quantity);
93
94              sale.Execute();
95              sale.PrintSummary();
96
97              stock.RemoveStock( quantity);
98
99          }
100         private static void PerformQueryStock(Stock stock)
101         {
102
103             stock.PrintSummary();
104         }
105 }
```

```
using System;


public class Stock
{
    private int _quantityOnHand;
    private string _name;
    // Constructor
    public Stock(string name, int initialLevel)
    {
        _name = name;
        _quantityOnHand = initialLevel;
    }

    //Method to add stock
   public bool AddStock(int quantityAdded)
   {
       if (quantityAdded > 0)
       {
           _quantityOnHand = _quantityOnHand +quantityAdded;
           return true;
       }
       else
       {
           return false;
       }
   }
      public bool RemoveStock(int quantityAdded)
   {
       if (quantityAdded > 0 && quantityAdded < QuantityOnHand)
       {
           _quantityOnHand = _quantityOnHand - quantityAdded;
           return true;
       }
       else
       {
           return false;
       }
   }
    //Property
    public string Name
    {
        get {return _name;} // "Get" is for read
        set{_name = value;} // "Set" is for write
    }

    //Property
    public int QuantityOnHand
    {
        get { return _quantityOnHand;}
        private set {_quantityOnHand = value;}
    }
   public void PrintSummary()
```

```
54      {
55          Console.WriteLine($"{Name}: {QuantityOnHand}");
56      }
57
58  }
```

```
using System;

public class StockSaleTransaction
{
    private readonly Stock _stock;
    private readonly decimal _price;
    private int _quantity;

    private bool _hasExecuted = false;
    private bool _success = false;

    public bool HasExecuted
    {
        get
        {
            return _hasExecuted;
        }
    }

    public bool Success
    {
        get
        {
            return _success;
        }
    }

    public StockSaleTransaction(Stock stock, decimal price, int quantity)
    {
        _stock = stock;
        _price = price;
        _quantity = quantity;
    }

    public void Execute()
    {
        if (HasExecuted)
        {
            throw new InvalidOperationException();
        }

        _hasExecuted = true;

        _success = _stock.RemoveStock(_quantity);
    }

    public void PrintSummary()
    {
        Console.Write($"SELL - {_stock.Name} x {_quantity} @ ${_price}");
        if ( ! HasExecuted)
        {
            Console.Write("Proposed");
        }
```

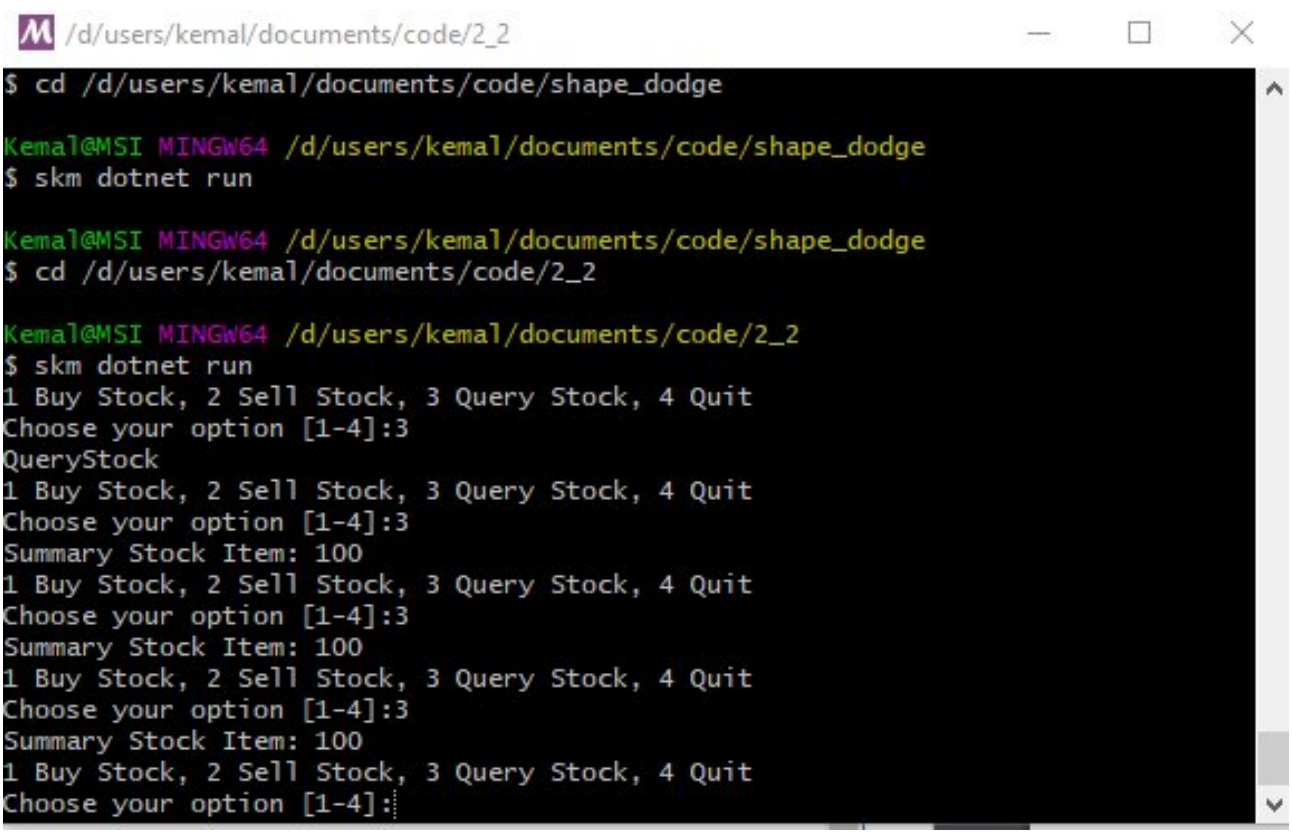
```
54          else if ( ! Success)
55          {
56              Console.Write("Failed");
57          }
58          Console.WriteLine();
59      }
60  }
```

```
using System;

public class StockPurchaseTransaction
{
    private readonly Stock _stock;
    private readonly decimal _price;
    private int _quantity;

    private bool _hasExecuted = false;
    private bool _success = false;

    public bool HasExecuted
    {
        get
        {
            return _hasExecuted;
        }
    }

    public bool Success
    {
        get
        {
            return _success;
        }
    }

    public StockPurchaseTransaction(Stock stock, decimal price, int quantity)
    {
        _stock = stock;
        _price = price;
        _quantity = quantity;
    }

    public void Execute()
    {
        if (HasExecuted)
        {
            throw new InvalidOperationException();
        }

        _hasExecuted = true;

        _success = _stock.AddStock(_quantity);
    }

    public void PrintSummary()
    {
        Console.Write($"SELL - {_stock.Name} x {_quantity} @ ${_price}");
        if ( ! HasExecuted)
        {
            Console.Write("Proposed");
        }
```

```
54          else if ( ! Success)
55          {
56              Console.Write("Failed");
57          }
58          Console.WriteLine();
59      }
60  }
```

```
using System;

public class StockAdjustmentTransaction
{
    private readonly Stock _stock;

    private int _quantity;

    private bool _hasExecuted = false;
    private bool _success = false;

    private readonly decimal _price = 0;


    public bool HasExecuted
    {
        get
        {
            return _hasExecuted;
        }
    }

    public bool Success
    {
        get
        {
            return _success;
        }
    }

    public StockAdjustmentTransaction(Stock stock, int quantity)
    {
        _stock = stock;
        _quantity = quantity;
    }

    public void Execute()
    {
        if (HasExecuted)
        {
            throw new InvalidOperationException();
        }

        _hasExecuted = true;
        if ( _quantity > 0)
        {
            _success = _stock.AddStock(_quantity);

        }
        else
        {
           _success = _stock.RemoveStock(- _quantity);
        }
```

```
54
55      }
56
57      public void PrintSummary()
58      {
59           Console.Write( $"Adjust - {_stock.Name} x {_quantity} @ ${_price}");
60          if ( ! HasExecuted)
61          {
62              Console.Write("Proposed");
63          }
64          else if ( ! Success)
65          {
66              Console.Write("Failed");
67          }
68          Console.WriteLine();
69      }
70  }
```


/d/users/kemal/documents/code/2_2
$ cd /d/users/kemal/documents/code/shape_dodge
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ skm dotnet run
Kemal@MSI MINGW64 /d/users/kemal/documents/code/shape_dodge
$ cd /d/users/kemal/documents/code/2_2
Kemal@MSI MINGW64 /d/users/kemal/documents/code/2_2
$ skm dotnet run
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:3
QueryStock
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:3
Summary Stock Item: 100
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:3
Summary Stock Item: 100
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:3
Summary Stock Item: 100
1 Buy Stock, 2 Sell Stock, 3 Query Stock, 4 Quit
Choose your option [1-4]:

15 Messy Code

What does this mess do? Lets fix it up!

Outcome	Weight
Evaluate Code	♦♦♦♦◊

Code refactoring is all about fixing the code

Outcome	Weight
Principles	♦♦♦♦◊

Code refactoring is all about fixing the code

Outcome	Weight
Justify	♦♦♦♦◊

Code refactoring is all about fixing the code

Date	Author	Comment
2019/08/04 00:37	Achmad Kemal	Ready to Mark
2019/08/04 13:53	Mengmeng Ge	A couple of things:1. Please check the following link for formatting the code:https://code.visualstudio.com/docs/editor/codebasics#_formatting2. Some variables still don't read well (e.g., line 17, private ShipThing a). Please make sure all variables are changed.Please check the formal definition of code refactoring.
2019/08/04 13:53	Mengmeng Ge	Fix and Resubmit
2019/08/04 23:39	Achmad Kemal	Ready to Mark
2019/08/05 08:51	Mengmeng Ge	Discuss
2019/08/05 15:25	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Messy Code

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/04 23:39

Tutor:
Mengmeng Ge

Outcome	**Weight**
Evaluate Code	♦♦♦♦◊
Principles	♦♦♦♦◊
Justify	♦♦♦♦◊

Code refactoring is all about fixing the code

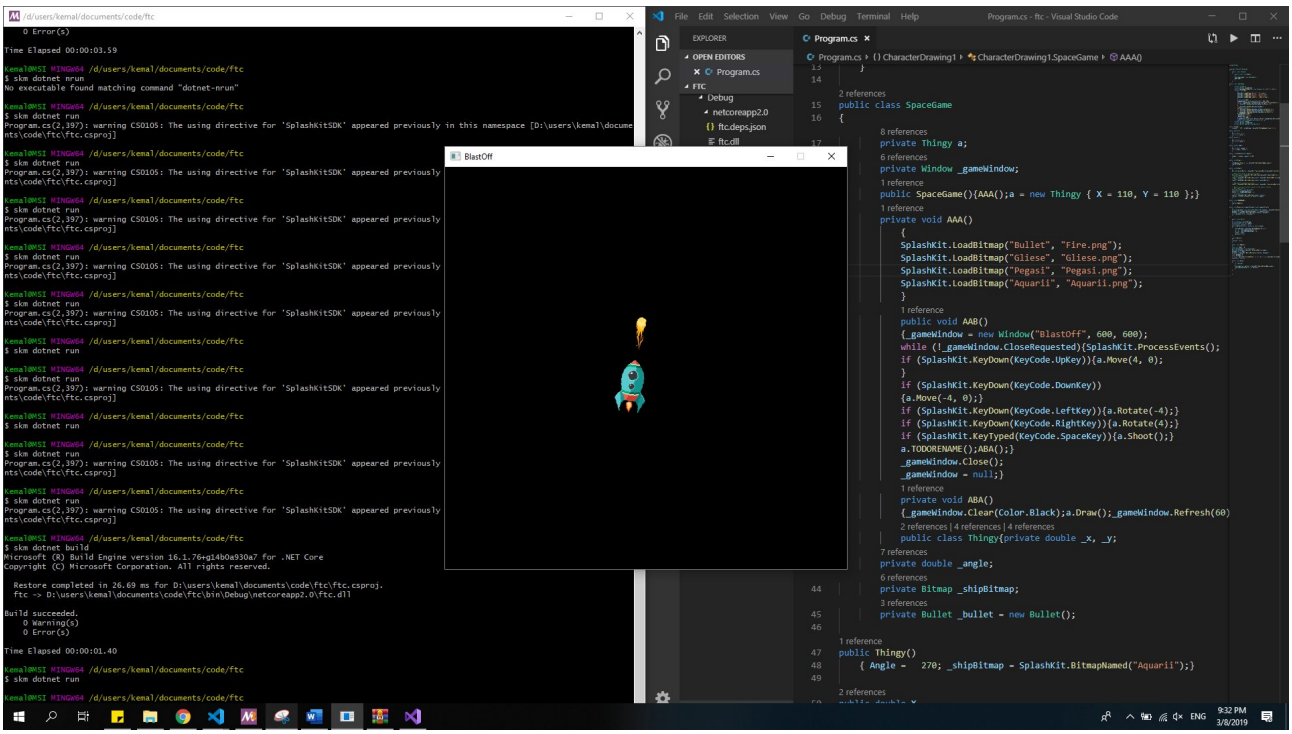
August 4, 2019

```
using System;
using SplashKitSDK;


namespace CharacterDrawing1
{
    public class Program
    {
        public static void Main()
        {
            SpaceGame game = new SpaceGame();
            game.HandleInput();
        }
    }

    public class SpaceGame
    {
        private ShipThing Ship;
        private Window _gameWindow;
        public SpaceGame() { Ships(); Ship = new ShipThing { X = 110, Y = 110 }; }

        private void Ships()
        {
            SplashKit.LoadBitmap("Bullet", "Fire.png");
            SplashKit.LoadBitmap("Gliese", "Gliese.png");
            SplashKit.LoadBitmap("Pegasi", "Pegasi.png");
            SplashKit.LoadBitmap("Aquarii", "Aquarii.png");
        }
        public void HandleInput()
        {
            _gameWindow = new Window("BlastOff", 600, 600);
            while (!_gameWindow.CloseRequested)
            {
                SplashKit.ProcessEvents();
                if (SplashKit.KeyDown(KeyCode.UpKey))
                {
                    Ship.Move(4, 0);
                }
                if (SplashKit.KeyDown(KeyCode.DownKey))
                {
                    Ship.Move(-4, 0);
                }
                if (SplashKit.KeyDown(KeyCode.LeftKey))
                {
                    Ship.Rotate(-4);
                }
                if (SplashKit.KeyDown(KeyCode.RightKey))
                {
                    Ship.Rotate(4);
                }
                if (SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    Ship.Shoot();
```

```
                }
                Ship.Bullets(); Window();
            }
            _gameWindow.Close();
            _gameWindow = null;
        }
        private void Window()
        {
            _gameWindow.Clear(Color.Black);
            Ship.Draw();
            _gameWindow.Refresh(60);
        }
    }

    public class ShipThing
    {
        private double _x, _y;
        private double _angle;
        private Bitmap _shipBitmap;
        private Bullet _bullet = new Bullet();

        public ShipThing()
        { Angle = 270; _shipBitmap = SplashKit.BitmapNamed("Aquarii"); }

        public double X
        {
            get { return _x; }
            set { _x = value; }
        }
        public double Y
        {
            get { return _y; }
            set { _y = value; }
        }

        public double Angle
        {
            get { return _angle; }
            set { _angle = value; }
        }

        public void Rotate(double amount)
        {
            _angle = (_angle + amount) % 360;
        }

        public void Draw()
        {
            _shipBitmap.Draw(_x, _y, SplashKit.OptionRotateBmp(_angle));
            _bullet.Draw();
        }

        public void Shoot()
```

```
107         {
108             Matrix2D anchorMatrix =
                ↪ SplashKit.TranslationMatrix(SplashKit.PointAt(_shipBitmap.Width /
                ↪ 2, _shipBitmap.Height / 2));
109
110             // Move centre point of picture to origin
111             Matrix2D result = SplashKit.MatrixMultiply(SplashKit.IdentityMatrix(),
                ↪ SplashKit.MatrixInverse(anchorMatrix));
112             // Rotate around origin
113             result = SplashKit.MatrixMultiply(result,
                ↪ SplashKit.RotationMatrix(_angle));
114             // Move it back...
115             result = SplashKit.MatrixMultiply(result, anchorMatrix);
116
117             // Now move to location on screen...
118             result = SplashKit.MatrixMultiply(result,
                ↪ SplashKit.TranslationMatrix(X, Y));
119
120             // Result can now transform a point to the ship's location
121             // Get right/centre
122             Vector2D vector = new Vector2D();
123             vector.X = _shipBitmap.Width;
124             vector.Y = _shipBitmap.Height / 2;
125             // Transform it...
126             vector = SplashKit.MatrixMultiply(result, vector);
127             _bullet = new Bullet(vector.X, vector.Y, Angle);
128         }
129
130         public void Bullets()
131         {
132             _bullet.Update();
133         }
134
135         public void Move(double amountForward, double amountStrafe)
136         {
137             Vector2D movement = new Vector2D(); Matrix2D rotation =
                ↪ SplashKit.RotationMatrix(_angle);
138             movement.X += amountForward; movement.Y += amountStrafe;
139             movement = SplashKit.MatrixMultiply(rotation, movement);
140             _x += movement.X; _y += movement.Y;
141         }
142     }
143
144     public class Bullet
145     {
146         private Bitmap _bulletBitmap;
147         private double _x, _y, _angle;
148         private bool _active = false;
149         public Bullet(double x, double y, double angle)
150         {
151             _bulletBitmap = SplashKit.BitmapNamed("Bullet");
152             _x = x - _bulletBitmap.Width / 2;
153             _y = y - _bulletBitmap.Height / 2;
```

```
154             _angle = angle;
155             _active = true;
156         }
157 
158         public Bullet()
159         {
160             _active = false;
161         }
162 
163         public void Update()
164         {
165             const int TOAST = 8;
166             Vector2D movement = new Vector2D();
167             Matrix2D rotation = SplashKit.RotationMatrix(_angle);
168             movement.X += TOAST;
169             movement = SplashKit.MatrixMultiply(rotation, movement);
170             _x += movement.X;
171             _y += movement.Y;
172             if ((_x > SplashKit.ScreenWidth() || _x < 0) || _y >
                ↪  SplashKit.ScreenHeight() || _y < 0)
173             { _active = false; }
174         }
175 
176         public void Draw()
177         {
178             if (_active)
179             {
180                 DrawingOptions options = SplashKit.OptionRotateBmp(_angle);
181                 _bulletBitmap.Draw(_x, _y, options);
182             }
183         }
184 
185 
186     }
187 }
```


BlastOff
Program.cs - ftc - Visual Studio Code
public class SpaceGame
{
private Thingy a;
private Window _gameWindow;
public SpaceGame(){AAA();a = new Thingy { X = 110, Y = 110 };}
private void AAA()
{
SplashKit.LoadBitmap("Bullet", "Fire.png");
SplashKit.LoadBitmap("Gliese", "Gliese.png");
SplashKit.LoadBitmap("Pegasi", "Pegasi.png");
SplashKit.LoadBitmap("Aquarii", "Aquarii.png");
}
public void AAB()
{_gameWindow = new Window("BlastOff", 600, 600);
while (!_gameWindow.CloseRequested){SplashKit.ProcessEvents();
if (SplashKit.KeyDown(KeyCode.UpKey)){a.Move(4, 0);
}
if (SplashKit.KeyDown(KeyCode.DownKey))
{a.Move(-4, 0);}
if (SplashKit.KeyDown(KeyCode.LeftKey)){a.Rotate(-4);}
if (SplashKit.KeyDown(KeyCode.RightKey)){a.Rotate(4);}
if (SplashKit.KeyTyped(KeyCode.SpaceKey)){a.Shoot();}
a.TODORENAME();ABA();}
_gameWindow.Close();
_gameWindow = null;}
private void ABA()
{_gameWindow.Clear(Color.Black);a.Draw();_gameWindow.Refresh(60)
public class Thingy{private double _x, _y;
private double _angle;
private Bitmap _shipBitmap;
private Bullet _bullet = new Bullet();
public Thingy()
{ Angle = 270; _shipBitmap = SplashKit.BitmapNamed("Aquarii");}

SIT771 (Task 4.2)

Achmad Mustafa Kemal

ID: 219374683

Task 4.2

My reflection:

First of all, my reflection is this code is very hard to read and then the programmer uses unintelligible word for variable in the program. Such as, “private thingy a”, “private void AAA” and “game AAB”. Also, the programmer who have this code never use code basic formatting because code formatting is important to implementation a code in the program. They should think about indentation, naming and use of case. The quality of code is making us to know how to fix the problem and understanding about the code logic. For example, if the programmer uses good code formatting in the program, the programmer will know flow of their code in the software development. In the software development section, there is activity called code refactoring. In general, Code refactoring is the process of reorganize existing code program without changing independent behaviour. The purpose of code refactoring is to improve nonfunctionally property inside program. Code refactoring can reduce complexity and improve code readability. So, the code look more extensibility and significant. In this task, we must make the code look tidier than before and change some variable. It took more time to code refactoring in this code because the code formatting of the is very horrible and incompetent to read. So, I must reformat the code. After reformatting, I can see the code easier and more understandable. Code refactoring is very affect to me in this task because fix somebody code is not always easy job to do it. Also, I rename some variable to make more understandable. For naming, I found that the name of variable is make the reader hard time to read. So, I must choose simple word for the variable. In the end, I feel like code refactoring is very useful to me when I help somebody in the software development.

16 Robot Dodge

Add dodging functionality to the robot class

Outcome	Weight
Build Programs	♦♦♦◊◊

The update method is to make the player hit the robot and then the robot will give new position in the window

Outcome	Weight
Design	♦◊◊◊◊

The update method is to make the player hit the robot and then the robot will give new position in the window

Outcome	Weight
Justify	♦◊◊◊◊

The update method is to make the player hit the robot and then the robot will give new position in the window

Date	Author	Comment
2019/08/05 21:44	Achmad Kemal	image comment
2019/08/05 21:44	Achmad Kemal	i don't understand with this part
2019/08/05 21:47	Achmad Kemal	image comment
2019/08/05 21:47	Achmad Kemal	this is my code
2019/08/06 16:35	Mengmeng Ge	Your RandomRobot() method returns a Robot object. You should call RandomRobot() method and then assign it _testRobot
2019/08/06 16:37	Mengmeng Ge	Like the following: _testRobot = RandomRobot();
2019/08/11 18:07	Achmad Kemal	I tried to my player to collided with my robot but it's doesn't work. Can you give a comment to my code?Thanks
2019/08/12 08:54	Mengmeng Ge	Please specify why it doesn't work. You will need to move the player to hit the robot. Did you debug your code?
2019/08/12 16:26	Achmad Kemal	well I try to fix my update but I thought my update method is worked but when I try to run my program, suddenly my robot move randomly and I'm not sure my robot collided with my player function is working or not. Please help me.Thank you
2019/08/12 16:26	Achmad Kemal	I can't give you screenshot that show my robot move randomly fast
2019/08/13 08:17	Mengmeng Ge	In RobotDodge class, line 28, what is RandomRobot. It should be RandomRobot method you create from line 63. In Update method, you should remove line 52 and 54. Instead, in line 58, you need to call RandomRobot method you create. Otherwide, there is no need to create RandomRobot method.
2019/08/13 08:21	Mengmeng Ge	Also at the current stage, the robot can not move yet. Everytime the player hit the robot, a new robot will be created.
2019/08/13 20:58	Achmad Kemal	Ready to Mark
2019/08/13 21:15	Achmad Kemal	https://youtu.be/nc7CahGQ_0I
2019/08/13 21:15	Achmad Kemal	here link my video
2019/08/13 21:16	Achmad Kemal	for this task
2019/08/14 08:26	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Robot Dodge

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/13 20:58

Tutor:
Mengmeng Ge

Outcome	Weight
Build Programs	♦♦♦◊◊
Design	♦◊◊◊◊
Justify	♦◊◊◊◊

The update method is to make the player hit the robot and then the robot will give new position in the window

August 13, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        Window gameWindow;

        gameWindow = new Window("Game for Task 2.3", 800, 600);
        //Player player = new Player (gameWindow);
        Robotdodge robotdodge = new Robotdodge(gameWindow);
        do
        {
            SplashKit.ProcessEvents();
            //player.StayOnWindow(gameWindow);
            robotdodge.HandleInput();
            robotdodge.Draw();
            //player.Draw();
            robotdodge.Update(gameWindow);
            gameWindow.Refresh(60);
            gameWindow.Clear(Color.White);

        }while(!robotdodge.Quit);
    }
}
```

```
using System;
using SplashKitSDK;


    //Constructor
    public class Player
    {
        //Property
        private Bitmap _playerBitmap;
        private double _x;
        private bool _quit = false;

        public double X
        {
            get {return _x;}
            set { _x = value;}
        }
        private double _y;
        public double Y
        {
            get {return _y;}
            set { _y = value;}
        }

        public bool quit
        {
            get
            {return _quit;}
            private set
            {_quit = value;}
        }

        public void Move (double dx, double dy)
        {

          _x = _x + dx;
          _y = _y + dy;
        }
        public int Width
        {
            get
            {
                return _playerBitmap.Width;
            }
        }

            public int Height
        {
           get
            {
                return _playerBitmap.Height;
            }
        }
```

```
54          public Player( Window gameWindow)
55          {
56              _playerBitmap  = new Bitmap("player", "curry.png");
57              X = (gameWindow.Width - Width) / 2;
58              Y = (gameWindow.Height - Height) / 2;
59
60          }
61          //Public Method
62          public void Draw()
63          {
64              _playerBitmap.Draw(X,Y);
65
66          }
67
68          public void HandleInput()
69          {
70
71              SplashKit.ProcessEvents();
72              int speed = 5;
73          // Arrow Keys for move function
74          // X is from public player
75          // Y is from public player
76              if (SplashKit.KeyDown(KeyCode.RightKey))
77              {
78              X += speed;
79              }
80
81              if (SplashKit.KeyDown(KeyCode.LeftKey))
82              {
83              X -= speed;
84              }
85              if (SplashKit.KeyDown(KeyCode.UpKey))
86              {
87              Y -= speed;
88              }
89              if (SplashKit.KeyDown(KeyCode.DownKey))
90              {
91              Y += speed;
92              }
93          // Key for quit button
94              if (SplashKit.KeyDown(KeyCode.EscapeKey))
95              {
96              quit = true;
97              }
98          }
99          // This is for player no to go out from windows
100         public void StayOnWindow(Window gameWindow)
101         {
102
103             const int GAP = 10;
104         // X is from public player
105         // Y is from public player
106             if (X < GAP)
```

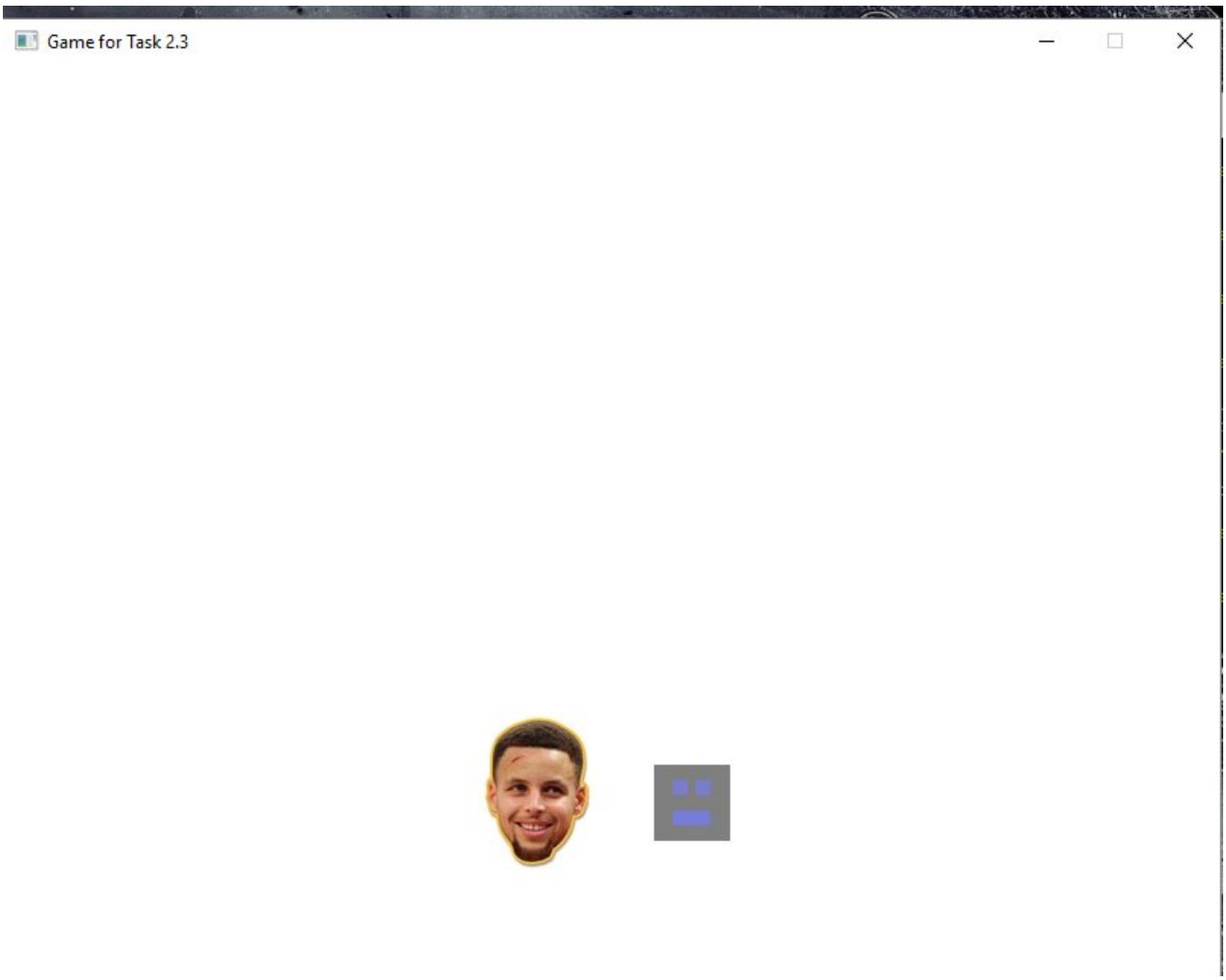
```
107             {
108                 X = GAP;
109             }
110             if (X > gameWindow.Width - Width - GAP)
111             {
112                 X = gameWindow.Width - Width - GAP;
113             }
114             if (Y < GAP)
115             {
116                 Y = GAP;
117             }
118             if (Y > gameWindow.Height - Height - GAP)
119             {
120                 Y = gameWindow.Height - Height - GAP;
121             }
122 
123 
124         }
125 
126 
127         //This method for Collision Function
128         public bool CollidedWith(Robot other)
129         {
130             //return true;
131             return _playerBitmap.CircleCollision(X,Y, other.CollisionCircle);
132         }
133 
134 }
```

```
using System;
using SplashKitSDK;

    public class Robot
    {
        //Properties
        private double X
        {get;set;}
        private double Y
        {get;set;}
        public Color MainColor;

        public int Width
        {get
        { return 50;}
        }
        public int Height
        {get
        { return 50;}
        }

        //Robot Collision
        public Circle CollisionCircle
        {
            get
            {
                return SplashKit.CircleAt(X, Y, 20);
            }
        }

        //Constructor to accepts random robot
        public Robot (Window gameWindow)
        {
            //Assign robot color
            MainColor = Color.RandomRGB(200);

            //Get Random location for robot
            X = SplashKit.Rnd(gameWindow.Width - Width);
            Y = SplashKit.Rnd(gameWindow.Height - Height);
        }

        //Create a robot
        public void Draw()
        {
            //Local Variables
            double leftX;
            double rightX;
            double eyeY;
            double mouthY;
            //Bitmap Nba = new Bitmap("Nba", "nba.jpg");

            //Assign LV
            leftX = X + 12;
```

```
54              rightX = X + 27;
55              eyeY = Y + 10;
56              mouthY = Y + 30;
57
58              //Draw a robot
59              SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
60              SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
61              SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
62              SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
63              SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
64              //SplashKit.DrawBitmap(Nba, X, Y);
65          }
66
67
68
69
70
71
72      }
```

```
using SplashKitSDK;
using System;

public class Robotdodge
{
    //Private Field
    private Player _Player;
    private Window _GameWindow;
    private Robot _TestRobot;


    public bool Quit
    {
        get
        {
            return _Player.quit;
        }
    }


    //Constructor
    public Robotdodge(Window gameWindow)
    {
        Player player = new Player(gameWindow);
        _GameWindow = gameWindow;
        _Player = player;
        RandomRobot(gameWindow);
    }

    // Manage the player
    public void Draw()
    {
        //_GameWindow.Refresh(60);
        _Player.Draw();
        _TestRobot.Draw();
        //_GameWindow.Clear(Color.White);
    }
    public void HandleInput()

    {
        _Player.HandleInput();
        _Player.StayOnWindow(_GameWindow);
    }


    //For Update Robot position
    public void Update(Window gameWindow)
    {

        if (_Player.CollidedWith(_TestRobot))
        {

        Robot RandomRobot = new Robot(gameWindow);
```

```
54          _TestRobot = RandomRobot;
55          }
56      }
57
58          public void RandomRobot(Window gameWindow)
59      {
60
61          Robot RandomRobot = new Robot(gameWindow);
62        _TestRobot = RandomRobot;
63
64      }
65
66
67
68
69  }
```


Game for Task 2.3

17 Concept Visualisation 1

Visualise the programming concepts we've covered so far

Outcome	Weight
Principles	♦♦♦◊◊

Concept Visualisation is make us to prevent project failure

Outcome	Weight
Design	♦♦♦◊◊

Concept Visualisation is make us to prevent project failure

Outcome	Weight
Justify	♦♦♦◊◊

Concept Visualisation is make us to prevent project failure

Date	Author	Comment
2019/08/12 01:11	Achmad Kemal	Ready to Mark
2019/08/12 08:56	Mengmeng Ge	Please add the explanation of method in your example.
2019/08/12 08:56	Mengmeng Ge	Fix and Resubmit
2019/08/12 15:50	Achmad Kemal	Ready to Mark
2019/08/13 08:09	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Concept Visualisation 1

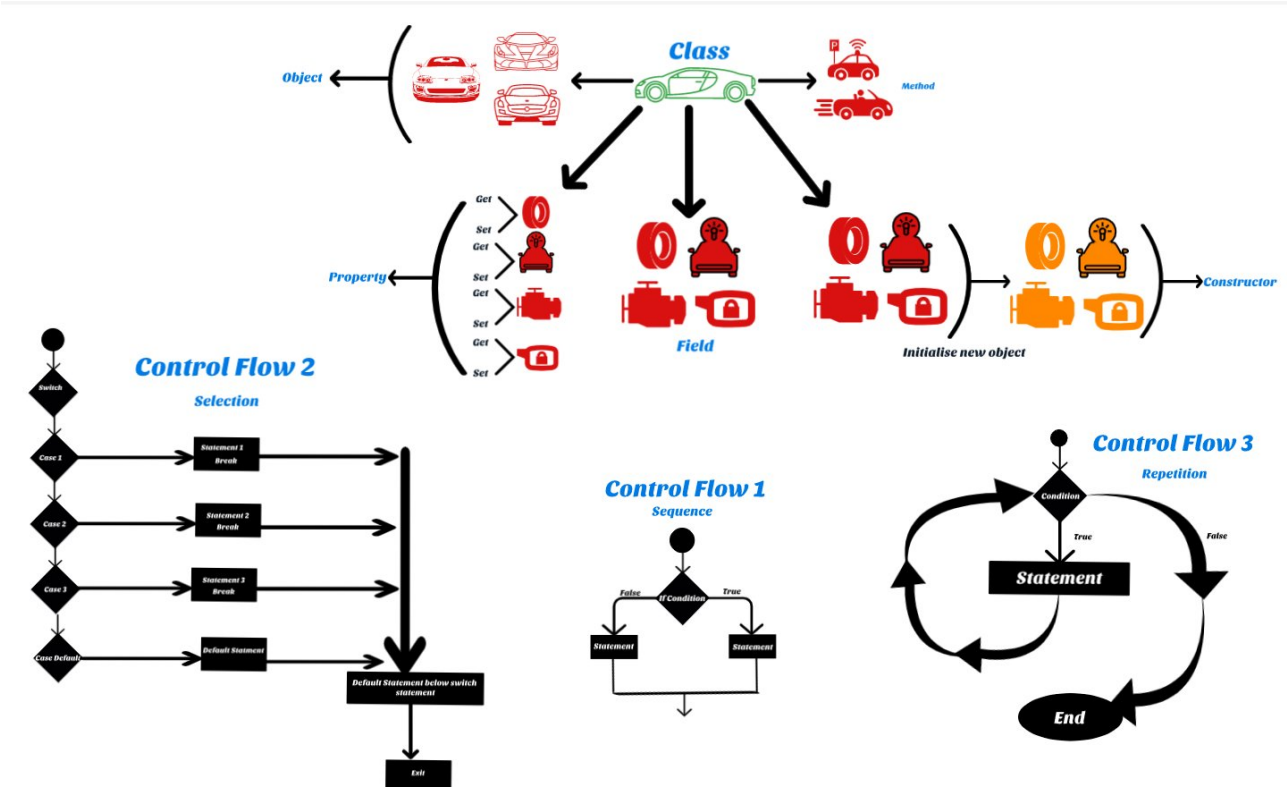
Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/12 15:50

Tutor:
Mengmeng Ge

Outcome	**Weight**
Principles	♦♦♦◊◊
Design	♦♦♦◊◊
Justify	♦♦♦◊◊

Concept Visualisation is make us to prevent project failure

August 12, 2019


Class
Object
Method
Property
Get
Set
Field
Initialise new object
Constructor

Control Flow 2
Selection
Switch
Case 1
Case 2
Case 3
Case Default
Statement 1
Break
Statement 2
Break
Statement 3
Break
Default Statment
Default Statement below switch statement
Exit

Control Flow 1
Sequence
False
If Condition
True
Statement
Statement

Control Flow 3
Repetition
Condition
True
False
Statement
End

SIT771

ACHMAD MUSTAFA KEMAL

ID: 219374683

In this task, I'm going explain about this concept visualisation. In this concept, you can see that Class is a data type that combine data and function to access data. For example, we can assume that class is a car. Car contain many parts and function in one car. Second, I will explain about method. Method is a contain statement that can implement when the statement is going to use. In the picture, Method can assume like car functionality. Method is must be declared within a class. For example, Car method is can do park the car in the parking spot and car can go speedy on the road. Object is a sample of a class that is builded firmly. We can use like type of car in the c#, Such as, Supra, Ferrari and Mercedes Benz. Property is a member that give a flexible mechanism to read, write and calculate of private field. For example, car can set the tyre. So, the car can move easily. Also, Car can get move when can have four tyres. Field is a member that represent memory location for saving a value. Field have many access modifier, such as, public, private, protected and internal. We can assume part of car is our field. For example, tyre, car light, engine and mirror. Constructor is a special method that can use for initialize the data member of new object. For example, tyre can have new tyre.

In the C#, there are control flow to handle a program. Such as, sequence, selection and repetition. Sequence known as basic control flow. For example, if statement is known as logical statement. If a car faster than bicycle, than result will be true or can be return as false. Selection statement is a statement that can have many case in the program. For example, Switch case is usually use for menu option function in the C# programming language. Repetition statement is like while statement. While statement will be used to executed or defend of statement while Boolean expression value is true.

18 Arrays

Create and manipulate arrays

Outcome	Weight
Build Programs	♦♦♦◊◊

With using Array, we can store multiple value

Outcome	Weight
Justify	♦♦◊◊◊

With using Array, we can store multiple value

Date	Author	Comment
2019/08/13 23:09	Achmad Kemal	image comment
2019/08/13 23:10	Achmad Kemal	Well, i almost done with array task but i go error in double variable
2019/08/13 23:10	Achmad Kemal	i follow what in the instruction said in the task
2019/08/14 08:24	Mengmeng Ge	If you check the error, it is probably unassigned local variable, which means you will need to ask the user to type a number and assign it to numberofValues before you pass it to create array.
2019/08/14 21:01	Achmad Kemal	Ready to Mark
2019/08/15 16:09	Mengmeng Ge	in line 25, you will need to ask the user to type a number for the length of the array.
2019/08/15 16:09	Mengmeng Ge	Fix and Resubmit
2019/08/15 22:41	Achmad Kemal	Ready to Mark
2019/08/16 12:54	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Arrays

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/15 22:41

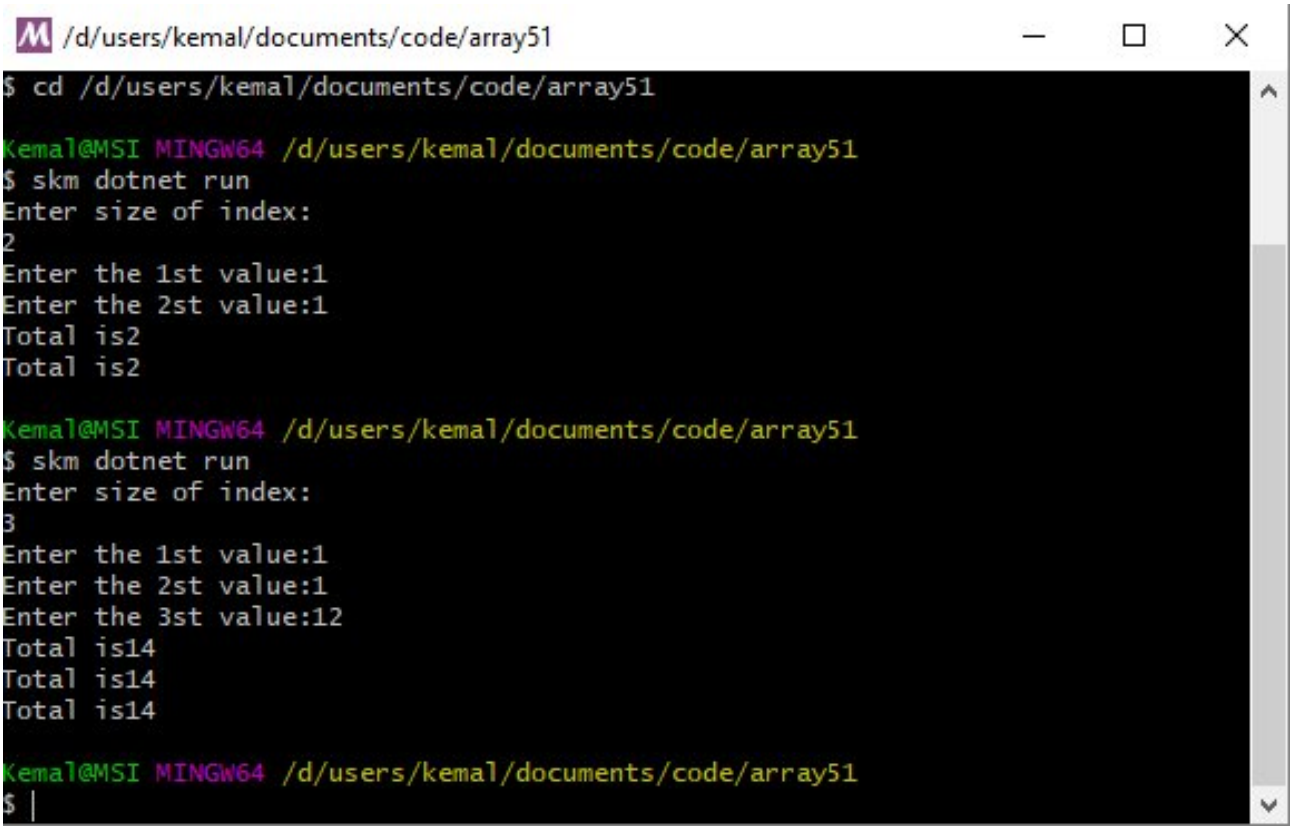
Tutor:
Mengmeng Ge

Outcome	**Weight**
Build Programs	♦♦♦◊◊
Justify	♦♦◊◊◊

With using Array, we can store multiple value

August 15, 2019

```
using System;
using SplashKitSDK;
using System.Linq;

public class Program
{
    public static int ReadDouble(string prompt)
    {
        Console.Write(prompt);

            while (true)
            {
                try
                {
                    return Int32.Parse(Console.ReadLine());
                }
                catch
                {
                    Console.WriteLine("Please enter a valid integer");
                }
            }
    }
    public static void Main()
    {
     int numberOfValues;
     Console.WriteLine("Enter size of index:");
     numberOfValues = int.Parse(Console.ReadLine());

     double [] values = new double[numberOfValues];

     for (int i = 0; i < numberOfValues; i++)
     {
         values[i] = ReadDouble($"Enter the {i + 1}st value:");

     }

     for (int i = 0; i < numberOfValues; i++)
      {
            double sum =  values.Sum();
            Console.WriteLine($"Total is" + sum);
      }

    }
}
```


/d/users/kemal/documents/code/array51
$ cd /d/users/kemal/documents/code/array51
Kemal@MSI MINGW64 /d/users/kemal/documents/code/array51
$ skm dotnet run
Enter size of index:
2
Enter the 1st value:1
Enter the 2st value:1
Total is2
Total is2
Kemal@MSI MINGW64 /d/users/kemal/documents/code/array51
$ skm dotnet run
Enter size of index:
3
Enter the 1st value:1
Enter the 2st value:1
Enter the 3st value:12
Total is14
Total is14
Total is14
Kemal@MSI MINGW64 /d/users/kemal/documents/code/array51
$

19 Lists

Create and manipulate lists

Outcome	Weight
Build Programs	♦♦♦♦◊

List Function is given us easy access to multiple stores value in the program

Outcome	Weight
Justify	♦♦◊◊◊

List Function is given us easy access to multiple stores value in the program

Date	Author	Comment
2019/08/15 00:26	Achmad Kemal	When I want to add value, the program wants me to type over and over. There is no end like an infinite loop. Also, it's happening with my print method. My print method also got an infinite loop. The problem with my sum method is there is no result from the sum method. Can you give a comment with my code?Thanks
2019/08/15 00:26	Achmad Kemal	image comment
2019/08/15 16:13	Mengmeng Ge	Please debug the code line by line. In main method, line 100 should be moved to be inside do.
2019/08/15 16:13	Mengmeng Ge	Fix and Resubmit
2019/08/16 12:04	Achmad Kemal	image comment
2019/08/16 12:04	Achmad Kemal	when i type 0, i can't add value to list
2019/08/16 12:04	Achmad Kemal	image comment
2019/08/16 12:04	Achmad Kemal	here my code
2019/08/16 12:05	Achmad Kemal	do you have any comment with my code?
2019/08/16 12:53	Mengmeng Ge	In line 60, you used _values.Count as the condition. But before you add any value, _values has no values thus the length is 0. You could either ask the user to type a number and use that as the length in the loop; or you only add one value in AddValueToList() method.
2019/08/16 18:21	Achmad Kemal	Ready to Mark
2019/08/16 18:33	Achmad Kemal	Ready to Mark
2019/08/17 19:43	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Lists

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/16 18:33

Tutor:
Mengmeng Ge

Outcome	**Weight**
Build Programs	♦♦♦♦◊
Justify	♦♦◊◊◊

List Function is given us easy access to multiple stores value in the program

August 16, 2019

```
using System;
using SplashKitSDK;
// using System.Linq;
using System.Collections.Generic;


public class Program
{
    private static List<double> _values = new List<double>();
    public static int ReadDouble(string prompt)
    {
        Console.Write(prompt);

            while (true)
            {
                try
                {

                    return Int32.Parse(Console.ReadLine());
                }
                catch
                {
                    Console.WriteLine("Please enter a valid integer");
                }
            }
    }
     public static int ReadInteger(string prompt)
     {
         Console.Write(prompt);

             while (true)
             {
                 try
                 {
                     return Int32.Parse(Console.ReadLine());
                 }
                 catch
                 {
                     Console.WriteLine("Please enter a valid integer");
                 }
             }
     }
    public enum UserOption
    {
        //Values
        AddValueToList,
        Sum,
        Print,
        Quit,

    }

    //This method use the length of list
```

```
54      //Declare the list and get the number of values and iterate to add the values
        ↪  to the list
55      public static void AddValueToList()
56      {
57          _values = new List<double>();
58          int numberOfValues;
59          Console.WriteLine("Enter size of index:");
60          numberOfValues = int.Parse(Console.ReadLine());
61          for ( int i = 0; i < numberOfValues; i++)
62          {
63          _values.Add(ReadDouble( $ "Enter the {i + 1}st value:"));
64          }
65      }
66
67      //both of them are produce same output and same result
68
69      //This method just user only add value one by one
70      // public static void AddValueToList()
71      // {
72
73      //    _values.Add(new double());
74
75
76      //     for ( int i = 0; i < _values.Count; i++)
77      //     {
78
79      //     _values[i] = ReadDouble($"Enter the {i + 1}st value:");
80      //     double numberofValues = _values.Count;
81      //     }
82
83      // }
84      public static void Sum()
85      {
86          // code goes here
87          double sum = 0;
88
89          foreach(double  numberofValues in _values)
90          {
91              sum += numberofValues;
92          }
93          Console.WriteLine( $ "The total is" + sum);
94      }
95      public static void Print()
96      {
97          // code goes here
98          foreach (double numberofValues in _values )
99          {
100             Console.WriteLine(numberofValues);
101         }
102     }
103
104     public static UserOption ReadUserOption()
105     {
```

```
106         Console.WriteLine("Enter 0 to add a value ");
107         Console.WriteLine("Enter 1 to add a sum all values");
108         Console.WriteLine("Enter 2 to print all values");
109         Console.WriteLine("Enter 3 to quit");
110 
111         int option = 3;
112         Int32.TryParse(Console.ReadLine(), out option);
113 
114         return (UserOption)option;
115     }
116     public static void Main()
117     {
118         UserOption userSelection;
119         do
120             {
121             userSelection = ReadUserOption();
122             Console.WriteLine(userSelection);
123              switch(userSelection)
124              {
125                  case UserOption.AddValueToList:
126                         AddValueToList();
127                         break;
128                  case UserOption.Sum:
129                         Sum();
130                         break;
131                  case UserOption.Print:
132                         Print();
133                         break;
134 
135              }
136             }while (userSelection != UserOption.Quit);
137     }
138 }
```

```
Kemal@MSI MINGW64 /d/users/kemal/documents/code/list52
$ skm dotnet run
Enter 0 to add a value
Enter 1 to add a sum all values
Enter 2 to print all values
Enter 3 to quit
0
AddValueToList
Enter size of index:
5
Enter the 1st value:10
Enter the 2st value:55
Enter the 3st value:23
Enter the 4st value:24
Enter the 5st value:46
Enter 0 to add a value
Enter 1 to add a sum all values
Enter 2 to print all values
Enter 3 to quit
1
Sum
The total is158
Enter 0 to add a value
Enter 1 to add a sum all values
Enter 2 to print all values
Enter 3 to quit
2
Print
10
55
23
24
46
Enter 0 to add a value
Enter 1 to add a sum all values
Enter 2 to print all values
Enter 3 to quit
3
Quit

Kemal@MSI MINGW64 /d/users/kemal/documents/code/list52
$
```

20 Warehouse

A warehouse has many products.

Outcome	Weight
Design	♦♦♦♦◊

This task is all about computerized management stock

Outcome	Weight
Justify	♦◊◊◊◊

This task is all about computerized management stock

Date	Author	Comment
2019/08/19 17:19	Achmad Kemal	Ready to Mark
2019/08/19 21:08	Mengmeng Ge	The result is not correct. When you type 3 to buy stock 1234, you should have 13 (12 in option 1 and 1 in option 3).
2019/08/19 21:15	Mengmeng Ge	It might be the problem of printsummary in your transaction classes. Please double check that.
2019/08/19 21:16	Mengmeng Ge	Fix and Resubmit
2019/08/24 12:48	Achmad Kemal	Ready to Mark
2019/08/24 12:50	Achmad Kemal	please give some feedback
2019/08/24 18:10	Mengmeng Ge	Complete
2019/08/24 18:23	Mengmeng Ge	stock list in warehouse class should not be static. It is associated with a warehouse object as every time you create a warehouse object, there will be a stock list created. Please fix that in next task submission.

Deakin University

Object Oriented Development

OnTrack Submission

Warehouse

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/24 12:48

Tutor:
Mengmeng Ge

Outcome	**Weight**
Design	♦♦♦♦◊
Justify	♦◊◊◊◊

This task is all about computerized management stock

August 24, 2019

```
using System;
using SplashKitSDK;

public enum MenuOption
{
    NewStock,
    FindStock,
    BuyStock,
    SellStock,
    AdjustmentStock,
    Quit,
}

public class Program
{

private static MenuOption ReadUserOption()
{
    int option;
    Console.WriteLine("1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit");

        do
        {

            try
            {
                string inputText;
                Console.Write("Choose an Option 1-6");
                inputText = Console.ReadLine();
                option = Convert.ToInt32(inputText);
            }
            catch
            {
                option = -1;
            }

        }while(option < 1 || option > 6);
        return (MenuOption)(option -1);
}

    public static void Main()
    {
        MenuOption userSelection;

        Warehouse test = new Warehouse();

        do
        {
            userSelection = ReadUserOption();
            Console.WriteLine(userSelection);
            switch(userSelection)
            {
```

```
53                  case MenuOption.NewStock:
54                  AddNewStock(test);
55                  break;
56
57                  case MenuOption.FindStock:
58                  FindStockItem(test);
59                  break;
60
61                  case MenuOption.BuyStock:
62                  PeformBuyStock(test);
63                  break;
64
65                  case MenuOption.SellStock:
66                  PeformSellStock(test);
67                  break;
68
69                  case MenuOption.AdjustmentStock:
70                  PerformAdjustStock(test);
71                  break;
72              }
73          }while (userSelection != MenuOption.Quit);
74      }
75      private static void AddNewStock(Warehouse fromWarehouse)
76      {
77          Console.WriteLine("Enter The Stock Name:");
78          string name = Console.ReadLine();
79          Console.WriteLine("Enter The Stock Code:");
80          string code = Console.ReadLine();
81          Console.WriteLine("Enter The Stock Quantity:");
82          int quantity = Convert.ToInt32(Console.ReadLine());
83          Stock stock = new Stock(name, quantity, code);
84          // stock.PrintSummary();
85          fromWarehouse.AddStockItem(stock);
86
87      }
88
89      private static Stock FindStockItem(Warehouse fromWarehouse)
90      {
91          Console.Write("Enter stock code:");
92          string code =  Console.ReadLine();
93          Stock result = fromWarehouse.GetStockItem(code);
94
95          if (result == null)
96          {
97              Console.WriteLine($"No stock found with {code}");
98          }
99          return result;
100     }
101
102     private static void PeformSellStock(Warehouse toWarehouse)
103     {
104         Stock stock = FindStockItem(toWarehouse);
105         if (stock == null) return;
```

```
106         decimal price = 0 ;
107         int quantity = 0;
108 
109          try
110          {
111             Console.WriteLine("Please enter price");
112             price = Convert.ToDecimal(Console.ReadLine());
113             Console.WriteLine("Please enter quantity");
114             quantity = Convert.ToInt32(Console.ReadLine());
115             StockSaleTransaction stocksale = new
                ↪  StockSaleTransaction(stock,price,quantity);
116             stocksale.Execute();
117             stocksale.PrintSummary();
118             stock.RemoveStock(quantity);
119             stock.PrintSummary();
120          }
121          catch
122          {
123              Console.WriteLine("You need to be carefull");
124          }
125 
126     }
127 
128         private static void PeformBuyStock(Warehouse toWarehouse)
129     {
130         Stock stock = FindStockItem(toWarehouse);
131         if (stock == null) return;
132         decimal price = 0;
133         int quantity = 0;
134 
135          try
136          {
137             Console.WriteLine("Please enter price");
138             price = Convert.ToDecimal(Console.ReadLine());
139             Console.WriteLine("Please enter quantity");
140             quantity = Convert.ToInt32(Console.ReadLine());
141             StockPurchaseTransaction stocksale = new
                ↪  StockPurchaseTransaction(stock, price,quantity);
142             stocksale.Execute();
143             stocksale.PrintSummary();
144             stock.AddStock(quantity);
145             stock.PrintSummary();
146          }
147          catch
148          {
149              Console.WriteLine("You need to be carefull");
150          }
151 
152     }
153 
154     private static void PerformAdjustStock(Warehouse toWarehouse)
155     {
156          Stock stock = FindStockItem(toWarehouse);
```

```
157         if (stock == null) return;
158         decimal price;
159         int quantity;
160 
161          try
162          {
163             Console.WriteLine("Please enter price");
164             price = Convert.ToDecimal(Console.ReadLine());
165             Console.WriteLine("Please enter quantity");
166             quantity = Convert.ToInt32(Console.ReadLine());
167             StockAdjustmentTransaction stocksale = new
                ↪  StockAdjustmentTransaction(stock, quantity);
168             stocksale.Execute();
169             stocksale.PrintSummary();
170          }
171          catch
172          {
173              Console.WriteLine("You need to be carefull");
174          }
175 
176 
177     }
178 
179 
180 }
```

```
using System;
using SplashKitSDK;
using System.Linq;
using System.Collections;
using System.Collections.Generic;

public class Warehouse
{

    private static List<Stock> _stockitems = new List<Stock>();

    //Method to add and get stock
    public void AddStockItem(Stock stock)
    {
        // _stockitems =  new List<Stock>();
        // int numberofstock;
        // Console.WriteLine("Enter size of index:");
        // numberofstock = int.Parse(Console.ReadLine());
       _stockitems.Add(stock);
    //    foreach(Stock stocks in _stockitems)
    //    {
    //        stocks.PrintSummary();
    //    }
    }
    public Stock GetStockItem(string code)
    {
       foreach(Stock stocks in _stockitems)
       {
        //    stocks.PrintSummary();
           if (stocks.Code == code)
           {
            Console.WriteLine(code);
            return stocks;
           }
       }
      return null;
    }

    //Method Overloading
    public void ExecuteTransaction(StockSaleTransaction StockSaleTransaction)
    {
        StockSaleTransaction.Execute();
    }
    public void ExecuteTransaction(StockPurchaseTransaction
     ↪  StockPurchaseTransaction)
    {
        StockPurchaseTransaction.Execute();
    }
     public void ExecuteTransaction(StockAdjustmentTransaction
      ↪  StockAdjustmentTransaction)
    {
        StockAdjustmentTransaction.Execute();
    }
```

```
52 }
```

```
using System;
using SplashKitSDK;

public class Stock
{
    //field
    private string _name;
    private int _quantityOnHand;
    private string _code;

    //Constructor
    public Stock(string name, int initialLevel, string code)
    {
        _name = name;
        _quantityOnHand = initialLevel;
        _code = code;
    }



    //Property
    public string Name
    {
        get{return _name;}// Get is for read
        set{_name = value;}// Set is for write
    }
    public string Code
    {
        get{return _code;}// Get is for read
        private set{_code = value;}// Set is for write

    }
    public int QuantityOnHand
    {
        get{return _quantityOnHand;}
        private set{ _quantityOnHand = value;}
    }

    //Method to add stock
   public bool AddStock(int quantityAdded)
   {

       if (quantityAdded > 0)
       {
           Console.WriteLine("check quantity");
           _quantityOnHand = _quantityOnHand + quantityAdded;
           return true;
       }
       else
       {
           return false;
       }
   }
```

```
54          public bool RemoveStock(int quantityAdded)
55      {
56          if (quantityAdded > 0 && quantityAdded < QuantityOnHand)
57          {
58              _quantityOnHand = _quantityOnHand - quantityAdded;
59              return true;
60          }
61          else
62          {
63              return false;
64          }
65      }
66
67       //Method to Print
68       public void PrintSummary()
69       {
70           Console.WriteLine($"{Name}: {QuantityOnHand} With Stock :{Code}");
71       }
72  }
```

```
Kemal@MSI MINGW64 /d/users/kemal/documents/code/warehouse53
$ skm dotnet run
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-6 1
NewStock
Enter The Stock Name:
cccc
Enter The Stock Code:
1234
Enter The Stock Quantity:
1
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-63
BuyStock
Enter stock code:1234
1234
Please enter price
12
Please enter quantity
1
check quantity
Purchase - cccc x 1 @ $12
check quantity
cccc: 3 With Stock :1234
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-6 4
SellStock
Enter stock code:1234
1234
Please enter price
12
Please enter quantity
2
SELL - cccc x 2 @ $12
cccc: 1 With Stock :1234
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-6 6
Quit
```

21 Many Robots

Add many robots to your program

Outcome	Weight
Build Programs	♦♦♦◊◊

Foreach loop is very useful to remove robot and check collision function

Outcome	Weight
Design	♦♦♦◊◊

Foreach loop is very useful to remove robot and check collision function

Outcome	Weight
Justify	♦♦♦◊◊

Foreach loop is very useful to remove robot and check collision function

Date	Author	Comment
2019/08/17 20:49	Achmad Kemal	Ready to Mark
2019/08/17 20:49	Achmad Kemal	https://youtu.be/4J9jsyZ06tE
2019/08/17 20:50	Achmad Kemal	This video for 5.4 task
2019/08/19 08:05	Mengmeng Ge	In line 78, you will also need to check whether the robot is off the screen or not.
2019/08/19 08:05	Mengmeng Ge	Discuss
2019/08/19 15:12	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Many Robots

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/19 13:26

Tutor:
Mengmeng Ge

Outcome	Weight
Build Programs	♦♦♦◊◊
Design	♦♦♦◊◊
Justify	♦♦♦◊◊

Foreach loop is very useful to remove robot and check collision function

August 19, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static void Main()
    {
        Window gameWindow;

        gameWindow = new Window("Wardell Curry Dodge", 800, 600);
        Player player = new Player (gameWindow);
        Robotdodge robotdodge = new Robotdodge(gameWindow, player);
        do
        {
            SplashKit.ProcessEvents();
            //player.StayOnWindow(gameWindow);
            robotdodge.HandleInput();
            robotdodge.Draw();
            //player.Draw();
            robotdodge.Update(gameWindow);
            //robotdodge.RandomRobot();
            gameWindow.Refresh(60);
            gameWindow.Clear(Color.White);

        }while(!robotdodge.Quit);
    }
}
```

```
using System;
using SplashKitSDK;


    //Constructor
    public class Player
    {
        //Property
        private Bitmap _playerBitmap;
        private double _x;
        private bool _quit = false;

        public double X
        {
            get {return _x;}
            set { _x = value;}
        }
        private double _y;
        public double Y
        {
            get {return _y;}
            set { _y = value;}
        }

        public bool quit
        {
            get
            {return _quit;}
            private set
            {_quit = value;}
        }

        public void Move (double dx, double dy)
        {

          _x = _x + dx;
          _y = _y + dy;
        }
        public int Width
        {
            get
            {
                return _playerBitmap.Width;
            }
        }

            public int Height
        {
           get
            {
                return _playerBitmap.Height;
            }
        }
```

```
54          public Player( Window gameWindow)
55          {
56              _playerBitmap  = new Bitmap("player", "curry1.png");
57              X = (gameWindow.Width - Width) / 2;
58              Y = (gameWindow.Height - Height) / 2;
59
60          }
61          //Public Method
62          public void Draw()
63          {
64              _playerBitmap.Draw(X,Y);
65          }
66
67          public void HandleInput()
68          {
69
70              SplashKit.ProcessEvents();
71              int speed = 5;
72          // Arrow Keys for move function
73          // X is from public player
74          // Y is from public player
75              if (SplashKit.KeyDown(KeyCode.RightKey))
76              {
77              X += speed;
78              }
79
80              if (SplashKit.KeyDown(KeyCode.LeftKey))
81              {
82              X -= speed;
83              }
84              if (SplashKit.KeyDown(KeyCode.UpKey))
85              {
86              Y -= speed;
87              }
88              if (SplashKit.KeyDown(KeyCode.DownKey))
89              {
90              Y += speed;
91              }
92          // Key for quit button
93              if (SplashKit.KeyDown(KeyCode.EscapeKey))
94              {
95              quit = true;
96              }
97          }
98          // This is for player no to go out from windows
99          public void StayOnWindow(Window gameWindow)
100         {
101             const int GAP = 10;
102         // X is from public player
103         // Y is from public player
104             if (X < GAP)
105             {
106                 X = GAP;
```

```
107             }
108             if (X > gameWindow.Width - Width - GAP)
109             {
110                 X = gameWindow.Width - Width - GAP;
111             }
112             if (Y < GAP)
113             {
114                 Y = GAP;
115             }
116             if (Y > gameWindow.Height - Height - GAP)
117             {
118                 Y = gameWindow.Height - Height - GAP;
119             }
120         }
121         //This method for Collision Function
122         public bool CollidedWith(Robot other)
123         {
124             //return true;
125             return _playerBitmap.CircleCollision(X,Y, other.CollisionCircle);
126         }
127 
128 }
```

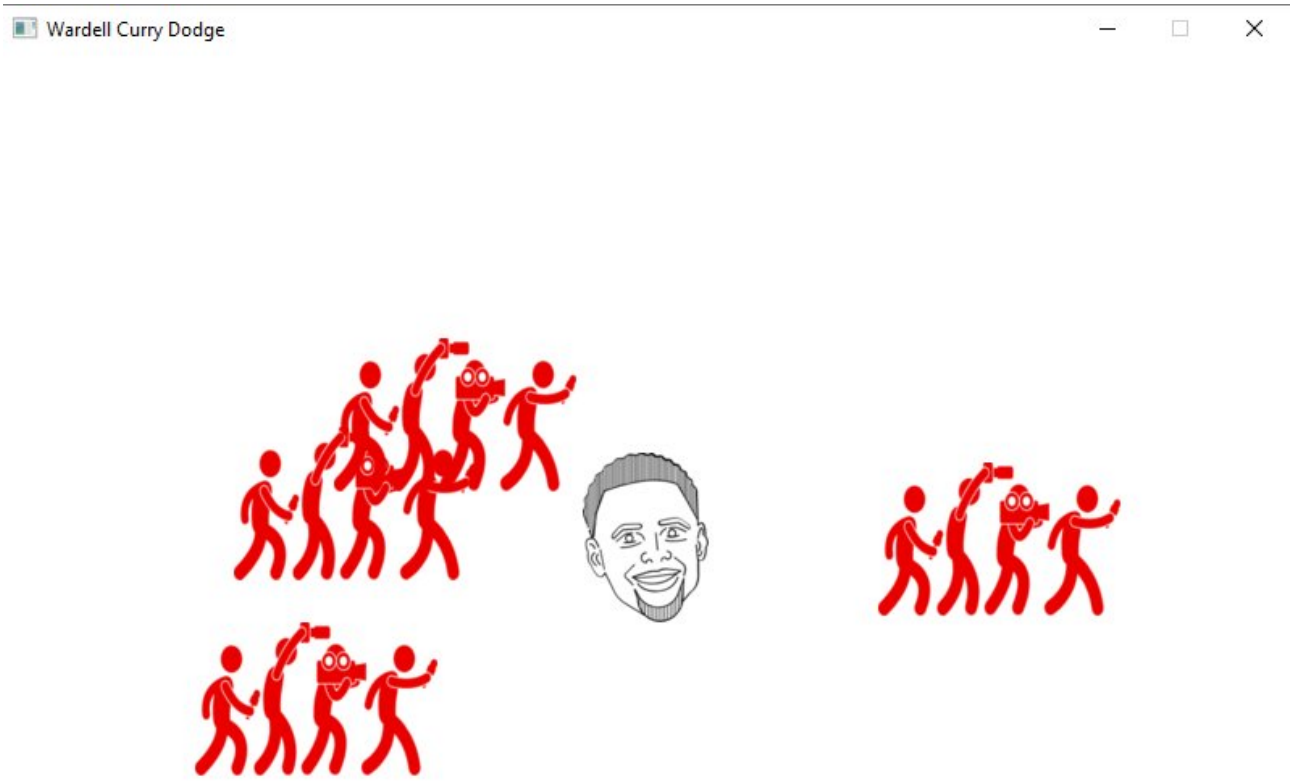
```
using System;
using System.Collections.Generic;
using SplashKitSDK;

public class Robot
{  //Properties
        private double X
        {get;set;}
        private double Y
        {get;set;}
        private Bitmap bitmap
        {get;set;}
        private Vector2D Velocity
        {get;set;}
        public Color MainColor;

    //Robot Collision
        public Circle CollisionCircle
        {
            get
            {
                return SplashKit.CircleAt(X, Y, 20);
            }
        }

    public int Width

    {get
        { return 50;}
    }
    public int Height

    {get
        { return 50;}
    }


    public Robot(Window gameWindow, Player player)
    {
        //Starting Point
        X = 0;
        Y = 0;
        const int SPEED = 4;

        MainColor = Color.RandomRGB(200);

        if (SplashKit.Rnd() < 0.5)
        {

        X = SplashKit.Rnd(gameWindow.Width);

            if (SplashKit.Rnd() < 0.5)

```

```
54                 Y = -Height;
55             else
56                 Y = gameWindow.Height;
57         }
58         Y = SplashKit.Rnd(gameWindow.Height);
59         if (SplashKit.Rnd() < 0.5)
60 
61             X = -Width;
62         else
63             X = gameWindow.Width;
64 
65          //Get a Point for the Robot
66         Point2D fromPt = new Point2D()
67             {
68                 X = X, Y = Y
69             };
70 
71         //Get a Point for the player
72         Point2D toPt = new Point2D()
73             {
74                 X = player.X, Y = player.Y
75             };
76 
77         //Calculate the direction to head
78             Vector2D dir;
79             dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
80 
81         //Set speed and assign to the Velocity
82             Velocity = SplashKit.VectorMultiply(dir, SPEED);
83 
84         //Get Random location for robot
85         //X = SplashKit.Rnd(gameWindow.Width - Width);
86         //Y = SplashKit.Rnd(gameWindow.Height - Height);
87     }
88 
89      //Create a robot
90     public void Draw()
91         {
92             //Local Variables
93             double leftX;
94             double rightX;
95             double eyeY;
96             double mouthY;
97             Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
98 
99             //Assign LV
100            leftX = X + 12;
101            rightX = X + 27;
102            eyeY = Y + 10;
103            mouthY = Y + 30;
104 
105            //Draw a robot
106            // SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
```

```
107             // SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
108             // SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
109             // SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
110             // SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
111             SplashKit.DrawBitmap(ppr,X, Y);
112         }
113
114     //Update Method
115     public void Update()
116     {
117
118             X += Velocity.X;
119             Y += Velocity.Y;
120     }
121
122     // The robot start from offscreen
123     public bool isOffscreen(Window screen)
124     {
125         if (X < -Width || X > screen.Width || Y < -Height || Y > screen.Height)
126         {
127             return true;
128         }
129         else
130         {
131             return false;
132         }
133     }
134
135
136
137
138 }
```

```
using System;
using System.Collections.Generic;
using SplashKitSDK;


public class Robotdodge
{
    //Private Field
    private Player _Player;
    private Window _GameWindow;
    private static List<Robot> _Robots = new List<Robot>();

    //private Robot _TestRobot;

    //Property
    public bool Quit
    {
        get { return _Player.quit; }
    }

     //Constructor
    public Robotdodge(Window gameWindow, Player player)
    {
        _GameWindow = gameWindow;
        _Player = player;

    }
    public void HandleInput()
    {
        _Player.HandleInput();
        _Player.StayOnWindow(_GameWindow);
    }

    //For Draw Method
    public void Draw()
    {
        //_GameWindow.Clear(Color.White);

        foreach (Robot robot in _Robots)
        {
            robot.Draw();
        }
        _Player.Draw();
        //_TestRobot.Draw();
        //_GameWindow.Refresh(60);
    }
    //For Update Method
    public void Update(Window gameWindow)
    {
        // This if stament when player hit the robot
        // if (_Player.CollidedWith(_Robots))
        // {
        //     Robot RandomRobot = new Robot(gameWindow);
```

```
54          //      _Robots = RandomRobot;
55          // }
56          foreach (Robot robot in _Robots)
57          {
58              robot.Update();
59          }
60          if (SplashKit.Rnd() < 0.05)
61          {
62              _Robots.Add(RandomRobot());
63          }
64          CheckCollisions();
65      }
66      public Robot RandomRobot()
67      {
68          return new Robot(_GameWindow, _Player);
69      }
70
71      //Private Method CheckCollisions
72      private void CheckCollisions()
73      {
74          List<Robot> _deleteRobots = new List<Robot>();
75
76          foreach (Robot robots in _Robots)
77          {
78              if (_Player.CollidedWith(robots) || robots.isOffscreen(_GameWindow))
79              {
80                  _deleteRobots.Add(robots);
81              }
82          }
83          foreach (Robot robots in _deleteRobots)
84          {
85              _Robots.Remove(robots);
86          }
87      }
88  }
```


Wardell Curry Dodge

22 Document Design

Create appropriate documentation for the Account class

Outcome	Weight
Design	♦♦♦♦◊

This UML is all about summary of stock management inventory program

Outcome	Weight
Justify	♦♦♦◊◊

This UML is all about summary of stock management inventory program

Date	Author	Comment
2019/08/19 17:24	Achmad Kemal	Can i use another software for this task?
2019/08/19 17:31	Achmad Kemal	For drawing an UML diagram
2019/08/19 17:33	Mengmeng Ge	Yes. You could.
2019/08/19 21:02	Achmad Kemal	Ready to Mark
2019/08/19 21:24	Mengmeng Ge	Arrows point to Stock class from transaction classes should be replaced with the one same as arrow from WareHouse to Stock. There is no solid arrow representation for any relationship. Please check the following link for different line notations.
2019/08/19 21:24	Mengmeng Ge	https://medium.com/@smagid_allThings/uml-class-diagrams-tutorial-step-by-step-520fd83b300b
2019/08/19 21:25	Mengmeng Ge	Fix and Resubmit
2019/08/25 16:03	Achmad Kemal	Ready to Mark
2019/08/25 16:04	Achmad Kemal	please give me a feedback
2019/08/25 19:47	Achmad Kemal	Ready to Mark
2019/08/26 08:59	Mengmeng Ge	Relationships between warehouse and program, between menuoption and program should be dependency. You could check the lecture notes for week 7 I uploaded. The first slide is about class relationship.
2019/08/26 08:59	Mengmeng Ge	Fix and Resubmit
2019/08/26 09:43	Achmad Kemal	i'm so sorry
2019/08/26 09:43	Achmad Kemal	i will resubmit again
2019/08/26 09:49	Achmad Kemal	Ready to Mark
2019/08/26 10:59	Mengmeng Ge	Excellent!
2019/08/26 11:00	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Document Design

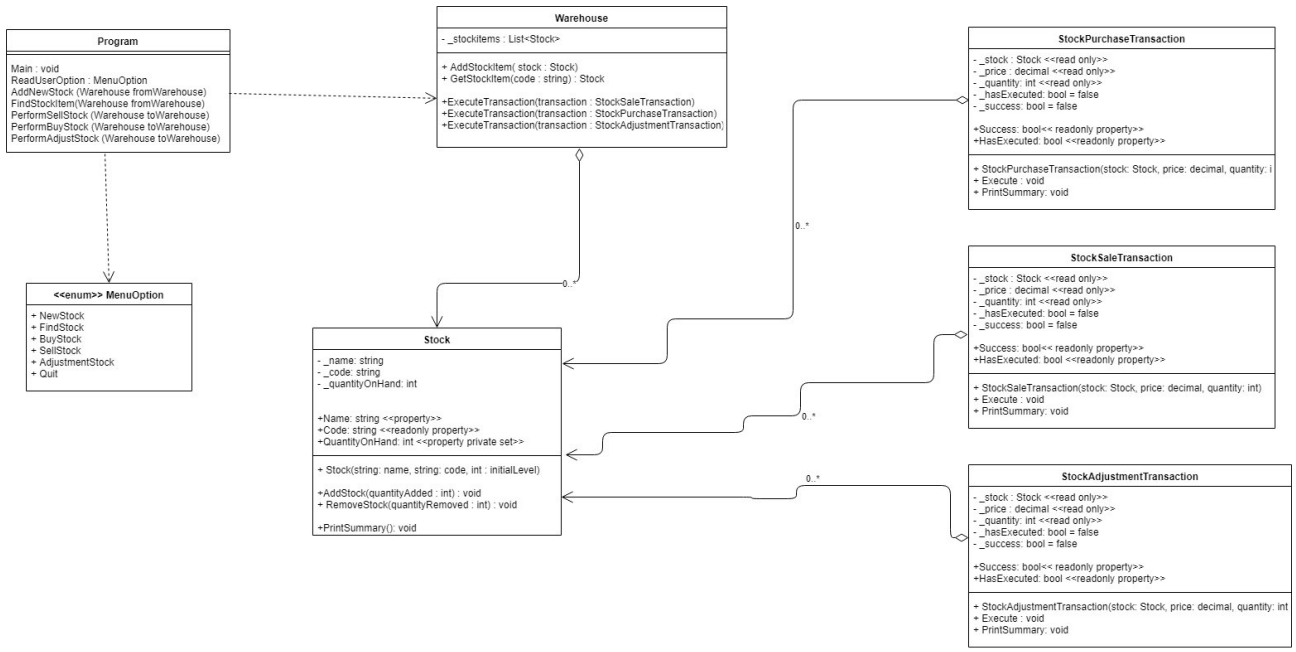
Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/26 09:49

Tutor:
Mengmeng Ge

Outcome	**Weight**
Design	♦♦♦♦◊
Justify	♦♦♦◊◊

This UML is all about summary of stock management inventory program

August 26, 2019


Program
Main : void
ReadUserOption : MenuOption
AddNewStock (Warehouse fromWarehouse)
FindStockItem(Warehouse fromWarehouse)
PerformSellStock (Warehouse toWarehouse)
PerformBuyStock (Warehouse toWarehouse)
PerformAdjustStock (Warehouse toWarehouse)
Warehouse
- _stockItems : List<Stock>
+ AddStockItem(stock : Stock)
+ GetStockItem(code : string) : Stock
+ExecuteTransaction(transaction : StockSaleTransaction)
+ExecuteTransaction(transaction : StockPurchaseTransaction)
+ExecuteTransaction(transaction : StockAdjustmentTransaction)
StockPurchaseTransaction
- _stock : Stock <<read only>>
- _price : decimal <<read only>>
- _quantity: int <<read only>>
- _hasExecuted: bool = false
- _success: bool = false
+Success: bool<< readonly property>>
+HasExecuted: bool <<readonly property>>
+ StockPurchaseTransaction(stock: Stock, price: decimal, quantity: i
+ Execute : void
+ PrintSummary: void
<<enum>> MenuOption
+ NewStock
+ FindStock
+ BuyStock
+ SellStock
+ AdjustmentStock
+ Quit
Stock
- _name: string
- _code: string
- _quantityOnHand: int
+Name: string <<property>>
+Code: string <<readonly property>>
+QuantityOnHand: int <<property private set>>
+ Stock(string: name, string: code, int : initialLevel)
+AddStock(quantityAdded : int) : void
+ RemoveStock(quantityRemoved : int) : void
+PrintSummary(): void
StockSaleTransaction
- _stock : Stock <<read only>>
- _price : decimal <<read only>>
- _quantity: int <<read only>>
- _hasExecuted: bool = false
- _success: bool = false
+Success: bool<< readonly property>>
+HasExecuted: bool <<readonly property>>
+ StockSaleTransaction(stock: Stock, price: decimal, quantity: int)
+ Execute : void
+ PrintSummary: void
StockAdjustmentTransaction
- _stock : Stock <<read only>>
- _price : decimal <<read only>>
- _quantity: int <<read only>>
- _hasExecuted: bool = false
- _success: bool = false
+Success: bool<< readonly property>>
+HasExecuted: bool <<readonly property>>
+ StockAdjustmentTransaction(stock: Stock, price: decimal, quantity: int
+ Execute : void
+ PrintSummary: void
0..*
0..*
0..*
0..*

23 Robot Dodge Changes

Implement changes to how the robot dodges

Outcome	Weight
Build Programs	♦♦♦♦◊

Planning to create a new class in the program is very useful to make the program work successfully

Outcome	Weight
Design	♦♦♦♦◊

Planning to create a new class in the program is very useful to make the program work successfully

Outcome	Weight
Justify	♦♦♦♦◊

Planning to create a new class in the program is very useful to make the program work successfully

Date	Author	Comment
2019/08/26 19:41	Achmad Kemal	Good evening Dr Mengmeng Ge
2019/08/26 19:42	Achmad Kemal	This is my UML diagram in the program
2019/08/26 19:42	Achmad Kemal	when i add bullet class in the program
2019/08/26 19:43	Achmad Kemal	i also explain how bullet works in the program
2019/08/26 19:43	Achmad Kemal	Can you check my PDF?
2019/08/26 19:43	Achmad Kemal	I hope this design will get positive result
2019/08/26 19:44	Achmad Kemal	So i can implement to the program
2019/08/26 19:44	Achmad Kemal	Thanks
2019/08/26 19:44	Achmad Kemal	Please give a feedback about my document design
2019/08/26 20:15	Achmad Kemal	pdf document
2019/08/26 20:53	Mengmeng Ge	You will need to add score and life into the design as well.
2019/08/26 20:53	Mengmeng Ge	Which method will be used to shoot the bullet?
2019/08/26 20:54	Mengmeng Ge	If you read the instruction, the bullet should shoot from player to robot.
2019/09/01 21:46	Achmad Kemal	pdf document
2019/09/01 21:46	Achmad Kemal	This is my updated UML
2019/09/01 21:47	Achmad Kemal	I hope this UML is releveant with this task
2019/09/02 09:37	Mengmeng Ge	In the Bullet class, you may also want to add the method of checking collision when bullet hit the robot. Besides, it might be better to declare a bullet list in RobotDodge class. Everytime the user shoots a bullet, a bullet object is added to the list; everytime a bullet hits a robot, the bullect object is removed from the list.
2019/09/02 13:17	Achmad Kemal	pdf document
2019/09/02 13:18	Achmad Kemal	i already edit my UML
2019/09/02 15:02	Mengmeng Ge	Complete
2019/09/02 15:07	Mengmeng Ge	Fix and Resubmit
2019/09/02 15:08	Achmad Kemal	Because i will not able to done this code by the end of day. Please grant my request
2019/09/08 14:16	Achmad Kemal	Ready to Mark
2019/09/08 14:35	Achmad Kemal	https://youtu.be/32gNooFSDh0
2019/09/08 14:36	Achmad Kemal	This is my video about this task
2019/09/08 16:51	Mengmeng Ge	Shooting bullets looks odd. Everytime you click the mouse, there will be one bullet shot from the player. You could double check whether you clear the window before you draw.
2019/09/08 16:52	Mengmeng Ge	Fix and Resubmit
2019/09/08 19:36	Achmad Kemal	Ready to Mark
2019/09/08 19:39	Achmad Kemal	https://youtu.be/VxiVIWOCE_o
2019/09/08 19:40	Achmad Kemal	this is updated video for robot bullet
2019/09/09 08:08	Mengmeng Ge	In Program class, you could move line 26 before draw method (21). The sequence is clear -> draw -> refresh.
2019/09/09 08:08	Mengmeng Ge	Fix and Resubmit
2019/09/09 22:08	Achmad Kemal	Ready to Mark
2019/09/09 22:18	Achmad Kemal	https://youtu.be/I0po08SfOi0
2019/09/09 22:18	Achmad Kemal	this is updated video for robot bullet
2019/09/10 08:33	Mengmeng Ge	Well done! How did you manage to use MouseClicked?
2019/09/10 08:33	Mengmeng Ge	Complete
2019/09/10 09:04	Achmad Kemal	Because of Splashkit ProcessEvent, i can't use mouse clicked and then i delete splashkit process in the Player class. Now, i can use mouse click for shot the bullet

Deakin University

Object Oriented Development

OnTrack Submission

Robot Dodge Changes

Submitted By:
Achmad Mustafa Kemal
akemal
2019/09/09 22:08

Tutor:
Mengmeng Ge

Outcome	Weight
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊
Justify	♦♦♦♦◊

Planning to create a new class in the program is very useful to make the program work successfully

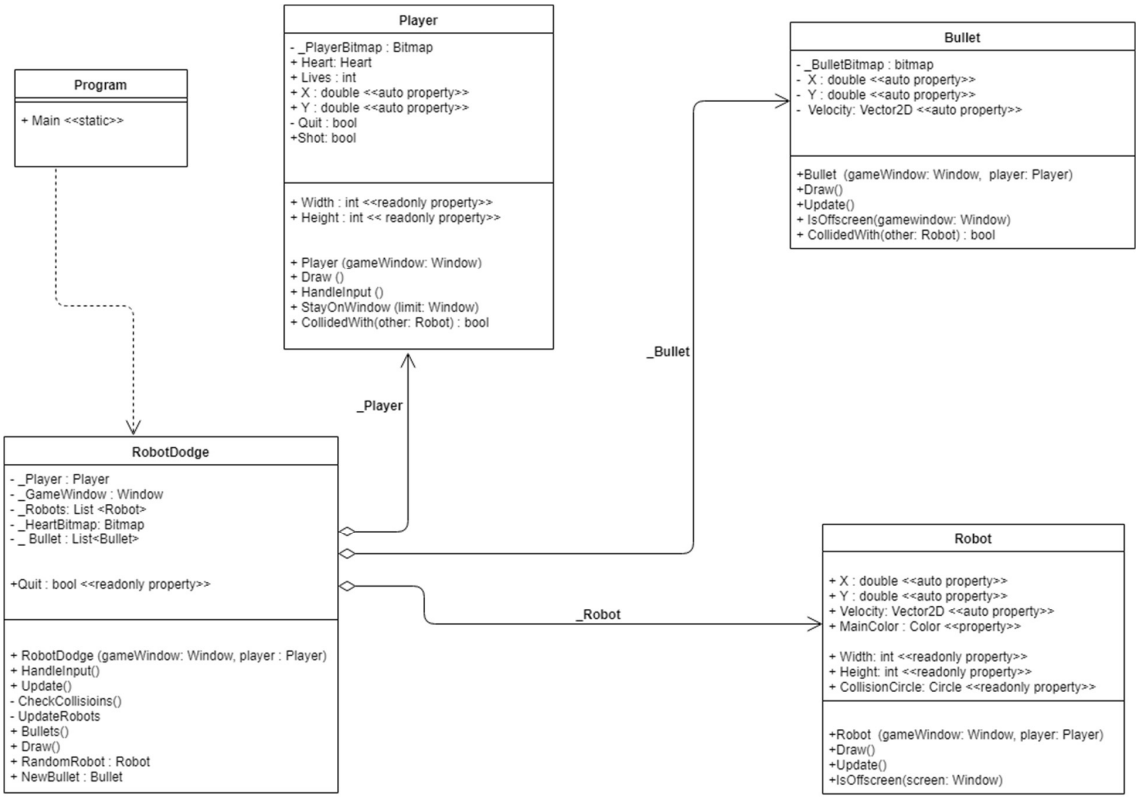
September 9, 2019

SIT771

Task 6.2

ACHMAD MUSTAFA KEMAL

219374683



This is my UML diagram when add Bullet class in the program. So, player will have bullet to shoot the robot. The bullet will be unlimited when the user want to use bullet to remove many robots in the program. The model of bullet will be like a basketball because the player character is Steph Curry (NBA player). The bullet will shot many angle in the game. The robot will be collided with the bullet if the user shoot a bullet to robot. The bullet can be drawing a shield for player. So, the player will be protected from robot when player draw a bullet.

```
using System;
using SplashKitSDK;

public class Program
{
    public static Timer myScore = new Timer ("My Timer");

    public static void Main()
    {
        Window gameWindow;
        gameWindow = new Window("Wardell Curry Dodge", 800, 600);
        Player player = new Player (gameWindow);
        myScore.Start();
        Robotdodge robotdodge = new Robotdodge(gameWindow, player);
        do
        {
            SplashKit.ProcessEvents();
            //player.StayOnWindow(gameWindow);
            robotdodge.HandleInput();
            gameWindow.Clear(Color.White);
            robotdodge.Update(gameWindow);
            robotdodge.Draw();
            gameWindow.Refresh(60);
            //player.Draw();
            //robotdodge.RandomRobot();
            SplashKit.DrawTextOnWindow(gameWindow, Convert.ToString(myScore.Ticks /
            ↪  1000), Color.Black, 50, 50);
            Console.WriteLine($"{myScore.Ticks} miliseconds have passed");
            Console.WriteLine($"Which is {myScore.Ticks/ 1000} seconds");
        }while(!robotdodge.Quit);
    }
}
```

```
using System;
using SplashKitSDK;

public class Player
{
    private Bitmap _playerBitmap;
    private Bitmap Heart;
    public int Lives = 5;

    private double _x;
    private bool _quit = false;
    public bool shot = false;
    public double X
    {
        get {return _x;}
        set { _x = value;}
    }
    private double _y;
    public double Y
    {
        get {return _y;}
        set { _y = value;}
    }
    public bool quit
    {
            get
            {return _quit;}
            private set
            {_quit = value;}
    }

    public void Move (double dx, double dy)
     {
        _x = _x + dx;
        _y = _y + dy;
     }
    public int Width
    {
        get
        {
            return _playerBitmap.Width;
        }
    }

    public int Height
    {
        get
        {
            return _playerBitmap.Height;
        }
    }

    public Player(Window gameWindow)
```

```
54      {
55          _playerBitmap = new Bitmap("curry", "curry4.png");
56          Heart = new Bitmap("heart", "heart.png");
57          X = (gameWindow.Width - Width) / 2;
58          Y = (gameWindow.Height - Height) / 2;
59      }
60
61      public void Draw()
62      {
63          _playerBitmap.Draw(X, Y);
64
65          for (int i = 1; i <= Lives; i++)
66          {
67              SplashKit.DrawBitmap(Heart, 750 - (i*35), 35);
68          }
69
70      }
71
72      public void HandleInput()
73      {
74
75          //SplashKit.ProcessEvents();
76          const int SPEED = 5;
77          // Arrow Keys for move function
78          // X is from public player
79          // Y is from public player
80          if (SplashKit.KeyDown(KeyCode.RightKey))
81          {
82              X += SPEED;
83          }
84          if (SplashKit.KeyDown(KeyCode.LeftKey))
85          {
86              X -= SPEED;
87          }
88
89          if (SplashKit.KeyDown(KeyCode.UpKey))
90          {
91              Y -= SPEED;
92          }
93          if (SplashKit.KeyDown(KeyCode.DownKey))
94          {
95              Y += SPEED;
96          }
97          // Key for quit button
98          if (SplashKit.KeyDown(KeyCode.EscapeKey))
99          {
100             quit = true;
101         }
102         if (SplashKit.MouseClicked(MouseButton.LeftButton))
103         {
104             shot = true;
105         }
106
```

```
107          if (Lives == 0)
108         {
109             quit = true;
110         }
111     }
112 
113     //This is for make live reduce
114     public void LiveLose()
115     {
116             Lives--;
117 
118     }
119     // This is for player no to go out from windows
120     public void StayOnWindow(Window gameWindow)
121     {
122     const int GAP = 10;
123     // X is from public player
124     // Y is from public player
125     if (X < GAP)
126     {
127      X = GAP;
128     }
129     if (X > gameWindow.Width - Width - GAP)
130     {
131     X = gameWindow.Width - Width - GAP;
132     }
133     if (Y < GAP)
134     {
135     Y = GAP;
136     }
137     if (Y > gameWindow.Height - Height - GAP)
138     {
139     Y = gameWindow.Height - Height - GAP;
140     }
141     }
142 
143   //This method for Collision Function
144     public bool CollidedWith(Robot other)
145     {
146         bool collide =_playerBitmap.CircleCollision(X,Y, other.CollisionCircle);
147 
148         // List<Bullet> _deleteBullets = new List<Bullet>();
149 
150         // foreach (Bullet bullets in _Bullet)
151         // {
152         //     if (bullets.CollidedWith(other))
153         //     {
154         //         _deleteBullets.Add(bullets);
155         //         collide = true;
156         //     }
157         // }
158 
159         // foreach (Bullet bullets in _deleteBullets)
```

```
160         // {
161         //     _Bullet.Remove(bullets);
162 
163         // }
164 
165         return collide;
166     }
167     // public void Bullets()
168     // {
169     // _Bullets.Update();
170     // }
171 }
```

```
using System;
using System.Collections.Generic;
using SplashKitSDK;

public abstract class Robot
{  //Properties
        public double X
        {get;set;}
        public double Y
        {get;set;}
        private Bitmap bitmap
        {get;set;}
        private Vector2D Velocity
        {get;set;}
        public Color MainColor;

    //Robot Collision
        public Circle CollisionCircle
        {
            get
            {
                return SplashKit.CircleAt(X, Y, 20);
            }
        }

    public int Width

    {get
        { return 50;}
    }
    public int Height

    {get
        { return 50;}
    }


    public Robot(Window gameWindow, Player player)
    {
        //Starting Point
        X = 0;
        Y = 0;
        const int SPEED = 4;

        MainColor = Color.RandomRGB(200);

        if (SplashKit.Rnd() < 0.5)
        {

        X = SplashKit.Rnd(gameWindow.Width);

            if (SplashKit.Rnd() < 0.5)

```

```
                Y = -Height;
            else
                Y = gameWindow.Height;
        }
        Y = SplashKit.Rnd(gameWindow.Height);
        if (SplashKit.Rnd() < 0.5)

            X = -Width;
        else
            X = gameWindow.Width;

         //Get a Point for the Robot
        Point2D fromPt = new Point2D()
            {
                X = X, Y = Y
            };

        //Get a Point for the player
        Point2D toPt = new Point2D()
            {
                X = player.X, Y = player.Y
            };

        //Calculate the direction to head
            Vector2D dir;
            dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));

        //Set speed and assign to the Velocity
            Velocity = SplashKit.VectorMultiply(dir, SPEED);

        //Get Random location for robot
        //X = SplashKit.Rnd(gameWindow.Width - Width);
        //Y = SplashKit.Rnd(gameWindow.Height - Height);
    }

//     //Create a robot
//         public void Draw()
//        {
//               //Local Variables
//               double leftX;
//               double rightX;
//               double eyeY;
//               double mouthY;
// //              //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");

// //              //Assign LV
//               leftX = X + 12;
//               rightX = X + 27;
//               eyeY = Y + 10;
//               mouthY = Y + 30;

// //              //Draw a robot
//               SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
```

```
107 //              SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
108 //              SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
109 //                 SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
110 //              SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
111 //             //SplashKit.DrawBitmap(ppr,X, Y);
112 //     }
113
114     public abstract void Draw();
115
116     //Update Method
117     public void Update()
118     {
119
120             X += Velocity.X;
121             Y += Velocity.Y;
122     }
123
124     // The robot start from offscreen
125     public bool isOffscreen(Window screen)
126     {
127         if (X < -Width || X > screen.Width || Y < -Height || Y > screen.Height)
128         {
129             return true;
130         }
131         else
132         {
133             return false;
134         }
135     }
136     }
137     //Draw a Robot Boxy
138     public class Boxy : Robot
139     {
140      public Boxy(Window _Gamewindow ,Player _Player): base (_Gamewindow, _Player){}
141
142    public override void Draw()
143        {
144             //Local Variables
145             double leftX;
146             double rightX;
147             double eyeY;
148             double mouthY;
149     //         //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
150
151     //         //Assign LV
152             leftX = X + 12;
153             rightX = X + 27;
154             eyeY = Y + 10;
155             mouthY = Y + 30;
156
157     //         //Draw a robot
158             SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
159             SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
```

```
160             SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
161             SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
162             SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
163     //          //SplashKit.DrawBitmap(ppr,X, Y);
164         }
165
166      }
167
168     // //Draw a Robot Roundy
169     public class Roundy: Robot
170     {
171      public Roundy(Window _Gamewindow ,Player _Player): base (_Gamewindow,
         ↪  _Player){}
172
173     public override void Draw()
174         {
175             //Local Variables
176             double leftX;
177             double rightX;
178             double midX;
179             double midY;
180             double eyeY;
181             double mouthY;
182     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
183
184     //          //Assign LV for X
185             leftX = X + 17;
186             midX = X + 25;
187             rightX = X + 33;
188
189     //          //Assign LV for Y
190             midY = Y + 25;
191             eyeY = Y + 20;
192             mouthY = Y + 35;
193
194     //          //Draw a robot
195              SplashKit.FillCircle(Color.Red,midX, midY, 25);
196              SplashKit.DrawCircle(Color.Gray, midX,midY, 25);
197              SplashKit.FillCircle(MainColor,leftX, eyeY, 5);
198              SplashKit.FillCircle(MainColor, rightX, eyeY, 5);
199              SplashKit.FillEllipse(Color.Gray, X, eyeY, 50,30);
200              SplashKit.DrawLine(Color.Black,X,mouthY,X + 50,Y + 35);
201     //          //SplashKit.DrawBitmap(ppr,X, Y);
202         }
203
204     }
205
206     //Draw another Robot
207     public class RobotPic : Robot
208      {
209      public RobotPic(Window _Gamewindow ,Player _Player): base (_Gamewindow,
         ↪  _Player){}
210
```

```
211      public override void Draw()
212          {
213     //          //Local Variables
214                  double leftX;
215                  double rightX;
216                  double eyeY;
217                  double mouthY;
218     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
219                Bitmap stephen = new Bitmap("Stephen", "stephen.png");
220
221     //          //Assign LV
222              leftX = X + 12;
223              rightX = X + 27;
224              eyeY = Y + 10;
225              mouthY = Y + 30;
226
227     //          //Draw a robot
228 //    SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
229 //     SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
230 //      SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
231 //    SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
232 //     SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
233     //          //SplashKit.DrawBitmap(ppr,X, Y);
234              SplashKit.DrawBitmap(stephen,X, Y);
235          }
236
237     }
238     // //Draw another Robot
239     public class RobotPic1 : Robot
240     {
241     public RobotPic1(Window _Gamewindow ,Player _Player): base (_Gamewindow,
        ↪  _Player){}
242
243     public override void Draw()
244        {
245     //          //Local Variables
246     //          // double leftX;
247     //          // double rightX;
248     //          // double eyeY;
249     //          // double mouthY;
250     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
251          Bitmap klay = new Bitmap("klay", "klay.png");
252
253     //          //Assign LV
254     //          // leftX = X + 12;
255     //          // rightX = X + 27;
256     //          // eyeY = Y + 10;
257     //          // mouthY = Y + 30;
258
259     //          //Draw a robot
260     //          // SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
261     //          // SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
262     //          // SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
```

```
263     //         // SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
264     // SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
265    //SplashKit.DrawBitmap(ppr,X, Y);
266             SplashKit.DrawBitmap(klay,X, Y);
267         }
268
269         }
270
271
272
273
274
275
```

```
using System;
using SplashKitSDK;



public class Bullet
{
        private Bitmap _BulletBitmap;
        private double X
        {get;set;}
        private double Y
        {get;set;}
        //private double _angle;
        private Vector2D Velocity
        {get;set;}

        //private bool _active  = false;


        public Bullet(Window gameWindow, Player player)
        {

            _BulletBitmap = SplashKit.LoadBitmap("bullet", "bb1.png");
            const int SPEED = 5;


            //This is X and Y to make bullet come out to player
            X = player.X + player.Width  / 2;
            Y = player.Y + player.Height  / 2;



            Point2D fromPt = new Point2D()
            {
               X = X, Y = Y
            };

            //Get a Point for bullet
         Point2D toPt = SplashKit.MousePosition();



        //Calculate the direction to head
            Vector2D dir;
            dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));

        //Set speed and assign to the Velocity
            Velocity = SplashKit.VectorMultiply(dir, SPEED);
        }

        public bool CollidedWith(Robot other)
        {
              return _BulletBitmap.CircleCollision(X,Y, other.CollisionCircle);
```

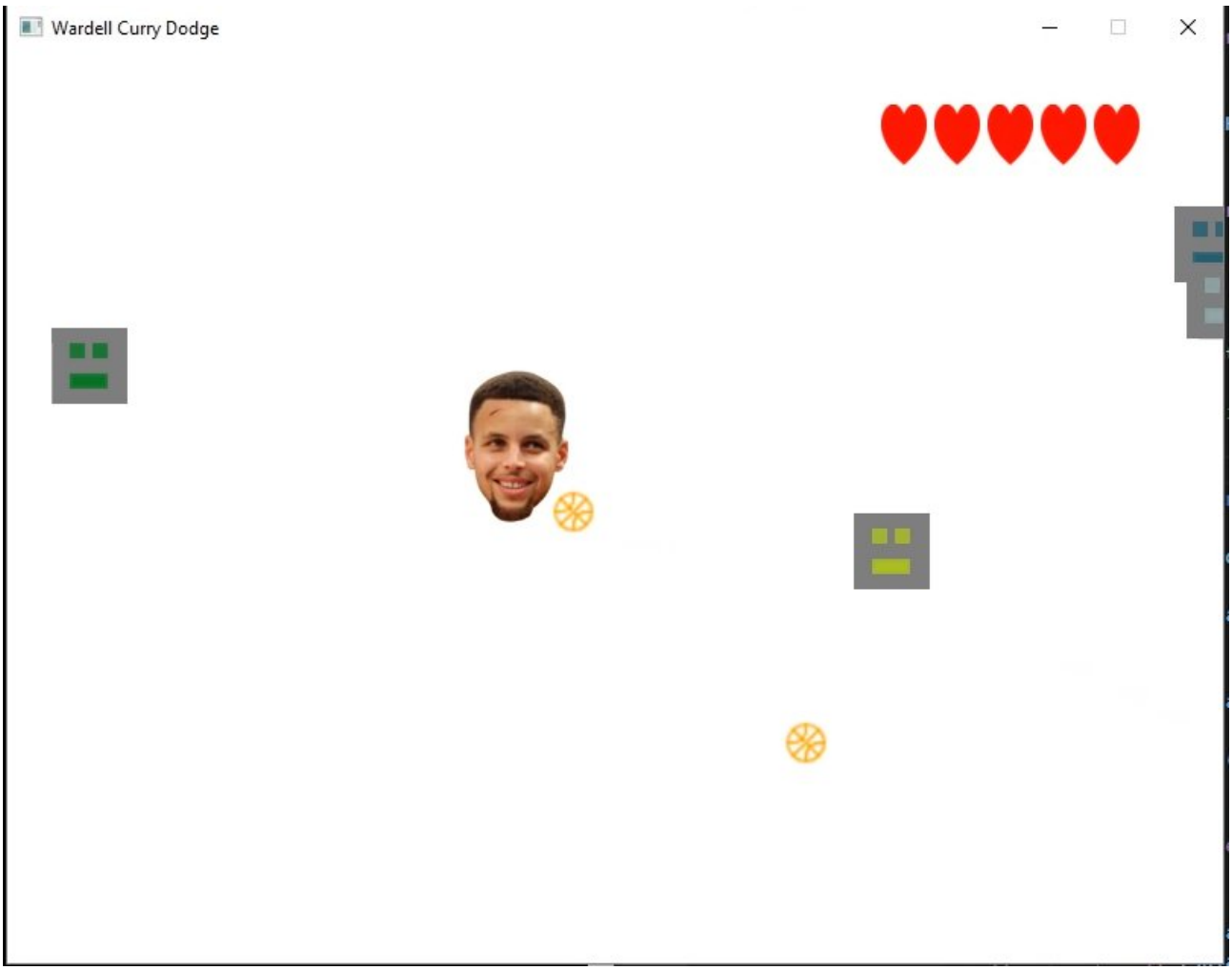
```
54          }
55
56          public void Draw()
57          {
58              SplashKit.DrawBitmap(_BulletBitmap, X,Y);
59          }
60          public void Update()
61          {
62
63              X += Velocity.X;
64              Y += Velocity.Y;
65
66          }
67          public bool isOffScreen (Window gameWindow)
68          {
69          if (X < -50 || X > gameWindow.Width || Y < -50 || Y > gameWindow.Height)
            ↪  return true;
70          return false;
71          }
72
73  }
```

```
using System;
using System.Collections.Generic;
using SplashKitSDK;


public class Robotdodge
{
    //Private Field
    public Player _Player;
    private Window _GameWindow;
    private static List<Robot> _Robots = new List<Robot>();
    private List<Bullet> _Bullets = new List<Bullet>();

   // private Bullet bullets;


    //private Robot _TestRobot;

    //Property
    public bool Quit
    {
        get { return _Player.quit; }
    }

     //Constructor
    public Robotdodge(Window gameWindow, Player player)
    {
       _GameWindow = gameWindow;
       _Player = player;
    }
    public void HandleInput()
    {
        _Player.HandleInput();
        _Player.StayOnWindow(_GameWindow);


        //this if statement make bullet can come out when the user click left
        ↪  button on the mouse
        if (_Player.shot == true)
         {
              _Player.shot = false;
           _Bullets.Add(NewBullet());
        }

        // if (SplashKit.MouseClicked(MouseButton.LeftButton))
        // {
        //     _Player.shot = true;
        //     _Bullets.Add(NewBullet());
        // }
        //  SplashKit.ProcessEvents();
    }
       //For Update Method

```

```
53      //For Draw Method
54      public void Draw()
55      {
56          //_GameWindow.Clear(Color.White);
57
58          //For appaer robot in the window
59          foreach (Robot robot in _Robots)
60          {
61              robot.Draw();
62          }
63           _Player.Draw();
64          //For Score appear in the window
65
66
67          foreach (Bullet bullets in _Bullets)
68          {
69               bullets.Draw();
70          }
71
72          //For Player that appear in the window
73
74          //_TestRobot.Draw();
75          //_GameWindow.Refresh(60);
76      }
77      public void Update(Window gameWindow)
78      {
79          // This if stament when player hit the robot
80          // if (_Player.CollidedWith(_Robots))
81          // {
82          //     Robot RandomRobot = new Robot(gameWindow);
83          //     _Robots = RandomRobot;
84          // }
85          foreach (Robot robot in _Robots)
86          {
87              robot.Update();
88          }
89          if (SplashKit.Rnd() < 0.05)
90          {
91              _Robots.Add(RandomRobot());
92          }
93          foreach (Bullet bullets in _Bullets)
94           {
95              bullets.Update();
96           }
97
98          CheckCollisions();
99      }
100     public Robot RandomRobot()
101     {
102          if (SplashKit.Rnd()< 0.5)
103          {
104              return new Boxy(_GameWindow,_Player);
105          }
```

```
106          else if (SplashKit.Rnd()< 0.5)
107          {
108              return new Roundy(_GameWindow,_Player);
109          }
110          else if (SplashKit.Rnd()< 0.5)
111             {
112              return new RobotPic(_GameWindow,_Player);
113         }
114         else
115          {
116             return new RobotPic1(_GameWindow,_Player);
117         }
118        // return new Robot(_GameWindow, _Player);
119
120     }
121     public Bullet NewBullet()
122     {
123         return new Bullet(_GameWindow, _Player);
124
125     }
126
127     //Private Method CheckCollisions
128     private void CheckCollisions()
129     {
130         List<Robot> _deleteRobots = new List<Robot>();
131         List<Bullet> _deleteBullets = new List<Bullet>();
132
133         foreach (Robot robots in _Robots)
134         {
135             if (_Player.CollidedWith(robots) || robots.isOffscreen(_GameWindow))
136             {
137                 _deleteRobots.Add(robots);
138                 _Player.LiveLose();
139             }
140
141        //To make bullet collided with robot
142          foreach( Bullet bullets in _Bullets)
143            {
144             if (bullets.CollidedWith(robots) || robots.isOffscreen(_GameWindow) )
145                 {
146                 _deleteRobots.Add(robots);
147                 _deleteBullets.Add(bullets);
148                 }
149            }
150         }
151         foreach (Robot robots in _deleteRobots)
152         {
153             _Robots.Remove(robots);
154         }
155         foreach (Bullet bullets in _deleteBullets)
156         {
157             _Bullets.Remove(bullets);
158         }
```

```
159         }
160 }
```


Wardell Curry Dodge

24 Custom Program Pitch

Pitch to us your idea for a custom program

Outcome	Weight
Design	♦♦♦♦◊

This task is about planning our custom program pitch

Outcome	Weight
Justify	♦♦♦♦◊

This task is about planning our custom program pitch

Date	Author	Comment
2019/08/25 20:26	Achmad Kemal	Ready to Mark
2019/08/26 09:00	Mengmeng Ge	Nice design! I'm looking forward to your program up and running!
2019/08/26 09:00	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Custom Program Pitch

Submitted By:
Achmad Mustafa Kemal
akemal
2019/08/25 20:26

Tutor:
Mengmeng Ge

Outcome	**Weight**
Design	♦♦♦♦◊
Justify	♦♦♦♦◊

This task is about planning our custom program pitch

August 25, 2019

SIT771

Task 6.3

Achmad Mustafa Kemal

219374683

For custom program pitch, I will build a C# game based. It will call rocket vs aliens. Basically, the rocket ship must survive from alien in the space atmosphere. The rocket ship will have an ammunition to shoot many aliens. The rocket will have 5 lives to survive from alien invasion. Also, the game provide scoreboard. The scoreboard will reset if the rocket has longer any live and the game will be start again. The development will use Splash Kit SDK. The Splashkit SDK is an API that use for a beginner's all-purpose software toolkit. Also, Splashkit is an open source. So, it's very useful for my custom program pitch.

In this game, there will be four classes in the program. There are program class, player class, game class and alien class.

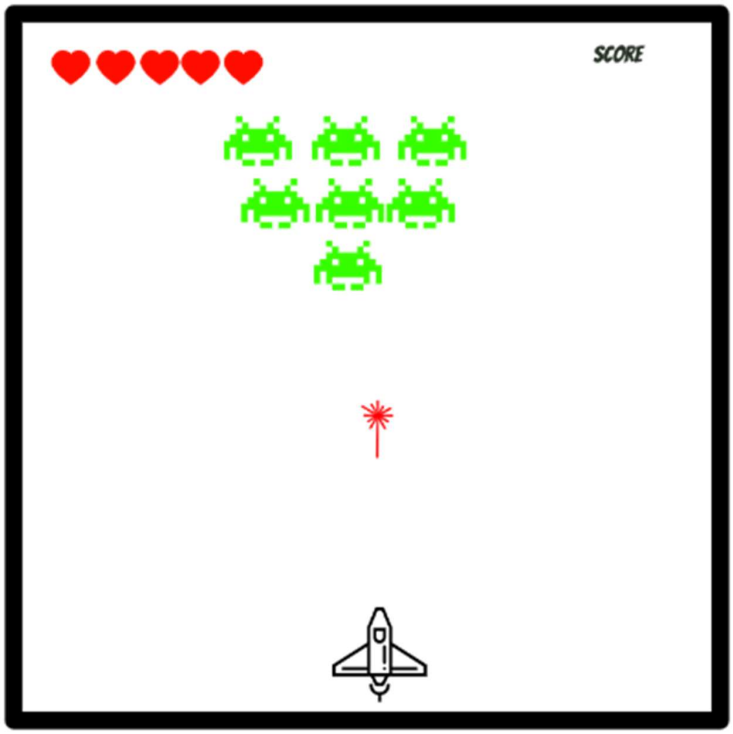


Figure 1.1

This figure is a display of a game that will be developed. That will have background when the game has been started. So, this is my custom program pitch. I hope this project will be done on time.

25 Abstract Transactions

Change all transactions to be abstract

Outcome	Weight
Build Programs	♦♦♦♦◊

Inheritance is very useful for method overloading

Outcome	Weight
Design	♦♦♦♦◊

Inheritance is very useful for method overloading

Outcome	Weight
Justify	♦♦♦♦◊

Inheritance is very useful for method overloading

Date	Author	Comment
2019/09/07 16:03	Achmad Kemal	Ready to Mark
2019/09/07 19:34	Mengmeng Ge	There are a couple of problems:1. In Warehouse class, in ExecuteTransaction method, you will need to add transaction into _transactions list.2. In Transaction class, in Execute method, you will need to check whether the transaction has been executed or not; if not, set _hasExecuted to true.3. In each child transaction class, you already declare _stock, _quantity and _price as protected in the base class which means you can use them in the child class. In constructor method, _stock and _quantity are already initialized in the base class. In Execute method, you will need to call base.Execute() first; “bool” before _success should be removed as it is already declared in the base class. Besides, there is an error in the class diagram. _success should be declared as protected in the base class then it could be modified in the child class.
2019/09/07 19:34	Mengmeng Ge	Fix and Resubmit
2019/09/08 17:48	Achmad Kemal	Ready to Mark
2019/09/08 19:25	Mengmeng Ge	Point 3 is not addressed. You should not remove the code to add/remove stock in each child transaction class. Only ”bool” data type should be removed.
2019/09/08 19:25	Mengmeng Ge	Fix and Resubmit
2019/09/08 21:44	Achmad Kemal	Ready to Mark
2019/09/09 08:10	Mengmeng Ge	We will need to discuss this today as your code is still not correct. You are confused about field and local variable.
2019/09/09 16:35	Achmad Kemal	Ready to Mark
2019/09/09 16:50	Achmad Kemal	Ready to Mark
2019/09/09 16:54	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Abstract Transactions

Submitted By:
Achmad Mustafa Kemal
akemal
2019/09/09 16:50

Tutor:
Mengmeng Ge

Outcome	**Weight**
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊
Justify	♦♦♦♦◊

Inheritance is very useful for method overloading

September 9, 2019

```
using System;
using SplashKitSDK;

public enum MenuOption
{
    NewStock,
    FindStock,
    BuyStock,
    SellStock,
    AdjustmentStock,
    Quit,
}

public class Program
{

private static MenuOption ReadUserOption()
{
    int option;
    Console.WriteLine("1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit");

        do
        {

            try
            {
                string inputText;
                Console.Write("Choose an Option 1-6");
                inputText = Console.ReadLine();
                option = Convert.ToInt32(inputText);
            }
            catch
            {
                option = -1;
            }

        }while(option < 1 || option > 6);
        return (MenuOption)(option -1);
}

    public static void Main()
    {
        MenuOption userSelection;

        Warehouse test = new Warehouse();

        do
        {
            userSelection = ReadUserOption();
            Console.WriteLine(userSelection);
            switch(userSelection)
            {
```

```
53                  case MenuOption.NewStock:
54                  AddNewStock(test);
55                  break;
56
57                  case MenuOption.FindStock:
58                  FindStockItem(test);
59                  break;
60
61                  case MenuOption.BuyStock:
62                  PeformBuyStock(test);
63                  break;
64
65                  case MenuOption.SellStock:
66                  PeformSellStock(test);
67                  break;
68
69                  case MenuOption.AdjustmentStock:
70                  PerformAdjustStock(test);
71                  break;
72              }
73          }while (userSelection != MenuOption.Quit);
74      }
75      private static void AddNewStock(Warehouse fromWarehouse)
76      {
77          Console.WriteLine("Enter The Stock Name:");
78          string name = Console.ReadLine();
79          Console.WriteLine("Enter The Stock Code:");
80          string code = Console.ReadLine();
81          Console.WriteLine("Enter The Stock Quantity:");
82          int quantity = Convert.ToInt32(Console.ReadLine());
83          Stock stock = new Stock(name, quantity, code);
84          // stock.PrintSummary();
85          fromWarehouse.AddStockItem(stock);
86
87      }
88
89      private static Stock FindStockItem(Warehouse fromWarehouse)
90      {
91          Console.Write("Enter stock code:");
92          string code =  Console.ReadLine();
93          Stock result = fromWarehouse.GetStockItem(code);
94
95          if (result == null)
96          {
97              Console.WriteLine($"No stock found with {code}");
98          }
99          return result;
100     }
101
102     private static void PeformSellStock(Warehouse toWarehouse)
103     {
104         Stock stock = FindStockItem(toWarehouse);
105         if (stock == null) return;
```

```
106         decimal price = 0 ;
107         int quantity = 0;
108 
109 
110             Console.WriteLine("Please enter price");
111             price = Convert.ToDecimal(Console.ReadLine());
112             Console.WriteLine("Please enter quantity");
113             quantity = Convert.ToInt32(Console.ReadLine());
114             StockSaleTransaction stocksale = new
                ↪  StockSaleTransaction(stock,price,quantity);
115             // stocksale.Execute();
116             stocksale.PrintSummary();
117             stock.RemoveStock(quantity);
118             //stock.PrintSummary();
119             toWarehouse.ExecuteTransaction(stocksale);
120             //toWarehouse.PrintTransactionHistory();
121 
122 
123     }
124 
125         private static void PeformBuyStock(Warehouse toWarehouse)
126     {
127         Stock stock = FindStockItem(toWarehouse);
128         if (stock == null) return;
129         decimal price = 0;
130         int quantity = 0;
131 
132 
133             Console.WriteLine("Please enter price");
134             price = Convert.ToDecimal(Console.ReadLine());
135             Console.WriteLine("Please enter quantity");
136             quantity = Convert.ToInt32(Console.ReadLine());
137             StockPurchaseTransaction stocksale = new
                ↪  StockPurchaseTransaction(stock, price,quantity);
138             //  stocksale.Execute();
139             stocksale.PrintSummary();
140             stock.AddStock(quantity);
141             //stock.PrintSummary();
142             toWarehouse.ExecuteTransaction(stocksale);
143             //toWarehouse.PrintTransactionHistory();
144 
145     }
146 
147     private static void PerformAdjustStock(Warehouse toWarehouse)
148     {
149          Stock stock = FindStockItem(toWarehouse);
150         if (stock == null) return;
151         decimal price;
152         int quantity;
153 
154 
155             Console.WriteLine("Please enter price");
156             price = Convert.ToDecimal(Console.ReadLine());
```

```
157             Console.WriteLine("Please enter quantity");
158             quantity = Convert.ToInt32(Console.ReadLine());
159             StockAdjustmentTransaction stocksale = new
                ↪  StockAdjustmentTransaction(stock,quantity);
160             // stocksale.Execute();
161             stocksale.PrintSummary();
162             toWarehouse.ExecuteTransaction(stocksale);
163             //toWarehouse.PrintTransactionHistory();
164
165     }
166
167
168 }
```

```
using System;
using SplashKitSDK;
using System.Linq;
using System.Collections;
using System.Collections.Generic;

public class Warehouse
{

    private List<Stock> _stockitems = new List<Stock>();
    private List<Transaction> _transactions = new List<Transaction>();

    //Method to add and get stock
    public void AddStockItem(Stock stock)
    {
        // _stockitems =  new List<Stock>();
        // int numberofstock;
        // Console.WriteLine("Enter size of index:");
        // numberofstock = int.Parse(Console.ReadLine());
       _stockitems.Add(stock);
    //    foreach(Stock stocks in _stockitems)
    //    {
    //        stocks.PrintSummary();
    //    }
    }
    public Stock GetStockItem(string code)
    {
       foreach(Stock stocks in _stockitems)
       {
        //    stocks.PrintSummary();
           if (stocks.Code == code)
           {
            Console.WriteLine(code);
            return stocks;
           }
       }
      return null;
    }

    public void ExecuteTransaction(Transaction transaction)
    {
        _transactions.Add(transaction);
        transaction.Execute();
    }

    public void PrintTransactionHistory()
    {
        foreach (Transaction transaction in _transactions)
        {
            transaction.PrintSummary();
        }
    }

```

```
54
55      //Method Overloading
56      // public void ExecuteTransaction(StockSaleTransaction StockSaleTransaction)
57      // {
58      //     StockSaleTransaction.Execute();
59      // }
60      // public void ExecuteTransaction(StockPurchaseTransaction
        ↪  StockPurchaseTransaction)
61      // {
62      //     StockPurchaseTransaction.Execute();
63      // }
64      //  public void ExecuteTransaction(StockAdjustmentTransaction
        ↪  StockAdjustmentTransaction)
65      // {
66      //     StockAdjustmentTransaction.Execute();
67      // }
68  }
```

```
using System;
using SplashKitSDK;

public class Stock
{
    //field
    private string _name;
    private int _quantityOnHand;
    private string _code;

    //Constructor
    public Stock(string name, int initialLevel, string code)
    {
        _name = name;
        _quantityOnHand = initialLevel;
        _code = code;
    }



    //Property
    public string Name
    {
        get{return _name;}// Get is for read
        set{_name = value;}// Set is for write
    }
    public string Code
    {
        get{return _code;}// Get is for read
        private set{_code = value;}// Set is for write

    }
    public int QuantityOnHand
    {
        get{return _quantityOnHand;}
        private set{ _quantityOnHand = value;}
    }

    //Method to add stock
   public bool AddStock(int quantityAdded)
   {

       if (quantityAdded > 0)
       {
           Console.WriteLine("Print");
           _quantityOnHand = _quantityOnHand + quantityAdded;
           return true;
       }
       else
       {
           return false;
       }
   }
```

```
54          public bool RemoveStock(int quantityAdded)
55      {
56          if (quantityAdded > 0 && quantityAdded < QuantityOnHand)
57          {
58              _quantityOnHand = _quantityOnHand - quantityAdded;
59              return true;
60          }
61          else
62          {
63              return false;
64          }
65      }
66
67       //Method to Print
68       public void PrintSummary()
69       {
70           Console.WriteLine($"{Name}: {QuantityOnHand} With Stock :{Code}");
71       }
72  }
```

```cs
using System;

public class StockSaleTransaction : Transaction
{
    // private readonly new Stock _stock;
    //private readonly new decimal _price;
    // private new int _quantity;

    // private bool _hasExecuted = false;
    // private bool _success = false;

    // public bool HasExecuted
    // {
    //     get
    //     {
    //         return _hasExecuted;
    //     }
    // }

    // public bool Success
    // {
    //     get
    //     {
    //         return _success;
    //     }
    // }

    public StockSaleTransaction(Stock stock, decimal price, int quantity):
    ↪ base(stock, quantity)
    {
        // _stock = stock;
        _price = price;
        // _quantity = quantity;
    }

    public override void Execute()
    {
        // if (HasExecuted)
        // {
        //     throw new InvalidOperationException();
        // }

        // _hasExecuted = true;

        base.Execute();
        bool _success = _stock.RemoveStock( _quantity);


    }

    public override void PrintSummary()
    {
        Console.Write($"SELL - {_stock.Name} x {_quantity} @ ${_price}");
```

```
53          if ( ! HasExecuted)
54          {
55              Console.Write("Proposed");
56          }
57          else if ( ! Success)
58          {
59              Console.Write("Failed");
60          }
61          Console.WriteLine();
62      }
63  }
```

```cs
using System;

public class StockPurchaseTransaction : Transaction
{
    // private readonly new Stock _stock;
    // private readonly new decimal _price;
    // private new int _quantity;

    //private bool _hasExecuted = false;
    //private bool _success = false;

    // public bool HasExecuted
    // {
    //     get
    //     {
    //         return _hasExecuted;
    //     }
    // }

    // public bool Success
    // {
    //     get
    //     {
    //         return _success;
    //     }
    // }

    public StockPurchaseTransaction(Stock stock, decimal price, int quantity) :
    ↪ base(stock, quantity)
    {
        // _stock = stock;
        _price = price;
        // _quantity = quantity;
    }

    public override void Execute()
    {
        // if (HasExecuted)
        // {
        //     throw new InvalidOperationException();
        // }

        // _hasExecuted = true;

        base.Execute();
        _success = _stock.AddStock(_quantity);
    }

    public override void PrintSummary()
    {
        Console.Write(SummaryLine);
        if ( ! HasExecuted)
        {
```

```
53              Console.Write("Proposed");
54          }
55          else if ( ! Success)
56          {
57              Console.Write("Failed");
58          }
59          Console.WriteLine();
60      }
61  }
```

```
using System;

public class StockAdjustmentTransaction : Transaction
{
    // private readonly new Stock _stock;

    // private new int _quantity;

    // private bool _hasExecuted = false;
    // private bool _success = false;

   // private readonly new decimal _price = 0;


    // public bool HasExecuted
    // {
    //     get
    //     {
    //         return _hasExecuted;
    //     }
    // }

    // public bool Success
    // {
    //     get
    //     {
    //         return _success;
    //     }
    // }

    public StockAdjustmentTransaction(Stock stock, int quantity) : base(stock,
    ↪ quantity)
    {
        // _stock = stock;
        // _quantity = quantity;
    }

    public override void Execute()
    {
        // if (HasExecuted)
        // {
        //     throw new InvalidOperationException();
        // }

        // _hasExecuted = true;
        base.Execute();
        if ( _quantity > 0)
        {
             _success = _stock.AddStock(_quantity);

        }
        else
        {
```

```
53                 _success = _stock.RemoveStock(- _quantity);
54             }
55
56
57         }
58
59         public override void PrintSummary()
60         {
61              Console.Write(SummaryLine);
62             if ( ! HasExecuted)
63             {
64                 Console.Write("Proposed");
65             }
66             else if ( ! Success)
67             {
68                 Console.Write("Failed");
69             }
70             Console.WriteLine();
71         }
72 }
```

```
using System;


public abstract class Transaction
{
    protected readonly Stock _stock;
    protected decimal _price;
    protected readonly int _quantity;

    private DateTime _dateStamp;

    private bool _hasExecuted = false;
    protected bool _success = false;


    public bool HasExecuted
    {
        get
        {
            return _hasExecuted;
        }
    }

    public bool Success
    {
        get
        {
            return _success;
        }
    }

    protected string SummaryLine
    {
        get
        {return $"Purchase - {_stock.Name} x {_quantity} @ ${_price}";}
        // if ( ! HasExecuted)
        // {
        //     Console.Write("Proposed");
        // }
        // else if ( ! Success)
        // {
        //     Console.Write("Failed");
        // }
        // Console.WriteLine();
        // }
    }


    public Transaction(Stock stock, int quantity)
    {
        _stock = stock;
        _quantity = quantity;
    }
```

```
54
55      public virtual void Execute()
56      {
57          _dateStamp = DateTime.Now;
58           if (HasExecuted)
59          {
60               throw new InvalidOperationException();
61          }
62
63           _hasExecuted = false;
64
65      }
66      public abstract void PrintSummary();
67
68  }
```

```
Kemal@MSI MINGW64 /d/users/kemal/documents/code/warehouse53
$ skm dotnet run
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-61
NewStock
Enter The Stock Name:
Razer
Enter The Stock Code:
1337
Enter The Stock Quantity:
12
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-62
FindStock
Enter stock code:1234
No stock found with 1234
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-63
BuyStock
Enter stock code:1337
1337
Please enter price
12
Please enter quantity
2
Purchase - Razer x 2 @ $12Proposed
Print
Print
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-64
SellStock
Enter stock code:1234
No stock found with 1234
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-62
FindStock
Enter stock code:1337
1337
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-64
SellStock
Enter stock code:1337
1337
Please enter price
3
Please enter quantity
6
SELL - Razer x 6 @ $3Proposed
1. New Stock, 2. Find Stock, 3. Buy Stock, 4. Sell Stock, 5. Adjust Stock 6. Quit
Choose an Option 1-6 5
AdjustmentStock
Enter stock code:1337
1337
Please enter price
12
Please enter quantity
5
Purchase - Razer x 5 @ $0Proposed
Print
```

26 Different Robots

Dadd different kinds of robots

Outcome	Weight
Principles	♦♦♦♦◊

Subclass is useful for make different

Outcome	Weight
Build Programs	♦♦♦♦◊

Subclass is useful for make different

Outcome	Weight
Justify	♦♦♦♦◊

Subclass is useful for make different

Date	Author	Comment
2019/09/09 16:24	Achmad Kemal	Because i need to time
2019/09/09 17:17	Achmad Kemal	Ready to Mark
2019/09/09 17:20	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Different Robots

Submitted By:
Achmad Mustafa Kemal
akemal
2019/09/09 17:17

Tutor:
Mengmeng Ge

Outcome	**Weight**
Principles	♦♦♦♦◊
Build Programs	♦♦♦♦◊
Justify	♦♦♦♦◊

Subclass is useful for make different

September 9, 2019

```
using System;
using SplashKitSDK;

public class Program
{
    public static Timer myScore = new Timer ("My Timer");

    public static void Main()
    {
        Window gameWindow;

        gameWindow = new Window("Wardell Curry Dodge", 800, 600);
        Player player = new Player (gameWindow);
        myScore.Start();
        Robotdodge robotdodge = new Robotdodge(gameWindow, player);
        do
        {
            SplashKit.ProcessEvents();
            //player.StayOnWindow(gameWindow);
            robotdodge.HandleInput();
            gameWindow.Clear(Color.White);
            robotdodge.Draw();
            gameWindow.Refresh(60);
            //player.Draw();
            robotdodge.Update(gameWindow);
            //robotdodge.RandomRobot();
            SplashKit.DrawTextOnWindow(gameWindow, Convert.ToString(myScore.Ticks /
            ↪  1000), Color.Black, 50, 50);
            Console.WriteLine( $ "{myScore.Ticks} miliseconds have passed");
            Console.WriteLine( $ "Which is {myScore.Ticks/ 1000} seconds");
        }while(!robotdodge.Quit);
    }
}
```

```
using System;
using System.Collections.Generic;
using SplashKitSDK;

public abstract class Robot
{  //Properties
        public double X
        {get;set;}
        public double Y
        {get;set;}
        private Bitmap bitmap
        {get;set;}
        private Vector2D Velocity
        {get;set;}
        public Color MainColor;

    //Robot Collision
        public Circle CollisionCircle
        {
            get
            {
                return SplashKit.CircleAt(X, Y, 20);
            }
        }

    public int Width

    {get
        { return 50;}
    }
    public int Height

    {get
        { return 50;}
    }


    public Robot(Window gameWindow, Player player)
    {
        //Starting Point
        X = 0;
        Y = 0;
        const int SPEED = 4;

        MainColor = Color.RandomRGB(200);

        if (SplashKit.Rnd() < 0.5)
        {

        X = SplashKit.Rnd(gameWindow.Width);

            if (SplashKit.Rnd() < 0.5)

```

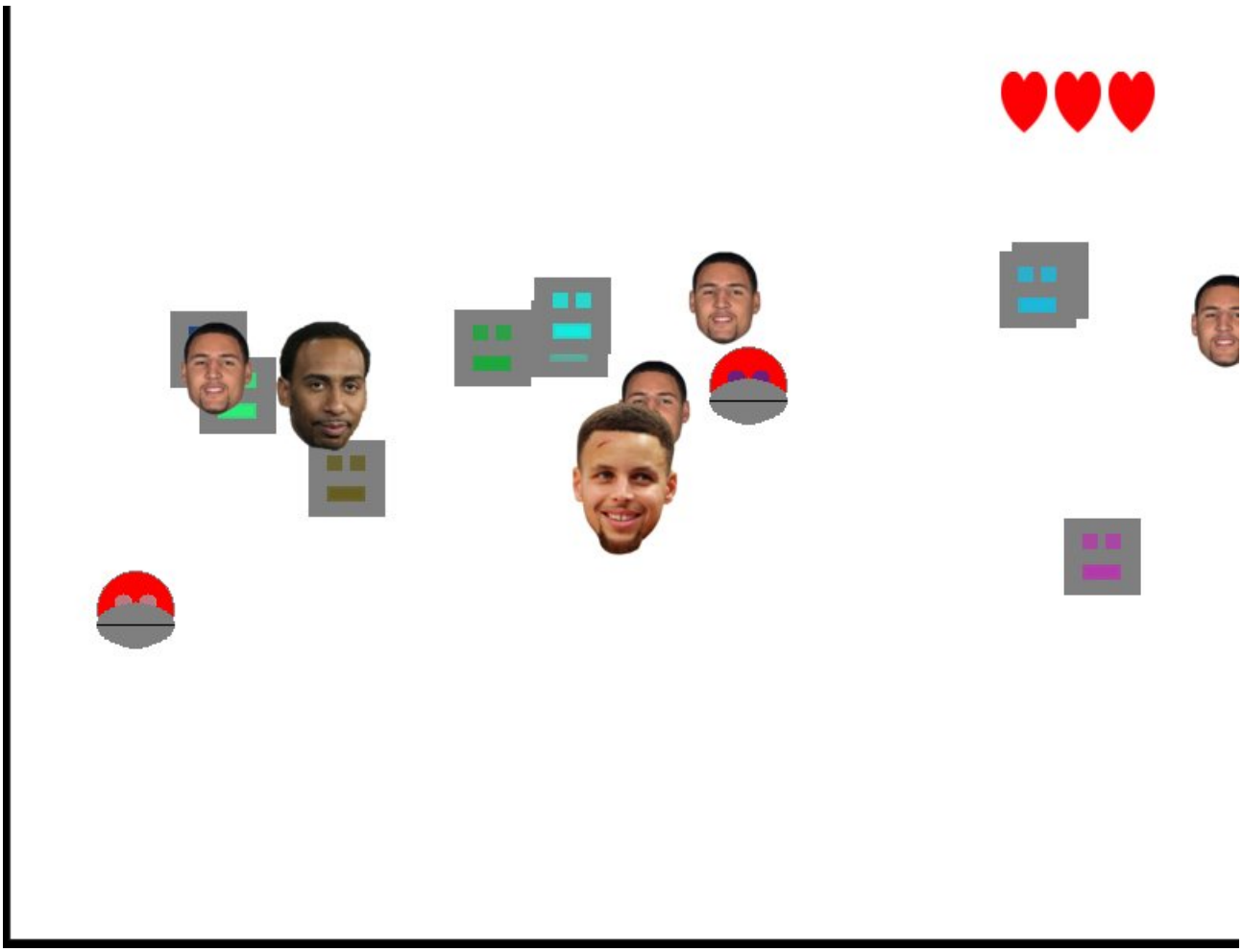
```
54                  Y = -Height;
55              else
56                  Y = gameWindow.Height;
57          }
58          Y = SplashKit.Rnd(gameWindow.Height);
59          if (SplashKit.Rnd() < 0.5)
60
61              X = -Width;
62          else
63              X = gameWindow.Width;
64
65           //Get a Point for the Robot
66          Point2D fromPt = new Point2D()
67              {
68                  X = X, Y = Y
69              };
70
71          //Get a Point for the player
72          Point2D toPt = new Point2D()
73              {
74                  X = player.X, Y = player.Y
75              };
76
77          //Calculate the direction to head
78              Vector2D dir;
79              dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
80
81          //Set speed and assign to the Velocity
82              Velocity = SplashKit.VectorMultiply(dir, SPEED);
83
84          //Get Random location for robot
85          //X = SplashKit.Rnd(gameWindow.Width - Width);
86          //Y = SplashKit.Rnd(gameWindow.Height - Height);
87      }
88
89  //      //Create a robot
90  //          public void Draw()
91  //         {
92  //                //Local Variables
93  //                double leftX;
94  //                double rightX;
95  //                double eyeY;
96  //                double mouthY;
97  // //               //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
98
99  // //               //Assign LV
100 //                leftX = X + 12;
101 //                rightX = X + 27;
102 //                eyeY = Y + 10;
103 //                mouthY = Y + 30;
104
105 // //               //Draw a robot
106 //                SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
```

```
107 //              SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
108 //              SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
109 //                 SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
110 //              SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
111 //             //SplashKit.DrawBitmap(ppr,X, Y);
112 //     }
113
114     public abstract void Draw();
115
116     //Update Method
117     public void Update()
118     {
119
120             X += Velocity.X;
121             Y += Velocity.Y;
122     }
123
124     // The robot start from offscreen
125     public bool isOffscreen(Window screen)
126     {
127         if (X < -Width || X > screen.Width || Y < -Height || Y > screen.Height)
128         {
129             return true;
130         }
131         else
132         {
133             return false;
134         }
135     }
136     }
137     //Draw a Robot Boxy
138     public class Boxy : Robot
139     {
140      public Boxy(Window _Gamewindow ,Player _Player): base (_Gamewindow, _Player){}
141
142    public override void Draw()
143        {
144              //Local Variables
145              double leftX;
146             double rightX;
147              double eyeY;
148              double mouthY;
149     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
150
151     //          //Assign LV
152             leftX = X + 12;
153               rightX = X + 27;
154              eyeY = Y + 10;
155              mouthY = Y + 30;
156
157     //          //Draw a robot
158             SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
159               SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
```

```
160             SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
161               SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
162              SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
163     //          //SplashKit.DrawBitmap(ppr,X, Y);
164         }
165
166      }
167
168     // //Draw a Robot Roundy
169     public class Roundy: Robot
170     {
171      public Roundy(Window _Gamewindow ,Player _Player): base (_Gamewindow,
         ↪  _Player){}
172
173     public override void Draw()
174         {
175             //Local Variables
176             double leftX;
177             double rightX;
178             double midX;
179             double midY;
180             double eyeY;
181             double mouthY;
182     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
183
184     //          //Assign LV for X
185             leftX = X + 17;
186             midX = X + 25;
187             rightX = X + 33;
188
189     //          //Assign LV for Y
190         midY = Y + 25;
191           eyeY = Y + 20;
192             mouthY = Y + 35;
193
194     //          //Draw a robot
195             SplashKit.FillCircle(Color.Red,midX, midY, 25);
196             SplashKit.DrawCircle(Color.Gray, midX,midY, 25);
197             SplashKit.FillCircle(MainColor,leftX, eyeY, 5);
198             SplashKit.FillCircle(MainColor, rightX, eyeY, 5);
199             SplashKit.FillEllipse(Color.Gray, X, eyeY, 50,30);
200             SplashKit.DrawLine(Color.Black,X,mouthY,X + 50,Y + 35);
201     //          //SplashKit.DrawBitmap(ppr,X, Y);
202         }
203
204     }
205
206     //Draw another Robot
207     public class RobotPic : Robot
208      {
209      public RobotPic(Window _Gamewindow ,Player _Player): base (_Gamewindow,
         ↪  _Player){}
210
```

```
211      public override void Draw()
212          {
213     //          //Local Variables
214                 double leftX;
215                  double rightX;
216                  double eyeY;
217                  double mouthY;
218     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
219               Bitmap stephen = new Bitmap("Stephen", "stephen.png");
220
221     //          //Assign LV
222      leftX = X + 12;
223     rightX = X + 27;
224     eyeY = Y + 10;
225     mouthY = Y + 30;
226
227     //          //Draw a robot
228 //    SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
229 //     SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
230 //      SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
231 //    SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
232 //     SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
233     //          //SplashKit.DrawBitmap(ppr,X, Y);
234               SplashKit.DrawBitmap(stephen,X, Y);
235          }
236
237     }
238     // //Draw another Robot
239     public class RobotPic1 : Robot
240     {
241     public RobotPic1(Window _Gamewindow ,Player _Player): base (_Gamewindow,
        ↪  _Player){}
242
243     public override void Draw()
244        {
245     //          //Local Variables
246     //          // double leftX;
247     //          // double rightX;
248     //          // double eyeY;
249     //          // double mouthY;
250     //          //Bitmap ppr = new Bitmap("Paparazzi", "paparazzi.png");
251          Bitmap klay = new Bitmap("klay", "klay.png");
252
253     //          //Assign LV
254     //          // leftX = X + 12;
255     //          // rightX = X + 27;
256     //          // eyeY = Y + 10;
257     //          // mouthY = Y + 30;
258
259     //          //Draw a robot
260     //          // SplashKit.FillRectangle(Color.Gray,X, Y, 50, 50);
261     //          // SplashKit.FillRectangle(MainColor,leftX, eyeY, 10, 10);
262     //          // SplashKit.FillRectangle(MainColor, rightX, eyeY, 10,10);
```

```
263     //         // SplashKit.FillRectangle(MainColor, leftX, mouthY, 25,10);
264     // SplashKit.FillRectangle(MainColor,leftX + 2,mouthY + 2,21,6);
265    //SplashKit.DrawBitmap(ppr,X, Y);
266             SplashKit.DrawBitmap(klay,X, Y);
267         }
268
269         }
270
271
272
273
274
275
```



27 Concept Visualisation 2

Visualise the programming concepts we've covered so far

Outcome	Weight
Principles	♦♦♦♦◊

Making a concept is an easy to understand fundamental of c#

Outcome	Weight
Justify	♦♦♦♦◊

Making a concept is an easy to understand fundamental of c#

Date	Author	Comment
2019/09/09 16:24	Achmad Kemal	Because I need time
2019/09/09 16:51	Achmad Kemal	because i need more time
2019/09/22 03:11	Achmad Kemal	Ready to Mark
2019/09/22 13:12	Mengmeng Ge	We learnt polymorphism of two types: method overriding and method overloading. Only method overriding exists under inheritance concept. So the expression is not correct.
2019/09/22 13:12	Mengmeng Ge	Fix and Resubmit
2019/09/22 15:54	Achmad Kemal	Ready to Mark
2019/09/22 15:57	Achmad Kemal	Can you give me feedback?
2019/09/22 15:57	Achmad Kemal	thanks
2019/09/22 17:26	Mengmeng Ge	I didn't see the change. For example, "Polymorphism is same like inheritance" is not correct.
2019/09/22 17:26	Mengmeng Ge	Fix and Resubmit
2019/09/22 17:30	Achmad Kemal	Ready to Mark
2019/09/22 18:06	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Concept Visualisation 2

Submitted By:
Achmad Mustafa Kemal
akemal
2019/09/22 17:30

Tutor:
Mengmeng Ge

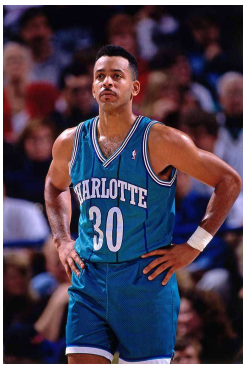
Outcome	**Weight**
Principles	♦♦♦♦◊
Justify	♦♦♦♦◊

Making a concept is an easy to understand fundamental of c#

September 22, 2019

Achmad Mustafa Kemal **ID: 219374683**

Task 8.2



Dell Curry

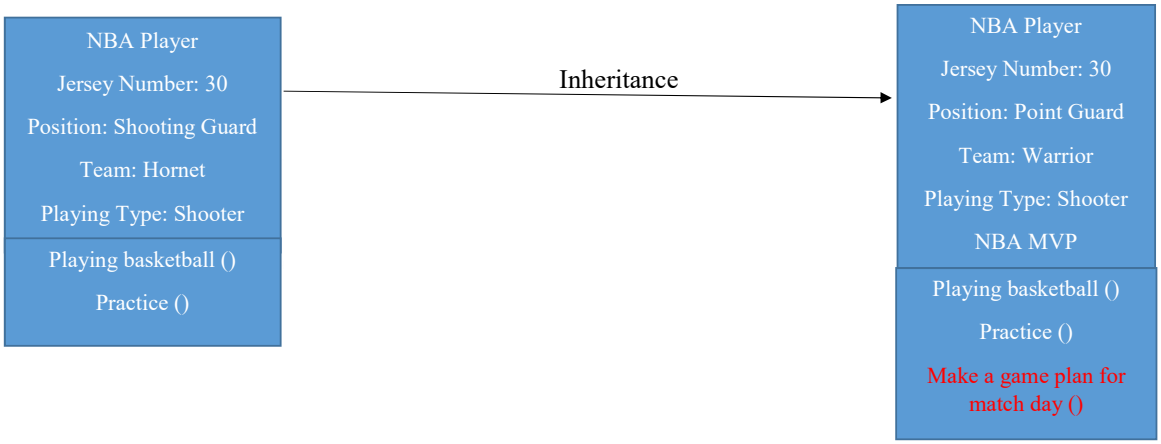
(Father)

Steph Curry

(Son)

Inheritance

(Curry's Family)



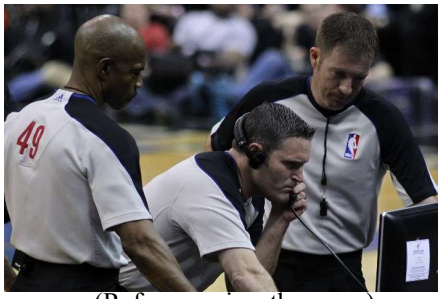
Achmad Mustafa Kemal **ID: 219374683**

Polymorphism

(Class: NBA Referee)

(Referee start game/jump ball)

(Referee make a call for foul)



(Referee review the game)

Achmad Mustafa Kemal **ID: 219374683**

Delegates and Lambda

Kevin Durant
(Basketball Player)

Bob Myers (Basketball Owner)

Steve Kerr
(Coach)

I just want to leave this team
I want to new experience
I hate this coach

Lambda operator

I just want to trade Kevin Durant
I want a new superstar to my team
I want a new stadium for next season

Kevin Durant is unhappy with the team

Steve Kerr want to new player

SIT771

Achmad Mustafa Kemal
ID: 219374683

Task 8.2

In this task, I'm going discuss about inheritance, polymorphism, delegates and lambda in this concept visualisation. First think that I'm going to discuss is inheritance. The definition of inheritance is a class that have dependency with another class. Inheritance is a method that can make class have same characteristics with their base class with inheriting a class. Inheritance can adopt an object that have same entity. A class that have derivative class call as a parent class or base class and derivate class call as a subclass. In the example of inheritance, we have two basketball players, such as, Dell curry and Steph Curry. Dell Curry is father from Steph Curry. So, Dell Curry is a parent class. On the other hand, we can assume that Steph Curry is a subclass because he is a son from Dell Curry. Dell Curry and Steph Curry are both have common information. Such as, both of them are NBA player, they also same wear jersey number 30. They also like shooting ball. But there are some behaviours and information that they both don't match. Such as, Dell Curry is a player from hornet and Steph curry is a player from warrior. Also, Steph curry is a point guard. So, Steph curry's job is handle their teammate in the basketball but Dell Curry is just shooter in the basketball court. In addition, Steph Curry is a NBA MVP and NBA Champion but Dell Curry is never be a NBA MVP and NBA Champion. They also have somewhat similar but not entirely the same. For example, Dell Curry and Steph curry are also playing basketball and practice their skill but only Steph Curry have behaviour that Dell Curry doesn't have like make a game plan for match day. Only Steph Curry can make a game plan for match day but Dell Curry doesn't have it that behaviour.

Polymorphism is known as one name many form. In the Object Oriented Program (OOP), polymorphism can be express as an interface that have many function. Both of them have base class and derived class. With Polymorphism, we can have two or more object of different type to respond with same request. For example, we can assume that referee is a class. With referee class, we can overriding method with using referee. For example, make a class for foul: referee, start game: referee, and review game: referee.

Delegates is a method that can reference a method with using delegates. With delegate, we can reference to method inside a delegate object. Also, a delegate in C# is have similarity concept with function pointer in C or C++. Usually, delegate used for event handlers for Windows GUI parts. In the concept of visualisation, Kevin Durant (Basketball player) doesn't want to his coach because they don't have good chemistry at the moment. So, Kevin Durant passed information to basketball owner (Bob Myers). So, Kevin hope Bob can passed information to Steve Kerr (Kevin Durant's coach) and received to him. Lambda Expression is used for anonymous function. Anonymous function containing expression and statements. Lambda have operator like "=>" that mean "goes to". For example, Kevin Durant and Steve Kerr have many statement that want to pass information to Bob Myers but they want to make to long. So, they make summarize their statements to make short and simple. In this concept visualisation, Kevin said "Kevin Durant is unhappy with the team" and Steve Kerr said "I want to new player". After that, they can passed their expression to Bob Myers (Basketball Owner).

28 Another Language

Pick up another programming language

Outcome	Weight
Principles	♦♦♦♦◊

Python is a good practice

Outcome	Weight
Build Programs	♦♦♦♦◊

Python is a good practice

Outcome	Weight
Design	♦♦♦♦◊

Python is a good practice

Date	Author	Comment
2019/09/21 20:09	Achmad Kemal	Ready to Mark
2019/09/22 12:55	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Another Language

Submitted By:
Achmad Mustafa Kemal
akemal
2019/09/21 20:09

Tutor:
Mengmeng Ge

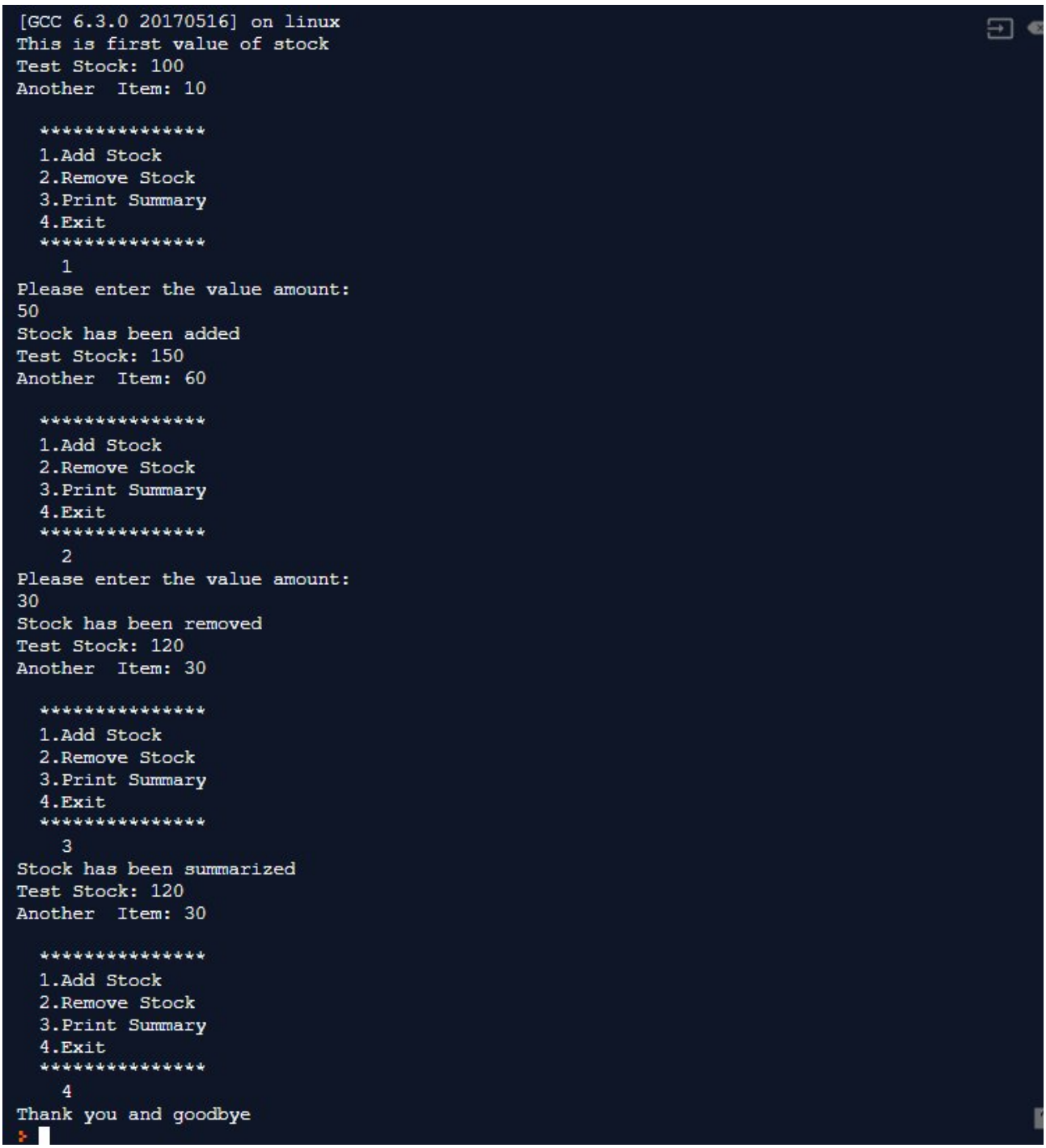
Outcome	**Weight**
Principles	♦♦♦♦◊
Build Programs	♦♦♦♦◊
Design	♦♦♦♦◊

Python is a good practice

September 21, 2019

```
class Stock:

    def __init__(self, name, initial_level):
        self.name = name
        self.quantity_on_hand = initial_level

    def addstock(self, stock_item):
      self.quantity_on_hand += amount
      return self.quantity_on_hand

    def removestock(self, stock_item):
      self.quantity_on_hand -= amount
      return self.quantity_on_hand

    def print_summary(self):
      print("%s: %d"%(self.name,self.quantity_on_hand))


def get_stock(add_item):
    print ("Please enter the %s amount:" %(add_item))
    stock_value = input("")
    stock_item = int(stock_value)
    return stock_item


print("This is first value of stock")
test = Stock("Test Stock", 100)
test.print_summary()

another_item = Stock("Another  Item",10)
another_item.print_summary()

line = 0
while line !=4:
  line = input("""
  ***************
  1.Add Stock
  2.Remove Stock
  3.Print Summary
  4.Exit
  ***************
    """)
  num = int(line)

  if num ==1:
    amount = get_stock("value")
    test.addstock(amount)
    another_item.addstock(amount)
    print("Stock has been added")
    test.print_summary()
    another_item.print_summary()
  elif num == 2:
    amount = get_stock("value")
```

```
54      test.removestock(amount)
55      another_item.removestock(amount)
56      print("Stock has been removed")
57      test.print_summary()
58      another_item.print_summary()
59    elif num == 3:
60      print("Stock has been summarized")
61      test.print_summary()
62      another_item.print_summary()
63    elif num == 4:
64      print("Thank you and goodbye")
65      break
```


[GCC 6.3.0 20170516] on linux
This is first value of stock
Test Stock: 100
Another Item: 10
****
1.Add Stock
2.Remove Stock
3.Print Summary
4.Exit
****
1
Please enter the value amount:
50
Stock has been added
Test Stock: 150
Another Item: 60
****
1.Add Stock
2.Remove Stock
3.Print Summary
4.Exit
****
2
Please enter the value amount:
30
Stock has been removed
Test Stock: 120
Another Item: 30
****
1.Add Stock
2.Remove Stock
3.Print Summary
4.Exit
****
3
Stock has been summarized
Test Stock: 120
Another Item: 30
****
1.Add Stock
2.Remove Stock
3.Print Summary
4.Exit
****
4
Thank you and goodbye

29 Help Others

Demonstrate your excellent technical and communication skills by helping others succeed in understanding the concepts and improving their programming skills.

Outcome	Weight
Justify	♦♦♦♦◊

Help others to better understanding between classmates

Date	Author	Comment
2019/09/23 02:01	Achmad Kemal	Ready to Mark
2019/09/23 07:29	Mengmeng Ge	Complete
2019/09/23 07:29	Mengmeng Ge	Excellent!
2019/09/25 09:22	Mengmeng Ge	Regarding this task, I would suggest to resubmit it next week in case you have more evidence of helping others before October 4.
2019/09/25 09:23	Mengmeng Ge	Fix and Resubmit
2019/10/01 15:21	Achmad Kemal	Ready to Mark
2019/10/01 15:21	Achmad Kemal	I make some changes on this submitted file
2019/10/01 22:08	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Help Others

Submitted By:
Achmad Mustafa Kemal
akemal
2019/10/01 15:21

Tutor:
Mengmeng Ge

Outcome	Weight
Justify	♦♦♦♦◊

Help others to better understanding between classmates

October 1, 2019

SIT771

Achmad Mustafa Kemal

219374683

Task 1.5

In this task, I'm going to write an evidence that help people get stuck with concept, handling with error and understanding the code.

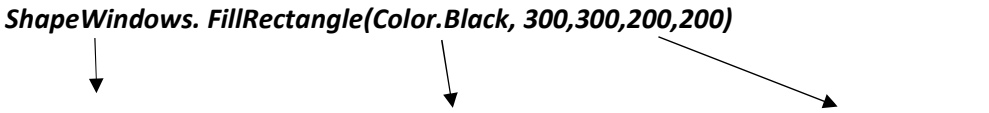
Here is my evidence:

Student Name: Atif Shehzad

Task: Shape Drawing (1.2)

He doesn't know to how to draw a Rectangle in the window. I tell about it.

With using "FillRectangle", he can draw rectangle in the windows. The concept of code will like this:



(Illustration code)

So, I tell him before you use FillRectangle, you must have declared variable that can use for draw FillRectangle. After that, he can create shape and adjust color and size whatever you want.

Task: Another Language (9.1)
He doesn't know about phyton because he never attend the class and he never about the fundamental of phyton. So, I explain to him about and guide him to write a phyton code.

Task: Making a scene (1.4)

He still unfamiliar with C# programming language and he don't know how to install Splash kit and write a C# code in the visual studio. He told me about his struggle. He also doesn't attend the first lecture in this tri semester because he just arrived in Australia. He doesn't know about Splash kit. In this task, we must setup our splash kit manager into our C# files. So, we create the object in the visual studio and run it. After that, I teach him how to setup splash kit manager and write a C# code. After I show how to setup it, he can do setup and work it with the task

Task: Validating Stock Action (3.2)

Atif confuse about the UML class diagram that given in the instruction. He doesn't know what is "+AddStock(quantityAdded: int): void". Also, he doesn't know about how to add method to stock class and add validation. Atif still struggle with C# again because he never heard about C# programming language. I show how to make add stock method and add validation. I tell him that everything "+" plus symbol mean public. So, the code will be like this:

(Illustration code)

public bool AddStock(int quantityAdded)

{

.... add code here

}

I tell him about Boolean value that can used to declare variables to store the Boolean values, such as, true and false. Now, he already know how to read uml class diagram and can write code about addstock method.

Student Name: Muni Swetha Sai Cirimavilla

Task: Warehouse (5.3)

Swetha confuse about method overloading in this task. So, I explain about method overloading. The task said that ExecuteTransaction listed three times in the UML class diagram. So, I said to her about how to method overloading works in the task. Method Overloading is a method that can use same time with using different parameters. The concept of code will like this:

(Illustration code)

public void ExecuteTransaction(StockSaleTransaction StockSaleTransaction)

{

.... add code here

}

public void ExecuteTransaction(StockPurchaseTransaction, StockPurchaseTransaction)

{

.... add code here

}

public void ExecuteTransaction(StockAdjustmentTransaction,StockAdjustmentTransaction)
{

.... add code here

}

After I explain to swetha about method overloading, she decide to try to write a code by herself and now she know and understand about method overloading

She also have problem with code indentation with her code. I show to her about code formatting. In the visual studio, you can use "***shift+alt+f*** to format documentation. So, swetha can understand her code herself.

Task: Abstract Transaction (7.1)

In this task, swetha doesn't use inconsistent variable, such as, sometime she use "toWarehouse" and sometime she use "toWare" in the . This behaviour can make people hard to read the code and hard to fix the error. I suggest her to make consistent variable and simple variable. Now, she use consistent and simple variable for her task. Her problem with this task is solved.

Student Name: Ankit Nakra

Task: Warehouse (5.3)

In this task, he want me to explain about public void AddStock because the task want to us to use Public void for AddStock method and not to use public static void. So, I explain about definition of public void and public static method. I said that public void is a method that contain that will be executed when called and public static void is a method that can access without creating a class. In this task, add stock method only have statement that every user add new stock in the program and the method will passed into the Warehouse's list of stock item. In the end, Ankit is now understand and know the difference between public void and public static void after I discuss and explain to him.

Student Name: Richard William Hester

Task: Name Tester (3.1)

He told me that he can't focus with his study that he doing right now. Even he know about C# programming but he said to hard to get back study again. He can't focus on many thing in the same time. I explain about this task. On this task, we must create simple program that guess name and guess the number. So, the user guess number between 1 and 100 and the program will letting if the guess number are lower or higher. I showed my code to him as reference.

Task: Document Design (6.1)

He doesn't know about visibility of class members. So, I explain to him about what is a visibility of class member, such as, + mean public, - mean private, # mean protected and ~ mean package. Also, I explain about relationship arrow. I said there are many type of relationship, such as, association, inheritance, implementation, dependency, aggregation and composition. So, association is a common relationship between 2 classes. Inheritance is a type of relationship where one linked class is a child class by virtue of assuming the same information and functionalities of parent class. Implementation is a relationship between two classes where one model element execute the behaviour that other model element identifies. Dependency is relationship between two classes that each class have dependency with another

class. Composition is relationship that if class can't stand alone, then this class need to composition relation to another class that which it depend. Aggregation is a relationship between classes that indicate their relationship. After I explain to him about that, he know about visibility of class member and how to draw relationship between classes in the UML class diagram.

Student Name: Marina Liu

Task: Moving the player (3.3)

Marina can't get put player stay on window. She struggle with how to put player stay on window when player move in the window. I also found this one hard to put player stay on window. With using Lecture's feedback, I suggest her that she can use gameWindow .Width in the condition. Also, I suggest her that she can use gameWindow.Height for check if player hit bottom of window. In addition, she need to consider gap of window

So, The concept of code will like this if the player want to stay on window:

(Illustration code)

```
public void StayOnWindow(Window gameWindow)
{
const int GAP = 10;
// X is from public player
// Y is from public player
    if (X < GAP)
    {
    X = GAP;
    }
    if (X > gameWindow.Width - Width - GAP)
    {
    X = gameWindow.Width - Width - GAP;
    }
    if (Y < GAP)
    {
    Y = GAP;
    }
```

```
        if (Y > gameWindow.Height - Height - GAP)
        {
        Y = gameWindow.Height - Height - GAP;
        }
}
```

After I explain about stay on window method for the player, she found the player stay on window. With the code that I showed to her, she can move the player to edge to window and still stay on window.

Task: Messy code (4.2)

In this task, I realized that this task about code indentation and code refactoring. It is because the source code from this task is very messy and hard to find it. Also, I found there are some variable that hard to understand. In the instruction, we must tidy up the code format and fix the issue. I suggest her to make better code formatting with using visual studio code. I suggest type "***Shift+Alt+f"*** to make code looks nicer than before. After, she can fix another problem like change variable to easy read and understand. In the end, she don't confuse to read this task code and she can run the code.

Task: Concept Visualisation 1 (4.4)

She struggle with this task because she don't understand about control flow sequence. So, I told her that control flow sequence is known as basic control flow. I suggest her to use if statement. The reason why I suggest her to use IF statement because this statement is very basic statement that used in the C# programming language to make value is true or false. My explanation is very useful to her to understanding sequence control flow in this task.

Student Name: Melissa Martina Aranha

Task: Moving the player (3.3)

She can't handle the player when the player hit the bottom. So, I told her that with using gamewindow height, player's height and GAP. So, the concept of code will like this:

```
if (Y > gameWindow.Height - Height - GAP) { Y = gameWindow.Height - Height - GAP; }
```

After I explain to her, she solve the problem.

Task: Messy Code (4.2)

She also found same problem with Marina. She can't read the messy code in this task. I always to suggest to do code indentation when they found code that hard to read and understand. I suggest type "***Shift+Alt+f"*** to make code looks nicer than before. After, she can fix another problem like change variable to easy read and understand. In the end, she don't confuse to read this task code and she can run the code. After that, she realize that bullet shot the rocket. I

suggest find human error that type wrong section. Then she found it. I told her that there are some variable that are hard to read and put in the wrong place.

Student Name: Luke Sciberras

Task: Moving the player (3.3)

He also struggle with player stay on window when player move to left and bottom. I also found difficult to make player stay on window. So, I also fix this problem with another my classmate. Marina, Melissa and Luke are face same problem in this task. I remember that lecture give an advice to how to fix the problem. For Luke, he can use gameWindow .Width in the condition. Also, I suggest him that he can use gameWindow.Height for check if player hit bottom of window. In addition, he need to consider gap of window.

The concept of code will be like this

(Ilustration code)

if (X > gameWindow.Width - Width - GAP){X = gameWindow.Width - Width - GAP;}

if (Y > gameWindow.Height - Height - GAP) 119 { 120 Y = gameWindow.Height - Height - GAP; }

In the end, Luke fix the problem after I explain how use gamewindow.Height, gamewindow.Width, player height and width and gap can check the edge of game window.

Task: Concept Visualisation 1 (4.4)

Luke confuse about selection control flow and repetition control flow. For my example, I use switch statement for selection control flow because switch statement is a statement that can handle many case in the program. I also give real example for switch case. I said that switch case usually use for menu option function in the C# programming language. For repetition control flow, I said repetition control flow is like while statement. While statement used to execute statement while Boolean expression value is true. After short explain that I gave to Luke, he now can give example for his control flow and now he also more understand about selection and repetition control flow.

30 Draft Learning Summary Report

Draft learning summary report

Outcome	Weight
Justify	♦♦♦♦♦

Make a summary report is the best way to evaluate what I have done in this subject

Date	Author	Comment
2019/09/24 00:43	Achmad Kemal	Ready to Mark
2019/09/24 08:28	Mengmeng Ge	Reflections look good to me. In portfolio overview, you will need to include the learning outcomes and relate tasks with each learning outcome.
2019/09/24 08:28	Mengmeng Ge	Fix and Resubmit
2019/09/24 13:03	Achmad Kemal	Ready to Mark
2019/09/24 15:31	Mengmeng Ge	It looks good now. Since you have two tasks that haven't been submitted, please update this report with the correct progress graph when submitting your final learning portfolio. There is no need to resubmit this task.
2019/09/24 15:31	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Draft Learning Summary Report

Submitted By:
Achmad Mustafa Kemal
akemal
2019/09/24 13:03

Tutor:
Mengmeng Ge

Outcome	Weight
Justify	♦♦♦♦♦

Make a summary report is the best way to evaluate what I have done in this subject

September 24, 2019

SIT771

Achmad Mustafa Kemal

Learning Summary Report

Achmad Mustafa Kemal
219374683

Self-Assessment Details

The following checklists provide an overview of my self-assessment for this unit.

	Pass (D)	Credit (C)	Distinction (B)	High Distinction (A)
Self-Assessment				✓

Self-Assessment Statement

	Included
Learning Summary Report	✓
Pass tasks complete	✓

Minimum Pass Checklist

	Included
All Credit Tasks are Complete on Doubtfire	✓

Minimum Credit Checklist (in addition to Pass Checklist)

	Included
Distinction tasks (other than Custom Program) are Complete	✓
Custom program meets Distinction criteria	✓

Minimum Distinction Checklist (in addition to Credit Checklist)

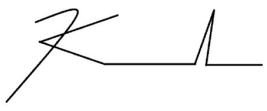
	Included
Something Awesome included	✓
Custom project meets HD requirements	

Minimum High Distinction Checklist (in addition to Distinction Checklist)

Declaration

I declare that this portfolio is my individual work. I have not copied from any other student's work or from any other source except where due acknowledgment is made explicitly in the text, nor has any part of this submission been written for me by another person.

Signature: **Achmad Mustafa Kemal**



Portfolio Overview

This portfolio includes work that demonstrates that I have achieve all Unit Learning Outcomes for SIT771 Unit Title to a **High Distinction** level.

I believe that I should get a High Distinction grade because I finish all task with a High distinction grade target. Also, I believe that I achieved with all unit learning outcomes for SIT771. I can draw concept visualization for C# programming language. I can build a custom program for my project in the SIT771 on track. I can know which principle of object-oriented programming including abstraction, encapsulation, inheritance and polymorphism that I can used to my code program. I can identify which problem that my friend faced in their code. I suggest the concept to make easy understanding. Then, my friend can figure out their problem. For High distinction candidate, I believe that I'm a high distinction candidate because I have a lot of knowledge about C# programming language after I learn in this subject. I can explain about the concept of C# to my classmates.

To become High Distinction candidate, I believe that I cover all aspects including intended learning outcomes. There are evaluate code, principles, build program, design and justify.

In the evaluate code, i must evaluate my code or code hat given in the task. So, I debugging the code to check if the code has some issue. Evaluate code is all about behavior to us for teach us to not go straight to run a code. We must evaluate code before run it. There are tasks that with this learning outcome. Such as, Task 1.3p (How many objects) and Task 4.2c (Messy Code). For Task 1.3p, we must evaluate code that given in the task and then I must evaluate that how many objects inside the code. I must evaluate the code very carefully in this task. For Task 4.2 c, I must use code refactoring and code formatting to evaluate the code in this task because the code is looks messy and we must evaluate and give reflection to this task.

In the principle learning outcome, I must apply and explain the principles of object-oriented programming including abstraction, encapsulation, inheritance and polymorphism. This learning outcome relate to Task 7.1p (Abstract transaction), Task 7.2c (Different Robot), Task 4.4c (Concept Visualization 1) and Task 8.2c (Concept Visualization 2). For task 7.1p, the task must apply abstract principle to execute transaction. In the Task 7.2c, I must abstract and polymorphism to make robot appear different type when the program has been run it. For Task 8.2c and Task 4.4c, I draw own concept visualization including class, object, construction, method, field, property, control flows, inheritance, polymorphism, delegate and lambdas. All of those tasks make me to easier to understanding to know the principle of object-oriented programming.

In the build program section, I think almost task make us build program. Such as, Task 7.3d (Custom Program Code), Task 7.1p (Abstract Transaction), Task 5.3p(Warehouse), Task 5.4c(Many robot), task 9.1c (Another Language) and Task 7.2c (Different Robot). I must implement and test the program.

In the design section, I must design, communicate and evaluate with using diagram. Task 6.1p (Document Design) is make me to design a UML Class diagram for Warehouse. With using UML Class diagram, I can see how the warehouse flow when the code program run it. Another

task is 7.3d (custom program code). Before implement a code in the custom program, I must make design of my custom program and then send to the lecture for approval.

The last learning outcome is aligning justify in this subject. I'm must give relevant proof from the code. I must justify the principle of object-oriented programming with code evidence. The task 8.1p (Code Interview – explanation) is related to justify learning outcome. So, I must explain the key of principle including abstraction, encapsulation, inheritance and polymorphism. Also, the task 7.3d (custom program code) also need to justify about what concept that I'm going to use for my custom program.

In my conclusion, I should get a High Distinction grade because I'm doing better in all tasks. I also give to this subject a best result in this tri semester.

Reflection

The most important things I learnt:

I learnt about how to create a language C# with splash kit. I learn a lot about C# programming language. In this subject, I realized that C# have many concepts to use it in the program. Such as, inheritance, polymorphism, delegates and lambdas. In this subject, I learn how to make code indentation and code refactoring. Code indentation is all about code formatting. I believe that code formatting is also important in the software development. Code Refactoring is a method that fix a problem without changing present method in the code. In the end, I learn what I expected about C# programming language.

The things that helped me most were:

Lecture class is a thing that helped me a lot because we can know the fundamental of C# programming language. Having a discussion with the lecture is helped me a lot because my unit chair gives good explanation and clear feedback. She always gives me respond when I need a help on my ontrack task. She always patience what problem that I have it on my code. She always has best solution to solve the problem. Furthermore, discussing with classmate is another thing that helped me most because we can have different idea and talk about it. Different idea is making me have more knowledge about concept of C#. In addition, cloud Deakin resource is also helped me a lot because when I late a class and the unit chair is busy. The cloud Deakin resource is helping me to fix the problem.

I found the following topics particularly challenging:

The challenging topics are all robot dodge task in this subject because the final result of the task is we can have developed C# game in this subject. There are a lot of time to write a code about this task and I must be carefully write a code in this task because we must sure player have bullet to shot robot, robot have different type, the user also can move player in the game window.

I found the following topics particularly interesting:

Messy code is a topic that I interested to do it because I must fix code that given by the task without changing present method that I already given. This task want me to do code formatting and share the reflection about this task. Concept visualization 1 and Concept visualization 2 are very interesting. So, we must explain about concept of C# programming with our own idea. For example, I draw mind map to discuss about class, object, constructor, method, field, property and control flows in the concept visualization 1 which is I found interesting that I can understanding what is C# concept. I also found better understanding in concept visualization 2.

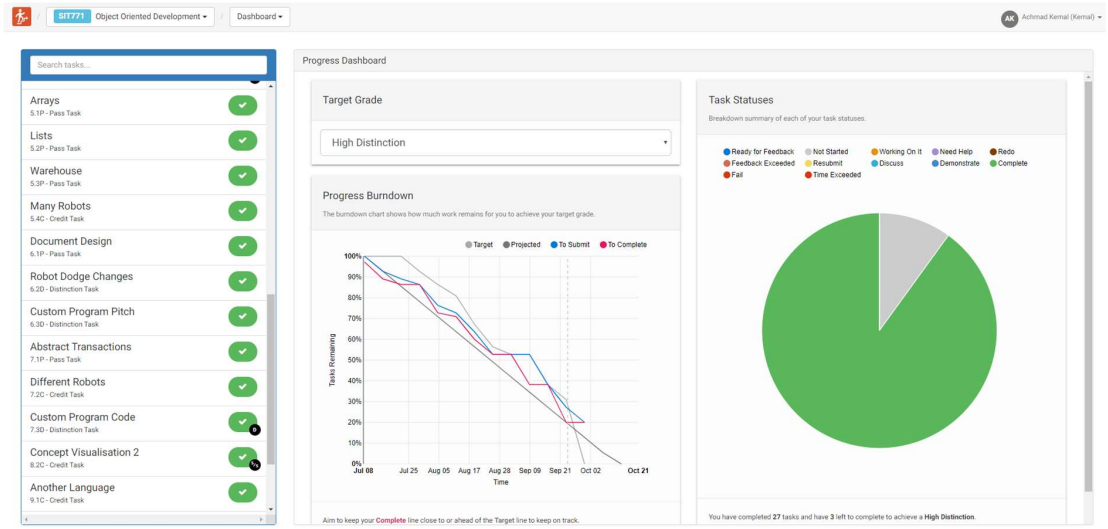
I feel I learnt these topics, concepts, and/or tools really well:

With UML class diagram, I learnt really well about C# programing. I know which one is public and private. UML class diagram is like guide map to develop a C# program code. I can identify about method, class and constructor.

I still need to work on the following areas:

I still need to work about how to have a code formatting because sometime I found that my code is very hard to read and understand. Sometime, I write unnecessary code that make confuses because I think too hard about my program. Choose simple variable name is important thing that I need to work on because I choose long variable that can cause problem.

My progress in this unit was …:



My progress in this unit is almost every pass, credit, distinction and high distinction task. Only few task left in my progress. After I finish all task, I hope I can get HD result.

This unit will help me in the future:

This unit will help me in my career because I found that many IT company use C# programming language to develop project. C# is a most common programming language. I'm very grateful that learn C# programming language.

If I did this unit again I would do the following things differently:

In the future, I would like to manage time about learn C# more in-depth because C# is new experience for me. Because I want C# become my ability of programming skill. So, I need to learn carefully if I did this unit again.

Other…:

My reflection is there are a lot concept that I should learn more in depth for better understanding. Because C# is not only IF logical statement but C# is all about inheritance, polymorphism, delegate, lambda and many more.

31 Custom Program Code

Upload the code for your custom program.

Outcome	Weight
Evaluate Code	♦♦♦◊◊

Custom program code is a pratice that can be useful for learning C#

Outcome	Weight
Principles	♦♦♦◊◊

Custom program code is a pratice that can be useful for learning C#

Outcome	Weight
Build Programs	♦♦♦◊◊

Custom program code is a pratice that can be useful for learning C#

Outcome	Weight
Design	♦♦♦◊◊

Custom program code is a pratice that can be useful for learning C#

Outcome	Weight
Justify	♦♦♦◊◊

Custom program code is a pratice that can be useful for learning C#

Date	Author	Comment
2019/09/23 15:40	Achmad Kemal	Ready to Mark
2019/09/23 16:57	Mengmeng Ge	Complete
2019/09/25 10:52	Mengmeng Ge	Fix and Resubmit
2019/09/25 10:53	Mengmeng Ge	I just sent you an email. Sorry for the wrong info given on Monday. You still need to meet HD standard for this task in order to get HD for this unit as this is the minimum requirement specified in the learning summary report.
2019/10/01 10:29	Achmad Kemal	Ready to Mark
2019/10/01 10:30	Achmad Kemal	Hey, i already edit my custom program
2019/10/01 10:55	Achmad Kemal	- Level 1Survival mode
2019/10/01 10:55	Achmad Kemal	- Final Boss Shoot radioactive logo for reduce final boss's livesFinal boss will shoot the rocketshipRadioactive have shield for defend radioactiveThe shield have a different type of shield
2019/10/01 22:02	Mengmeng Ge	The design of the game is very interesting!
2019/10/01 22:03	Mengmeng Ge	Complete

Deakin University

Object Oriented Development

OnTrack Submission

Custom Program Code

Submitted By:
Achmad Mustafa Kemal
akemal
2019/10/01 10:29

Tutor:
Mengmeng Ge

Outcome	**Weight**
Evaluate Code	♦♦♦◊◊
Principles	♦♦♦◊◊
Build Programs	♦♦♦◊◊
Design	♦♦♦◊◊
Justify	♦♦♦◊◊

Custom program code is a pratice that can be useful for learning C#

October 1, 2019

```
using System;
using SplashKitSDK;

public enum MenuOption
{
    Level1,
    FinalLevel,
    Quit
}

public class Program
{

    private static MenuOption ReadUserOption()
    {
        int option;


        Console.WriteLine("*** Welcome to Spaceship Vs Alien By Achmad Mustafa
        ↪ Kemal***");
        do
        {
            Console.WriteLine("*** 1. Level 1, 2. Final Boss, 3. Quit ***");
            try
            {
                string inputText;
                Console.WriteLine("*** Choose an Option 1-3 ***");
                inputText = Console.ReadLine();
                option = Convert.ToInt32(inputText);
            }
            catch
            {
                option = -1;
            }

        } while (option < 1 || option > 3);
        return (MenuOption)(option - 1);
    }
    //public static Timer myScore = new Timer("My Timer");
    public static Timer myScore = new Timer("My Timer");
    public static Timer myScore1 = new Timer("My Timer");

    private static void Main()
    {
        MenuOption userSelection;

        do
        {
            userSelection = ReadUserOption();
            Console.WriteLine(userSelection);
            switch (userSelection)
            {
                case MenuOption.Level1:
```

```
                    Level1();
                    break;

                case MenuOption.FinalLevel:
                    FinalBoss();
                    break;

            }
        } while (userSelection != MenuOption.Quit);
        // }

        // Window gamewindow;
        // gamewindow = new Window("Spaceship vs Alien", 1000, 900);
        // rocketship rocketship = new rocketship(gamewindow);

        // SoundEffect classicbgm_gamestart = new SoundEffect("gamestart",
        ↪  "Galaxian_Start.wav");
        // classicbgm_gamestart.Play();

        // SplashKit.Delay(500);
        // SoundEffect classicbgm = new SoundEffect("cs", "pacman.mp3");
        // classicbgm.Play();


        // myScore.Start();
        // spaceshipgame spaceshipgame = new spaceshipgame(gamewindow, rocketship);
        //      do
        //      {
        //          SplashKit.ProcessEvents();
        //          spaceshipgame.HandleInput();
        //          gamewindow.Clear(Color.Black);
        //          spaceshipgame.Update(gamewindow);
        //          spaceshipgame.Draw(gamewindow);
        //          gamewindow.Refresh(60);
        //          Console.WriteLine($"{myScore.Ticks} miliseconds have passed");
        //          Console.WriteLine($"Which is {myScore.Ticks / 1000} seconds");
        //      } while (!spaceshipgame.PlayerQuit);
    }
    public static void Level1()
    {
        Window gamewindow;
        gamewindow = new Window("Spaceship vs Alien", 1000, 900);
        rocketship rocketship = new rocketship(gamewindow);
        SoundEffect classicbgm_gamestart = new SoundEffect("gamestart1",
        ↪  "Galaxian_Start.wav");
        classicbgm_gamestart.Play();
        SplashKit.Delay(2000);
        SoundEffect classicbgm = new SoundEffect("cs", "pacman.mp3");
        classicbgm.Play();
        myScore.Start();
        spaceshipgame spaceshipgame = new spaceshipgame(gamewindow, rocketship);
        do
        {
```

```
104             SplashKit.ProcessEvents();
105             spaceshipgame.HandleInput();
106             gamewindow.Clear(Color.Black);
107             spaceshipgame.Update(gamewindow);
108             spaceshipgame.Draw(gamewindow);
109             Console.WriteLine($"{myScore.Ticks} miliseconds have passed");
110             Console.WriteLine($"Which is {myScore.Ticks / 1000} seconds");
111             gamewindow.Refresh(60);
112         } while (!spaceshipgame.PlayerQuit);
113         gamewindow.Close();
114         classicbgm.Stop();
115     }
116 
117     public static void FinalBoss()
118     {
119         Window gamewindow1;
120         gamewindow1 = new Window("Final Boss Level", 1000, 900);
121         rocketship rocketship = new rocketship(gamewindow1);
122         finalbigalien bigalien = new finalbigalien(gamewindow1);
123         SoundEffect classicbgm_finalboss = new SoundEffect("gamestart2",
            ↪ "FinalRound.mp3");
124         classicbgm_finalboss.Play();
125         //gamewindow1.Clear(Color.Black);
126         // gamewindow.DrawText("This is Final Boss Level", Color.Green, 100, 380);
127         // gamewindow.Refresh(60);
128         SplashKit.Delay(4000);
129         SoundEffect classicbgm_finalboss1 = new SoundEffect("final", "15.mp3");
130         classicbgm_finalboss1.Play();
131         myScore1.Start();
132         finalboss fb = new finalboss(gamewindow1, rocketship, bigalien);
133         do
134         {
135             SplashKit.ProcessEvents();
136             fb.HandleInput();
137             gamewindow1.Clear(Color.Black);
138             fb.Update(gamewindow1);
139             fb.Draw(gamewindow1);
140             Console.WriteLine($"{myScore1.Ticks} miliseconds have passed");
141             Console.WriteLine($"Which is {myScore1.Ticks / 1000} seconds");
142             gamewindow1.Refresh(60);
143         } while (!fb.FQuit && !fb.FinalBossQuit);
144         gamewindow1.Close();
145         classicbgm_finalboss1.Stop();
146     }
147 
148 }
149 using System;
150 using SplashKitSDK;
151 
152 
153 
154 public class rocketship
155 {
```

```
156     private Bitmap _rocketshipbitmap;
157
158     private Bitmap heart;
159
160     public int Lives = 5;
161
162     private double _x;
163
164     private double _y;
165
166     private bool _quit = false;
167
168     public bool shot = false;
169
170     public bool shot_fire = false;
171
172     public bool shot_star = false;
173
174     public double X
175     {
176         get { return _x; }
177         set { _x = value; }
178     }
179
180     public double Y
181     {
182         get { return _y; }
183         set { _y = value; }
184     }
185
186     public bool quit
187     {
188          get { return _quit; }
189          private set { _quit = value; }
190     }
191
192     public int Width
193     {
194         get
195         {
196             return _rocketshipbitmap.Width;
197         }
198     }
199
200     public int Height
201     {
202         get
203         {
204             return _rocketshipbitmap.Height;
205         }
206     }
207
208     public Circle CollisionCircle
```

```
209     {
210         get
211         {
212             return SplashKit.CircleAt(X, Y, 20);
213         }
214     }
215 
216     public rocketship(Window gamewindow)
217     {
218         _rocketshipbitmap = new Bitmap("spaceship", "spaceship.png");
219         heart = new Bitmap("heart", "retro_heart1.png");
220         X = (gamewindow.Width - Width) / 2;
221         Y = (gamewindow.Height - Height) / 2;
222     }
223 
224     public void Draw()
225     {
226         _rocketshipbitmap.Draw(X, Y);
227 
228         for (int i = 1; i <= Lives; i++)
229         {
230             SplashKit.DrawBitmap(heart, 975 - (i * 40), 830);
231         }
232 
233     }
234 
235     public void HandleInput(Window gameWindow)
236     {
237         const int SPEED = 5;
238         // Arrow Keys for move function
239         if (SplashKit.KeyDown(KeyCode.RightKey))
240         {
241             X += SPEED;
242         }
243         if (SplashKit.KeyDown(KeyCode.LeftKey))
244         {
245             X -= SPEED;
246         }
247 
248         if (SplashKit.KeyDown(KeyCode.UpKey))
249         {
250             Y -= SPEED;
251         }
252         if (SplashKit.KeyDown(KeyCode.DownKey))
253         {
254             Y += SPEED;
255         }
256         //Key for quit button
257 
258         if (SplashKit.KeyDown(KeyCode.EscapeKey))
259         {
260             quit = true;
261 
```

```
262         }
263         if (SplashKit.KeyTyped(KeyCode.SpaceKey))
264         {
265             shot = true;
266         }
267 
268         if(SplashKit.KeyTyped(KeyCode.AKey))
269         {
270             shot_fire = true;
271         }
272 
273         if(SplashKit.KeyTyped(KeyCode.SKey))
274         {
275             shot_star = true;
276         }
277 
278         if (Lives == 0)
279         {
280             quit = true;
281         }
282     }
283 
284     public void LiveLose()
285     {
286         Lives--;
287 
288 
289     }
290 
291     public void StayOnWindow(Window gameWindow)
292     {
293         const int GAP = 5;
294         if (X < GAP)
295         {
296             X = GAP;
297         }
298         if (X > gameWindow.Width - Width - GAP)
299         {
300             X = gameWindow.Width - Width - GAP;
301         }
302         if (Y < GAP)
303         {
304             Y = GAP;
305         }
306         if (Y > gameWindow.Height - Height - GAP)
307         {
308             Y = gameWindow.Height - Height - GAP;
309         }
310         if (Y < gameWindow.Height - Height - GAP)
311         {
312             Y = gameWindow.Height - Height - GAP;
313         }
314     }
```

```
315
316     public bool CollidedWith(alien other)
317     {
318         bool collide = _rocketshipbitmap.CircleCollision(X, Y,
            ↪  other.CollisionCircle);
319
320         return collide;
321     }
322
323     public bool CollidedWith1(babullet other)
324     {
325         bool collide = _rocketshipbitmap.CircleCollision(X, Y,
            ↪  other.CollisionCircle);
326
327         return collide;
328     }
329 }
330
331 using System;
332 using SplashKitSDK;
333 public abstract class shield
334 {
335
336     public double X {get; set;}
337     public double Y {get; set;}
338     public Vector2D Velocity {get; set;}
339
340
341     public int Width
342     {
343         get { return 50;}
344     }
345
346     public int Height
347     {
348         get { return 50;}
349     }
350
351     public Circle CollisionCircle
352     {
353
354         get
355         {
356             return SplashKit.CircleAt(X + Width/2, Y + Height/2 ,20);
357         }
358     }
359
360     public shield(Window gameWindow, rocketship player){}
361     public void Update()
362     {
363
364         X += Velocity.X;
365         Y += Velocity.Y;
```

```
366
367     }
368     public abstract void Draw();
369     public bool IsOffscreen(Window gameWindow)
370     {
371
372         if(X < -Width || X > gameWindow.Width || Y < -Height || Y>
            ↪  gameWindow.Height) return true;
373
374         return false;
375     }
376 }
377
378 using System;
379 using SplashKitSDK;
380 using System.Collections.Generic;
381
382 public class spaceshipgame
383 {
384     //Private Field
385     public rocketship _rocketship;
386     private Window _gamewindow;
387     public static Timer myScore = new Timer("My Timer");
388     private static List<alien> _aliens = new List<alien>();
389     private List<bullet> _Bullets = new List<bullet>();
390
391     //Property
392     public bool PlayerQuit
393     {
394          get { return _rocketship.quit; }
395     }
396
397     public spaceshipgame(Window gamewindow, rocketship rocketship)
398     {
399         _gamewindow = gamewindow;
400         _rocketship = rocketship;
401     }
402
403     public void HandleInput()
404     {
405         _rocketship.HandleInput(_gamewindow);
406         _rocketship.StayOnWindow(_gamewindow);
407
408         //this if statement make bullet can come out when the user click left
            ↪  button on the mouse
409         if (_rocketship.shot == true)
410         {
411             _rocketship.shot = false;
412             SoundEffect laser = new SoundEffect("laser", "Galaga_Fire.wav");
413             laser.Play();
414             _Bullets.Add(LaserBullet());
415
416         }
```

```
417
418         //this if statement make bullet can come out when the user click left
            ↪  button on the mouse
419         if (_rocketship.shot_fire == true)
420         {
421             _rocketship.shot_fire = false;
422             SoundEffect fire_bullet = new SoundEffect("fire", "Galaga_Fire.wav");
423             fire_bullet.Play();
424             _Bullets.Add(TriangleBullet());
425
426         }
427
428         //this if statement make bullet can come out when the user click left
            ↪  button on the mouse
429         if (_rocketship.shot_star == true)
430         {
431             _rocketship.shot_star = false;
432             SoundEffect star_bullet = new SoundEffect("fire", "Galaga_Fire.wav");
433             star_bullet.Play();
434             _Bullets.Add(StarBullet());
435
436         }
437     }
438
439     public void Draw(Window gamewindow)
440     {
441         _rocketship.Draw();
442
443         foreach (bullet bullets in _Bullets)
444         {
445             bullets.Draw();
446         }
447
448         foreach (alien aliens in _aliens)
449         {
450             aliens.Draw();
451         }
452         SplashKit.DrawTextOnWindow(gamewindow, Convert.ToString(myScore.Ticks /
            ↪  1000), Color.White, 50, 50);
453     }
454
455     public void Update(Window gameWindow)
456     {
457         foreach (alien alien in _aliens)
458         {
459             alien.Update();
460         }
461         if (SplashKit.Rnd() < 0.05)
462         {
463             _aliens.Add(RandomAlien());
464         }
465         foreach (bullet bullets in _Bullets)
466         {
```

```
            bullets.Update();
        }
        CheckCollisions();
    }

    //Private Method CheckCollisions
    private void CheckCollisions()
    {
        List<alien> _deleteAliens = new List<alien>();
        List<bullet> _deleteBullets = new List<bullet>();


        foreach (alien aliens in _aliens)
        {
            if (_rocketship.CollidedWith(aliens))
            {
                _deleteAliens.Add(aliens);
                _rocketship.LiveLose();
            }
            if (aliens.isOffscreen(_gamewindow))
            {
                _deleteAliens.Add(aliens);
            }
            foreach (bullet bullets in _Bullets)
            {
                if (bullets.CollidedWith(aliens) || aliens.isOffscreen(_gamewindow))
                {
                    _deleteAliens.Add(aliens);
                    _deleteBullets.Add(bullets);
                }

            }

        }

        foreach (alien aliens in _deleteAliens)
        {
            _aliens.Remove(aliens);
        }
        foreach (bullet bullets in _deleteBullets)
        {
            _Bullets.Remove(bullets);
        }
    }
    public alien RandomAlien()
    {
        //return new alien(_gamewindow, _rocketship);
        if (SplashKit.Rnd() < 0.5)
        {
            return new alien1(_gamewindow, _rocketship);
        }
        else if (SplashKit.Rnd() < 0.5)
        {
```

```
            return new alien2(_gamewindow, _rocketship);
        }
        else
        {
            return new alien3(_gamewindow, _rocketship);
        }
    }
    public bullet TriangleBullet()
    {
        return new Triangle(_gamewindow, _rocketship);
    }

    public bullet LaserBullet()
    {
        return new Laser(_gamewindow, _rocketship);

    }

    public bullet StarBullet()
    {
        return new Star(_gamewindow, _rocketship);

    }
}
using System;
using SplashKitSDK;

public abstract class alien
{
    public double X
    { get; set; }
    public double Y
    { get; set; }
    private Bitmap bitmap
    { get; set; }
    private Vector2D Velocity
    { get; set; }

    public Circle CollisionCircle
    {
        get
        {
            return SplashKit.CircleAt(X, Y, 20);
        }
    }

    public int Width

    {
        get
        { return 50; }
    }
    public int Height
```

```
573
574     {
575         get
576         { return 50; }
577     }
578
579     public alien(Window gamewindow, rocketship rocketship)
580     {
581         X = 0;
582         Y = 0;
583         const int SPEED = 3;
584
585         if (SplashKit.Rnd() < 0.5)
586         {
587
588             X = SplashKit.Rnd(gamewindow.Width);
589
590             Y = -Height;
591         }
592         else
593         {
594             X = -Width;
595             Y = -Height;
596         }
597
598         // if (SplashKit.Rnd() < 0.5)
599
600         //     X = -Width;
601         // else
602         //     X = gamewindow.Width;
603         //Get a Point for the Robot
604         Point2D fromPt = new Point2D()
605         {
606             X = X,
607             Y = Y
608         };
609
610         //Get a Point for the player
611         Point2D toPt = new Point2D()
612         {
613             X = rocketship.X,
614             Y = rocketship.Y
615         };
616
617         //Calculate the direction to head
618         Vector2D dir;
619         dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
620
621         //Set speed and assign to the Velocity
622         Velocity = SplashKit.VectorMultiply(dir, SPEED);
623     }
624
625     // public void Draw()
```

```
626     // {
627     //     Bitmap rocketship = new Bitmap("alien", "alien.png");
628     //     SplashKit.DrawBitmap(rocketship, X, Y);
629     // }
630     public abstract void Draw();
631     public void Update()
632     {
633
634         X += Velocity.X;
635         Y += Velocity.Y;
636     }
637
638     public bool isOffscreen(Window screen)
639     {
640         if (X < -Width || X > screen.Width || Y < -Height || Y > screen.Height)
641         {
642             return true;
643         }
644         else
645         {
646             return false;
647         }
648     }
649 }
650
651 public class alien1 : alien
652 {
653     public alien1(Window gamewindow, rocketship rocketship) : base(gamewindow,
      ↪  rocketship) { }
654
655     public override void Draw()
656     {
657        Bitmap alien1 = new Bitmap("alien1", "alien1.png");
658        SplashKit.DrawBitmap(alien1, X, Y);
659     }
660 }
661
662 public class alien2 : alien
663 {
664     public alien2(Window gamewindow, rocketship rocketship) : base(gamewindow,
      ↪  rocketship) { }
665
666     public override void Draw()
667     {
668        Bitmap alien2 = new Bitmap("alien2", "alien2.png");
669        SplashKit.DrawBitmap(alien2, X, Y);
670     }
671 }
672
673 public class alien3 : alien
674 {
675     public alien3(Window gamewindow, rocketship rocketship) : base(gamewindow,
      ↪  rocketship) { }
```

```
676
677     public override void Draw()
678     {
679         Bitmap alien3 = new Bitmap("alien3", "alien3.png");
680         SplashKit.DrawBitmap(alien3, X, Y);
681     }
682 }
683
684 using SplashKitSDK;
685
686 public class babullet
687 {
688     public double X
689     { get; set; }
690     public double Y
691     { get; set; }
692     private Vector2D Velocity
693     { get; set;}
694     private Bitmap _BulletBitmapBigAlien;
695
696     public Circle CollisionCircle
697     {
698         get
699         {
700             return SplashKit.CircleAt(X, Y, 20);
701         }
702     }
703
704     public babullet(Window gamewindow, rocketship rocketship, finalbigalien
      ↪ finalbigalien)
705     {
706         _BulletBitmapBigAlien = SplashKit.LoadBitmap("bulletbigalien", "bb1.png");
707         X = finalbigalien.X+ finalbigalien.Width/2;
708         Y = finalbigalien.Height + finalbigalien.Y;
709         const int SPEED = 5;
710
711         Point2D fromPt = new Point2D()
712         {
713             X = X,
714             Y = Y,
715         };
716
717         Point2D toPt = new Point2D()
718         {
719             X = rocketship.X,
720             Y = rocketship.Y
721         };
722         // Calculate the direction to head.
723         Vector2D dir;
724         dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
725         // Set the speed and assign to the Velocity
726         Velocity = SplashKit.VectorMultiply(dir, SPEED);
727     }
```

```
728     public void Update()
729     {
730         X += Velocity.X;
731         Y += Velocity.Y;
732     }
733     public void Draw()
734     {
735         SplashKit.FillCircle(Color.Green, X, Y, 10);
736         //SplashKit.DrawBitmap(_BulletBitmapBigAlien, X, Y);
737     }
738     public bool CollidedWith (rocketship other)
739     {
740         return SplashKit.CirclesIntersect(CollisionCircle,other.CollisionCircle);
741     }
742     public bool IsOffScreen (Window gamewindow)
743     {
744         if (X < -10 || X > gamewindow.Width || Y < -10 || Y > gamewindow.Height)
            ↪  return true;
745         return false;
746     }
747 }
748 using SplashKitSDK;
749 public class brickwall : shield
750 {
751     public brickwall(Window gameWindow, rocketship rocketship) : base ( gameWindow,
        ↪  rocketship)
752     {
753     // public double X { get; set; }
754     // public double Y { get; set; }
755     // public Color MainColor { get; set; }
756     // public Vector2D Velocity { get; set; }
757
758
759     // public int Width
760     // {
761     //     get { return 50; }
762     // }
763
764     // public int Height
765     // {
766     //     get { return 50; }
767     // }
768
769     // public Circle CollisionCircle
770     // {
771
772     //     get
773     //     {
774     //         return SplashKit.CircleAt(X + Width / 2, Y + Height / 2, 20);
775     //     }
776     // }
777
778     // public brickwall(Window gamewindow, rocketship rocketship)
```

```
779     // {
780         // X = (gamewindow.Width - Width)/2;
781         // Y = (gamewindow.Height - Height)/2;
782         X = -10;
783         Y = gameWindow.Height - 400;
784         const int SPEED = 3;
785 
786 
787         Point2D fromPt = new Point2D()
788         {
789             X = X,
790             Y = Y
791         };
792 
793         //Get a Point for the Player
794         Point2D toPt = new Point2D()
795         {
796             X = 0,
797             Y = Y
798         };
799 
800         //Calculate the direction to head.
801         Vector2D dir;
802         dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
803 
804         //Set the speed and assign to the Velocity
805         Velocity = SplashKit.VectorMultiply(dir, SPEED);
806     }
807 
808     public override void Draw()
809     {
810         Bitmap brick = new Bitmap("brick", "Brickwall1.png");
811         SplashKit.DrawBitmap(brick,X, Y);
812     }
813 
814     // public void Update()
815     // {
816 
817     //     X += Velocity.X;
818     //     Y += Velocity.Y;
819 
820     // }
821     // public bool IsOffscreen(Window screen)
822     // {
823 
824     //     if(X < -Width || X > screen.Width || Y < -Height || Y> screen.Height)
        ↪  return true;
825 
826     //     return false;
827     // }
828 
829 }
830 using System;
```

```
using SplashKitSDK;


public abstract class bullet
{
    private Bitmap _BulletBitmap;
    // private Bitmap _BulletBitmap1;
    public double X
    { get; set; }
    public double Y
    { get; set; }
    //private double _angle;
    private Vector2D Velocity
    { get; set; }

    //private bool _active  = false;
    public Circle CollisionCircle
    {
        get
        {
            return SplashKit.CircleAt(X, Y, 20);
        }
    }
    public bullet(Window gamewindow, rocketship rocketship)
    {

        _BulletBitmap = SplashKit.LoadBitmap("bullet", "bullet.png");
        // _BulletBitmap1 = SplashKit.LoadBitmap("fire_bullet", "Fire.png");
        const int SPEED = 5;


        //This is X and Y to make bullet come out to player
        X = rocketship.X;
        Y = rocketship.Y - rocketship.Height;



        Point2D fromPt = new Point2D()
        {
            X = X,
            Y = Y
        };

        //Get a Point for bullet
        //  Point2D toPt = SplashKit.MousePosition();
        Point2D toPt = new Point2D()
        {
            X = X,
            Y = -rocketship.Y
        };


        //Calculate the direction to head
```

```
884         Vector2D dir;
885         dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
886 
887         //Set speed and assign to the Velocity
888         Velocity = SplashKit.VectorMultiply(dir, SPEED);
889     }
890 
891     public bool CollidedWith(alien other)
892     {
893         return _BulletBitmap.CircleCollision(X, Y, other.CollisionCircle);
894     }
895 
896     public bool CollidedWith1(shield other)
897     {
898         return _BulletBitmap.CircleCollision(X, Y, other.CollisionCircle);
899     }
900 
901     public bool CollidedWith3(radioactive other)
902     {
903             return _BulletBitmap.CircleCollision(X, Y, other.CollisionCircle);
904     }
905     // public void Draw()
906     // {
907     //     SplashKit.DrawBitmap(_BulletBitmap, X,Y);
908     // }
909 
910     public abstract void Draw();
911 
912     public void Update()
913     {
914 
915         X += Velocity.X;
916         Y += Velocity.Y;
917 
918     }
919     public bool isOffScreen(Window gamewindow)
920     {
921         if (X < -50 || X > gamewindow.Width || Y < -50 || Y > gamewindow.Height)
            ↪  return true;
922         return false;
923     }
924 
925 }
926 
927 public class Laser : bullet
928 {
929     public Laser(Window gamewindow, rocketship rocketship) : base(gamewindow,
        ↪  rocketship) { }
930 
931     public override void Draw()
932     {
933         Bitmap _BulletBitmap = new Bitmap("bullet", "bullet.png");
934         SplashKit.DrawBitmap(_BulletBitmap, X, Y);
```

```
935     }
936 }
937
938 public class Triangle : bullet
939 {
940     public Triangle(Window gamewindow, rocketship rocketship) : base(gamewindow,
        ↪ rocketship) { }
941
942     public override void Draw()
943     {
944         Bitmap _BulletBitmap1 = new Bitmap("triangle", "triangle.png");
945         SplashKit.DrawBitmap(_BulletBitmap1, X, Y);
946     }
947 }
948
949 public class Star : bullet
950 {
951     public Star(Window gamewindow, rocketship rocketship) : base(gamewindow,
        ↪ rocketship) { }
952
953     public override void Draw()
954     {
955         Bitmap _BulletBitmap1 = new Bitmap("star", "star.png");
956         SplashKit.DrawBitmap(_BulletBitmap1, X, Y);
957     }
958 }
959 using System;
960 using SplashKitSDK;
961 public class castle : shield
962 {
963     public castle(Window gameWindow, rocketship rocketship) : base ( gameWindow,
        ↪ rocketship)
964     {
965     // public double X { get; set; }
966     // public double Y { get; set; }
967     // public Color MainColor { get; set; }
968     // public Vector2D Velocity { get; set; }
969
970
971     // public int Width
972     // {
973     //     get { return 50; }
974     // }
975
976     // public int Height
977     // {
978     //     get { return 50; }
979     // }
980
981     // public Circle CollisionCircle
982     // {
983
984     //     get
```

```
985     //     {
986     //         return SplashKit.CircleAt(X + Width / 2, Y + Height / 2, 20);
987     //     }
988     // }
989
990     // public brickwall(Window gamewindow, rocketship rocketship)
991     // {
992         // X = (gamewindow.Width - Width)/2;
993         // Y = (gamewindow.Height - Height)/2;
994         X = -10;
995         Y = gameWindow.Height - 300;
996         const int SPEED = 3;
997
998
999         Point2D fromPt = new Point2D()
1000        {
1001            X = X,
1002            Y = Y
1003        };
1004
1005        //Get a Point for the Player
1006        Point2D toPt = new Point2D()
1007        {
1008            X = 0,
1009            Y = Y
1010        };
1011
1012        //Calculate the direction to head.
1013        Vector2D dir;
1014        dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));
1015
1016        //Set the speed and assign to the Velocity
1017        Velocity = SplashKit.VectorMultiply(dir, SPEED);
1018    }
1019
1020    public override void Draw()
1021    {
1022        Bitmap castle = new Bitmap("castle", "castle1.png");
1023        SplashKit.DrawBitmap(castle,X, Y);
1024    }
1025
1026    // public void Update()
1027    // {
1028
1029    //     X += Velocity.X;
1030    //     Y += Velocity.Y;
1031
1032    // }
1033    // public bool IsOffscreen(Window screen)
1034    // {
1035
1036    //     if(X < -Width || X > screen.Width || Y < -Height || Y> screen.Height)
        ↪  return true;
```

```
1037
1038        //      return false;
1039        // }
1040
1041    }
1042    using System;
1043    using SplashKitSDK;
1044
1045
1046
1047    public class finalbigalien
1048    {
1049        private Bitmap _fabitmap;
1050
1051        private Bitmap cheart;
1052
1053        public int Lives = 5;
1054
1055        private double _x;
1056
1057        private double _y;
1058
1059        private bool _quit = false;
1060
1061        // public bool shot = false;
1062
1063        //public bool shot_fire = false;
1064
1065        // public bool shot_star = false;
1066
1067        public double X
1068        {
1069            get { return _x; }
1070            set { _x = value; }
1071        }
1072
1073        public double Y
1074        {
1075            get { return _y; }
1076            set { _y = value; }
1077        }
1078
1079        public bool quit
1080        {
1081             get { return _quit; }
1082             private set { _quit = value; }
1083        }
1084
1085        public int Width
1086        {
1087            get
1088            {
1089                return _fabitmap.Width;
```

```
        }
    }

    public int Height
    {
        get
        {
            return _fabitmap.Height;
        }
    }


    public Circle CollisionCircle
    {

        get
        {
            return SplashKit.CircleAt(X + Width/2, Y + Height/2 ,20);
        }
    }

    public finalbigalien(Window gamewindow)
    {
        _fabitmap = new Bitmap("fb", "bigboss1.png");
        cheart = new Bitmap("cheart", "chemical.png");
        X = (gamewindow.Width - Width) / 2;
        Y = (gamewindow.Height - Height) / 50;
    }

    public void Draw()
    {
        _fabitmap.Draw(X, Y);

        for (int i = 1; i <= Lives; i++)
        {
            SplashKit.DrawBitmap(cheart, 975 - (i * 60), 10);
        }

    }

    public void HandleInput(Window gameWindow)
    {
        // const int SPEED = 5;
        // // Arrow Keys for move function
        // if (SplashKit.KeyDown(KeyCode.RightKey))
        // {
        //     X += SPEED;
        // }
        // if (SplashKit.KeyDown(KeyCode.LeftKey))
        // {
        //     X -= SPEED;
        // }

```

```
        // if (SplashKit.KeyDown(KeyCode.UpKey))
        // {
        //     Y -= SPEED;
        // }
        // if (SplashKit.KeyDown(KeyCode.DownKey))
        // {
        //     Y += SPEED;
        // }
        //Key for quit button

        if (SplashKit.KeyDown(KeyCode.EscapeKey))
        {
            quit = true;

        }
        // if (SplashKit.KeyTyped(KeyCode.SpaceKey))
        // {
        //     shot = true;
        // }

        // if(SplashKit.KeyTyped(KeyCode.KKey))
        // {
        //      shot_fire = true;
        // }

        // if(SplashKit.KeyTyped(KeyCode.SKey))
        // {
        //     shot_star = true;
        // }

        if (Lives == 0)
        {
         quit = true;
        }
    }

    public void LiveLose()
    {
        Lives--;
        // if (Lives == 0)
        // {
        //     quit = true;
        // }

    }


    // public void StayOnWindow(Window gameWindow)
    // {
    //     const int GAP = 5;
    //     if (X < GAP)
    //     {
    //         X = GAP;
```

```
1196        //      }
1197        //      if (X > gameWindow.Width - Width - GAP)
1198        //      {
1199        //          X = gameWindow.Width - Width - GAP;
1200        //      }
1201        //      if (Y < GAP)
1202        //      {
1203        //          Y = GAP;
1204        //      }
1205        //      if (Y > gameWindow.Height - Height - GAP)
1206        //      {
1207        //          Y = gameWindow.Height - Height - GAP;
1208        //      }
1209        //      if (Y < gameWindow.Height - Height - GAP)
1210        //      {
1211        //          Y = gameWindow.Height - Height - GAP;
1212        //      }
1213        // }
1214
1215          public bool CollidedWith(bullet other)
1216          {
1217              bool collide = _fabitmap.CircleCollision(X, Y, other.CollisionCircle);
1218
1219              return collide;
1220          }
1221      }
1222
1223      using System;
1224      using SplashKitSDK;
1225      using System.Collections.Generic;
1226
1227
1228      public class finalboss
1229      {
1230          public rocketship _rocketship;
1231          private Window _gamewindow;
1232          public finalbigalien _bigalien;
1233
1234          private List<shield> _shield = new List<shield>();
1235
1236          private List<bullet> _Bullets = new List<bullet>();
1237
1238          private List<radioactive> _rda = new List<radioactive>();
1239
1240          private List<babullet> _Bulletsba = new List<babullet>();
1241
1242
1243          public bool FQuit
1244          {
1245              get { return _rocketship.quit; }
1246          }
1247
1248          public bool FinalBossQuit
```

```
1249        {
1250            get { return _bigalien.quit; }
1251        }
1252        public finalboss(Window gamewindow, rocketship rocketship, finalbigalien
            ↪ bigalien)
1253        {
1254            _gamewindow = gamewindow;
1255            _rocketship = rocketship;
1256            _bigalien = bigalien;
1257        }
1258
1259        public void HandleInput()
1260        {
1261            _rocketship.HandleInput(_gamewindow);
1262            _rocketship.StayOnWindow(_gamewindow);
1263            _bigalien.HandleInput(_gamewindow);
1264
1265            if (_rocketship.shot == true)
1266            {
1267                _rocketship.shot = false;
1268                SoundEffect laser = new SoundEffect("laser", "Galaga_Fire.wav");
1269                laser.Play();
1270                _Bullets.Add(LaserBullet());
1271            }
1272            if (_rocketship.shot_fire == true)
1273            {
1274                _rocketship.shot_fire = false;
1275                SoundEffect fire_bullet = new SoundEffect("fire", "Galaga_Fire.wav");
1276                fire_bullet.Play();
1277                _Bullets.Add(TriangleBullet());
1278
1279            }
1280            if (_rocketship.shot_star == true)
1281            {
1282                _rocketship.shot_star = false;
1283                SoundEffect star_bullet = new SoundEffect("fire", "Galaga_Fire.wav");
1284                star_bullet.Play();
1285                _Bullets.Add(StarBullet());
1286            }
1287        }
1288
1289        public void Draw(Window gamewindow)
1290        {
1291            _bigalien.Draw();
1292
1293            _rocketship.Draw();
1294
1295            foreach (shield shields in _shield)
1296            {
1297                shields.Draw();
1298            }
1299            foreach (bullet bullets in _Bullets)
1300            {
```

```
            bullets.Draw();
        }

        foreach (babullet bulletsba in _Bulletsba)
        {
            bulletsba.Draw();
        }

         foreach (radioactive ra in _rda)
        {
            ra.Draw();
        }
    }

    public void Update(Window gameWindow)
    {
        foreach (shield shield in _shield)
        {
            shield.Update();
        }
        if (SplashKit.Rnd() < 0.04)
        {
            _shield.Add(RandomShield());
        }
        foreach (bullet bullets in _Bullets)
        {
            bullets.Update();
        }
        foreach (babullet bulletba in _Bulletsba)
        {
            bulletba.Update();
        }
        if (SplashKit.Rnd() < 0.01)
        {
            _Bulletsba.Add(NewBulletBA());
        }
        foreach (radioactive rda in _rda)
        {
            rda.Update();
        }
        if (SplashKit.Rnd() < 0.01)
        {
            _rda.Add(NewRA());
        }

        CheckCollisions();
    }

    private void CheckCollisions()
    {
        List<radioactive> _deleteRA = new List<radioactive>();
        List<babullet> _deleteBulletsBA = new List<babullet>();
        List<bullet> _deleteBullets = new List<bullet>();
```

```
        List<shield> _deleteShields = new List<shield>();



        foreach (babullet babullet in _Bulletsba)
        {
            if (_rocketship.CollidedWith1(babullet))
            {
                _deleteBulletsBA.Add(babullet);
                _rocketship.LiveLose();
            }
            if (babullet.IsOffScreen(_gamewindow))
            {
                _deleteBulletsBA.Add(babullet);
            }
        }
        foreach (radioactive rda in _rda)
        {
            if (rda.isOffscreen(_gamewindow))
            {
                _deleteRA.Add(rda);
            }
            foreach (bullet bullets in _Bullets)
            {
                if (bullets.CollidedWith3(rda))
                {
                    _deleteRA.Add(rda);
                    _deleteBullets.Add(bullets);
                    _bigalien.LiveLose();
                }
            }
        }
        foreach (shield shield in _shield)
        {
            if (shield.IsOffscreen(_gamewindow))
            {
                _deleteShields.Add(shield);

            }
            foreach (bullet bullets in _Bullets)
            {
                if (bullets.CollidedWith1(shield))
                {
                    _deleteShields.Add(shield);
                    _deleteBullets.Add(bullets);
                }
            }
        }

        foreach (radioactive ra in _deleteRA)
        {
            _rda.Remove(ra);
        }
```

```
1407            foreach (babullet ba in _deleteBulletsBA)
1408            {
1409                _Bulletsba.Remove(ba);
1410            }
1411            foreach (shield shield in _deleteShields)
1412            {
1413                _shield.Remove(shield);
1414            }
1415            foreach (bullet bullets in _deleteBullets)
1416            {
1417                _Bullets.Remove(bullets);
1418            }
1419        }
1420
1421        public babullet NewBulletBA()
1422        {
1423            babullet newBullet = new babullet(_gamewindow, _rocketship, _bigalien);
1424            return newBullet;
1425        }
1426        public radioactive NewRA()
1427        {
1428            radioactive NewRA = new radioactive(_gamewindow, _rocketship);
1429            return NewRA;
1430        }
1431        public bullet TriangleBullet()
1432        {
1433            return new Triangle(_gamewindow, _rocketship);
1434        }
1435
1436        public bullet LaserBullet()
1437        {
1438            return new Laser(_gamewindow, _rocketship);
1439
1440        }
1441        public bullet StarBullet()
1442        {
1443            return new Star(_gamewindow, _rocketship);
1444
1445        }
1446        public shield RandomShield()
1447        {
1448            if (SplashKit.Rnd() < 0.4)
1449            {
1450                return new brickwall(_gamewindow, _rocketship);
1451            }
1452            else
1453            {
1454                return new castle(_gamewindow, _rocketship);
1455            }
1456        }
1457    }
1458
1459    using System;
```

```
using System.Collections.Generic;
using SplashKitSDK;

public class radioactive
{  //Properties
    private double X
    { get; set; }
    private double Y
    { get; set; }
    private Bitmap bitmap
    { get; set; }
    private Vector2D Velocity
    { get; set; }
    public Color MainColor;

    public Circle CollisionCircle
    {
        get
        {
            return SplashKit.CircleAt(X, Y, 20);
        }
    }

    public int Width

    {
        get
        { return 50; }
    }
    public int Height

    {
        get
        { return 50; }
    }


    public radioactive(Window gameWindow, rocketship rocketship)
    {
        //Starting Point
        X = -10;
        Y = gameWindow.Height - 500;
        const int SPEED = 3;

        Point2D fromPt = new Point2D()
        {
            X = X,
            Y = Y
        };

        //Get a Point for the Player
        Point2D toPt = new Point2D()
        {
```

```
            X = 0,
            Y = Y
        };

        //Calculate the direction to head
        Vector2D dir;
        dir = SplashKit.UnitVector(SplashKit.VectorPointToPoint(fromPt, toPt));

        //Set speed and assign to the Velocity
        Velocity = SplashKit.VectorMultiply(dir, SPEED);
    }

    //Create a robot
    public void Draw()
    {
        Bitmap radioactive = new Bitmap("radioactive", "radioactive1.png");
        SplashKit.DrawBitmap(radioactive, X, Y);
    }

    //Update Method
    public void Update()
    {

        X += Velocity.X;
        Y += Velocity.Y;
    }

    // The robot start from offscreen
    public bool isOffscreen(Window screen)
    {
        if (X < -Width || X > screen.Width || Y < -Height || Y > screen.Height)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}
```

Video Link:

https://youtu.be/Gq19iMoVB-s

Game Description

- Level 1

Survival mode

- Final Boss

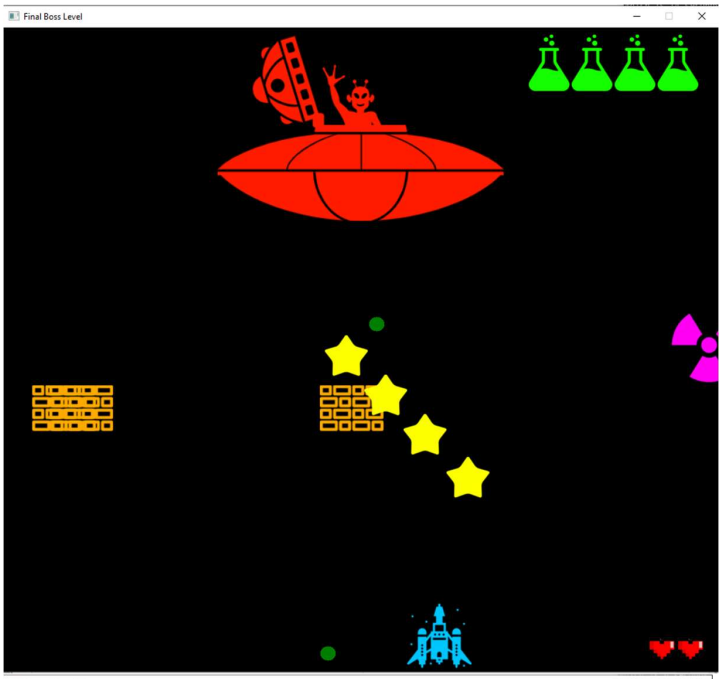
(This radioactive logo)

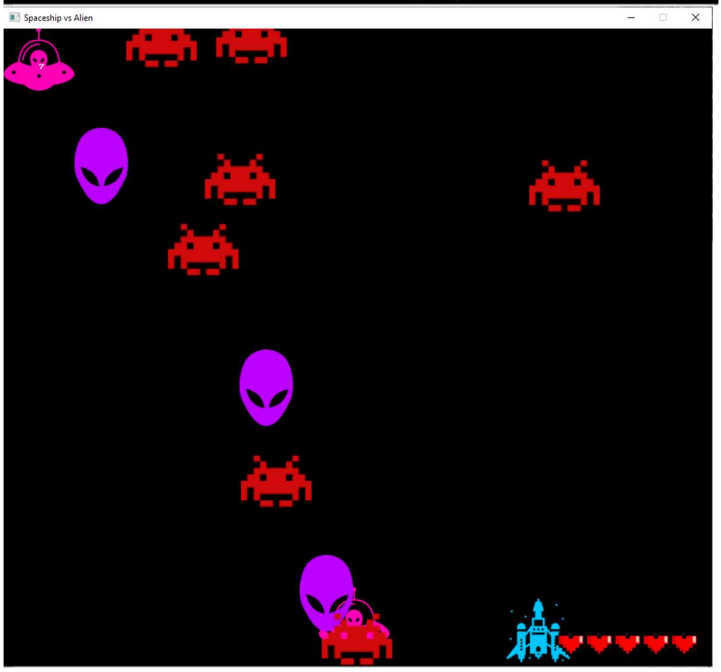
Rocketship must shoot radioactive logo for reduce final boss's lives

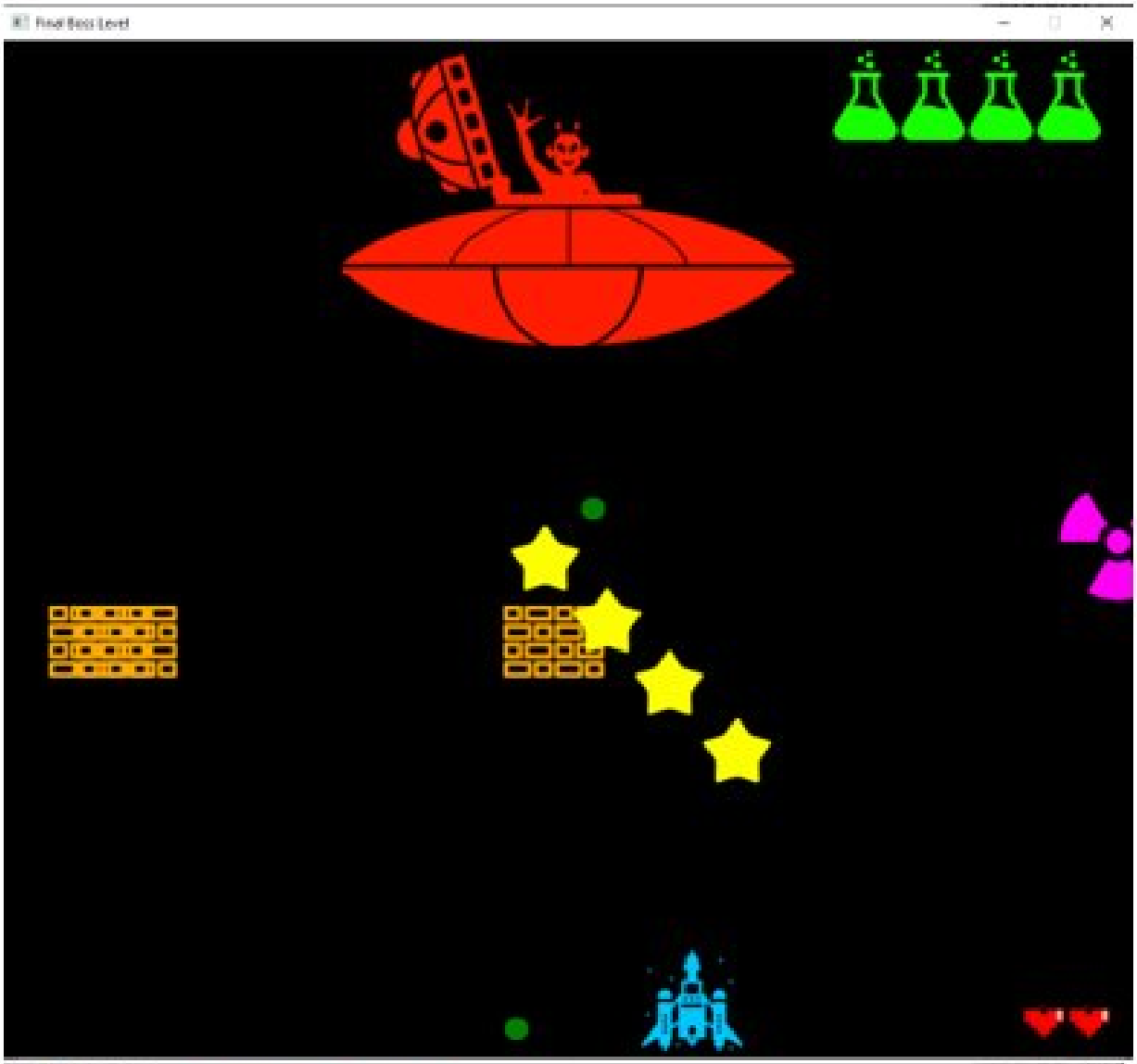
Final boss will shoot the Rocketship(player)

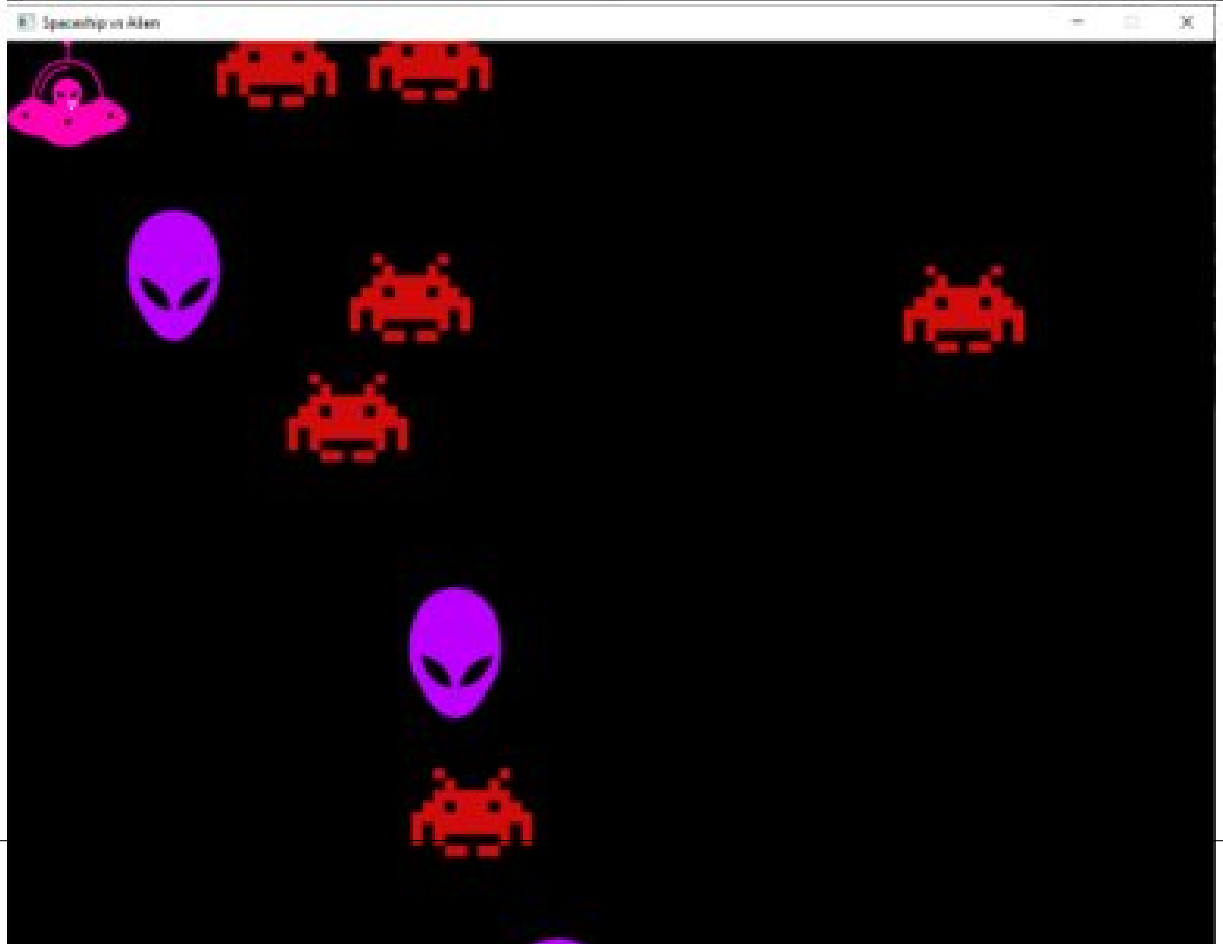
Radioactive have shield for defend radioactive

The shield has a different type of shield


Final Boss Level


Spaceship vs Alien


Final Boss Level


Spaceship vs Alien

32 Something Awesome

Create something awesome

Outcome	Weight
Evaluate Code	♦♦♦♦♦

Something awesome is awesome thing to do it.

Outcome	Weight
Principles	♦♦♦♦♦

Something awesome is awesome thing to do it.

Outcome	Weight
Build Programs	♦♦♦♦♦

Something awesome is awesome thing to do it.

Outcome	Weight
Design	♦♦♦♦♦

Something awesome is awesome thing to do it.

Outcome	Weight
Justify	♦♦♦♦♦

Something awesome is awesome thing to do it.

Date	Author	Comment
2019/10/02 16:06	Achmad Kemal	https://youtu.be/BrXk2sC0AC4
2019/10/02 16:07	Achmad Kemal	The video length is 20: 50
2019/10/02 16:08	Achmad Kemal	Can you check my video?
2019/10/02 16:19	Achmad Kemal	i explain about inheritance, abstraction and help others
2019/10/02 17:30	Achmad Kemal	Hey Dr Mengmeng Ge, can you check my work for this task? Sorry for late submission
2019/10/02 17:31	Achmad Kemal	Ready to Mark
2019/10/02 17:31	Mengmeng Ge	Time Exceeded
2019/10/02 17:31	Achmad Kemal	Can you check my work
2019/10/02 17:33	Achmad Kemal	?
2019/10/02 17:33	Achmad Kemal	Please
2019/10/02 18:56	Mengmeng Ge	Ready to Mark
2019/10/02 19:11	Mengmeng Ge	Thx. There are non-recognized characters in your submission in the evidence section. There are typos in your slides (e.g., inheritance instead of inherintance in your basketball player example). Please include the correct evidence part, double check the grammar and resubmit the report.
2019/10/02 19:11	Mengmeng Ge	Fix and Resubmit
2019/10/02 19:12	Mengmeng Ge	There is no need to record the video again.
2019/10/02 22:10	Achmad Kemal	Ready to Mark
2019/10/02 22:20	Achmad Kemal	Ready to Mark
2019/10/02 22:39	Achmad Kemal	i already put youtube link for see the slide
2019/10/02 23:05	Achmad Kemal	i delete power point slide screenshot because make the file have some issue
2019/10/03 00:48	Achmad Kemal	And now the submitted file is readable to you
2019/10/03 07:46	Mengmeng Ge	Complete
2019/10/03 07:50	Mengmeng Ge	Please note abstraction is different from abstract class.
2019/10/03 07:50	Mengmeng Ge	Abstraction is similar to the concept of a black box. Input goes in, black box does something, output comes out. It doesn't matter what happens in the black box, all you have to know is that it works.
2019/10/03 07:51	Mengmeng Ge	Like the menu option used in the program.
2019/10/03 08:02	Achmad Kemal	Okay i will remember that
2019/10/03 08:02	Achmad Kemal	Thank you for your advice

Deakin University

Object Oriented Development

OnTrack Submission

Something Awesome

Submitted By:
Achmad Mustafa Kemal
akemal
2019/10/02 22:20

Tutor:
Mengmeng Ge

Outcome	**Weight**
Evaluate Code	♦♦♦♦♦
Principles	♦♦♦♦♦
Build Programs	♦♦♦♦♦
Design	♦♦♦♦♦
Justify	♦♦♦♦♦

Something awesome is awesome thing to do it.

October 2, 2019

SIT771

Name: Achmad Mustafa Kemal

ID: 219374683

Details:

Title: Abstract and Inheritance

Content:

- Abstract
- Inheritance
- Help others

YouTube Link: https://www.youtube.com/watch?v=BrXk2sC0AC4&t=3s

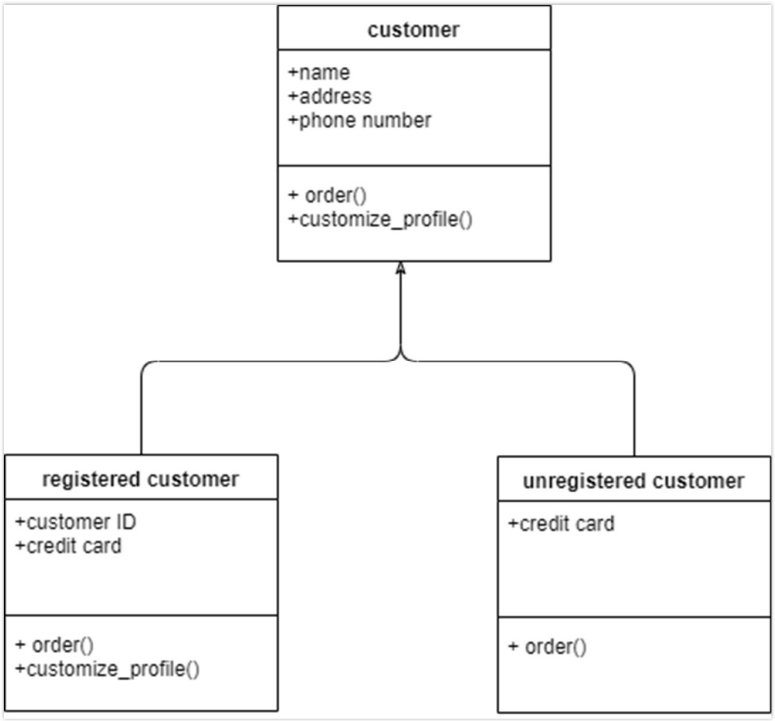
Also, I include power point in the end explanation of my note of my presentation.

The reason why I choose abstract, inheritance and help others(dedication):

Because some people confuse about abstract principle and they think abstract and interface are same principle. So, I want to discuss about Abstract. Let them know that abstract is difference between abstract and interface. I also give some example like UML class diagram and code example. Also, I explain about Inheritance in the video. People still not sure about inheritance principle. People still not sure about the function of inheritance. I think inheritance very useful when I have task for this subject. Such as, I build a program for task 7.2 (different type robot) and 7.1 (abstract transaction) but few of classmate still confuse about inheritance principle. Now, I going to explain about inheritance principle. Also, I put robot dodge code because the program implemented inheritance principle.

Note from my presentation:

- **Abstraction**
  - *Abstraction is Abstraction is a fundamental characteristic possessed by an entity (object) that distinguishes the entity from all other types of entities.*
  - *With abstraction, by compiling an entity's basic features, we can decrease com plexity.*
  - *Class that contains the abstract keyword with some of its methods (not all abstract method) is known as an Abstract Base Class.*
  - *All abstract techniques must be used by derived classes, otherwise they stay ab stract*



- **(UML Example)**
    - *Abstract class is customer*
    - *Abstract method is order and customize profile*
    - *Derived class are registered customer and unregistered customer*

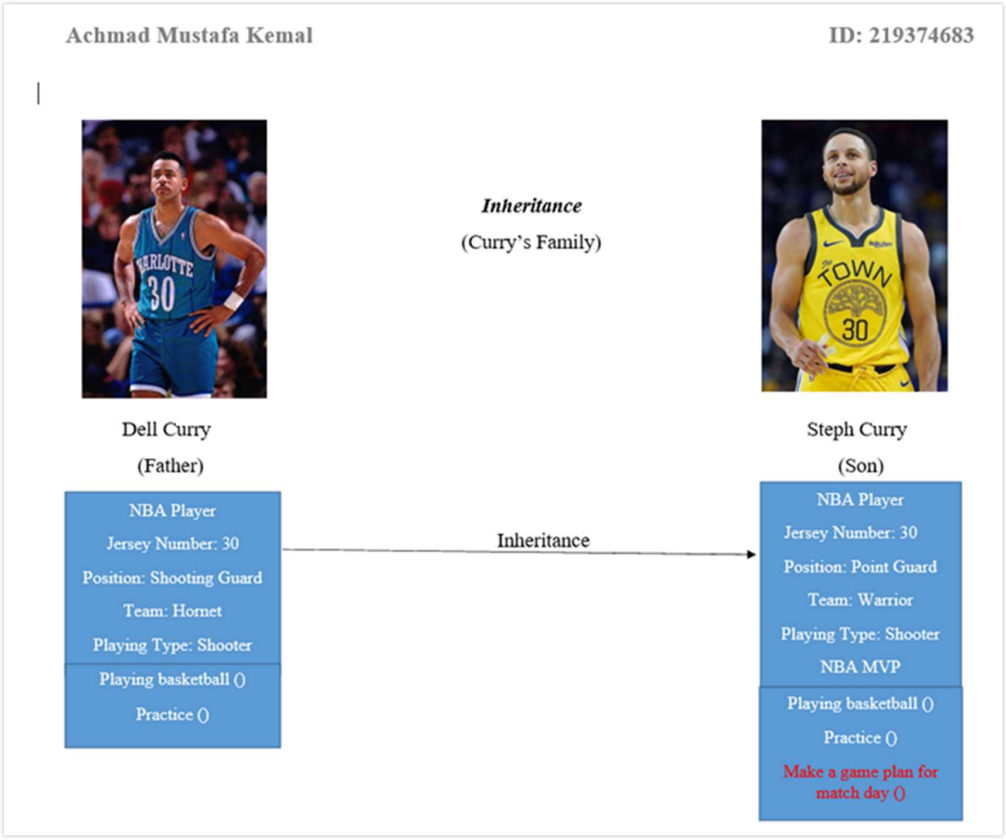
- **Abstract Vs Interface**
- *Abstract Class is a class specifically created for inheritance purposes.*
- *The purpose of making this abstract class is to make a general definition for the classes that will be derived from it.*

    - **Abstract**
        - *Methods can be static It contains both declaration and definition part. It contains constructor.*
        - *An abstract class can provide complete, default code and/or just the details that have to be overridden.*

- *Interface defines a (signature) of a collection of methods without a body.*
    - **Interface**
        - *Method cannot be static. It contains only a declaration part. It does not contain constructor.*
        - *An interface cannot provide any code, just the signature.*

Inheritance

The definition of inheritance is a class that have dependency with another class. Inheritance is a method that can make class have same characteristics with their base class with inheriting a class. Inheritance can adopt an object that have same entity. A class that have derivative class call as a parent class or base class and derivate class call as a subclass.



In the example of inheritance, we have two basketball players, such as, Dell curry and Steph Curry. Dell Curry is father from Steph Curry. So, Dell Curry is a parent class. On the other hand, we can assume that Steph Curry is a subclass because he is a son from Dell Curry. Dell Curry and Steph Curry are both have common information. Such as, both of them are NBA player, they also same wear jersey number 30. They also like shooting ball. But there are some behaviours and information that they both don't match. Such as, Dell Curry is a player from hornet and Steph curry is a player from warrior. Also, Steph curry is a point guard. So, Steph curry's job is handling their teammate in the basketball but Dell Curry is just shooter in the basketball court. In addition, Steph Curry is an NBA MVP and NBA Champion but Dell Curry is never be an NBA MVP. They also have somewhat similar but not entirely the same. For example, Dell Curry and Steph curry are also playing basketball and practice their skill but only Steph Curry have behaviour that Dell Curry doesn't have like make a game plan for match day. Only Steph Curry can make a game plan for match day but Dell Curry doesn't have it that behaviour.

(Show robot dodge code for example)

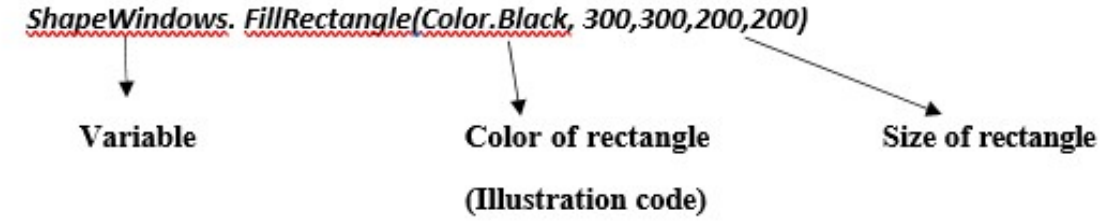
Help other

Evidence:

Student Name: Atif Shehzad Task:

Task: Shape Drawing (1.2)

He doesn't know to how to draw a Rectangle in the window. I tell about it. With using "FillRectangle", he can draw rectangle in the windows. The concept of code will like this:



(Illustration code)

So, I tell him before you use FillRectangle, you must have declared variable that can use for draw FillRectangle. After that, he can create shape and adjust color and size whatever you want.

Task: Another Language (9.1)

He doesn't know about phyton because he never attends the class and he never about the fundamental of phyton. So, I explain to him about and guide him to write a phyton code.

Task: Making a scene (1.4)

He still unfamiliar with C# programming language and he don't know how to install Splash kit and write a C# code in the visual studio. He told me about his struggle. He also doesn't attend the first lecture in this tri semester because he just arrived in Australia. He doesn't know about Splash kit. In this task, we must setup our splash kit manager into our C# files. So, we create the object in the visual studio and run it. After that, I teach him how to setup splash kit manager and write a C# code. After I show how to setup it, he can do setup and work it with the task

Task: Validating Stock Action (3.2)

Atif confuse about the UML class diagram that given in the instruction. He doesn't know what is "+AddStock(quantityAdded: int): void". Also, he doesn't know about how to add method to stock class and add validation. Atif still struggle with C# again because he never heard about C# programming language. I show how to make add stock method and add validation. I tell him that everything "+" plus symbol mean public. So, the code will be like this:

```
public bool AddStock(int quantityAdded)
                                        {
                                    .... add code here
                                        }
```

I tell him about Boolean value that can used to declare variables to store the Boolean values, such as, true and false. Now, he already knows how to read UML class diagram and can write code about addstock method.

Student Name: Muni Swetha Sai Cirimavilla

Task: Warehouse (5.3)

Swetha confuse about method overloading in this task. So, I explain about method overloading. The task said that ExecuteTransaction listed three times in the UML class diagram. So, I said to her about how to method overloading works in the task. Method Overloading is a method that can use same time with using different parameters. The concept of code will like this:

```
public void ExecuteTransaction(StockSaleTransaction StockSaleTransaction)
{
            .... add code here

}
public void ExecuteTransaction(StockPurchaseTransaction, StockPurchaseTransaction)
{

            .... add code here

}
public void ExecuteTransaction(StockAdjustmentTransaction,StockAdjustmentTransaction)
{

            .... add code here

}
```

After I explain to Swetha about method overloading, she decides to try to write a code by herself and now she knows and understand about method overloading

She also has problem with code indentation with her code. I show to her about code formatting. In the visual studio, you can use "***shift+alt+f*** to format documentation. So, Swetha can understand her code herself.

Task: Abstract Transaction (7.1)

In this task, Swetha doesn't use inconsistent variable, such as, sometime she uses "toWarehouse" and sometime she uses "toWare" in the program. This behavior can make people hard to read the code and hard to fix the error. I suggest her to make consistent variable and simple variable. Now, she uses consistent and simple variable for her task. Her problem with this task is solved.

Student Name: Ankit Nakra

Task: Warehouse (5.3)

In this task, he wants me to explain about public void AddStock because the task wants to us to use Public void for AddStock method and not to use public static void. So, I explain about definition of public void and public static method. I said that public void is a method that contain that will be executed when called and public static void is a method that can access without creating a class. In this task, add stock method only have statement that every user add new stock in the program and the method will passed into the Warehouse's list of stock item. In the end, Ankit is now understanding and know the difference between public void and public static void after I discuss and explain to him.

Student Name: Richard William Hester

Task: Name Tester (3.1)

He told me that he can't focus with his study that he doing right now. Even he knows about C# programming but he said to hard to get back study again. He can't focus on many things in the same time. I explain about this task. On this task, we must create simple program that guess name and guess the number. So, the user guess number between 1 and 100 and the program will be letting if the guess number are lower or higher. I showed my code to him as reference.

Task: Document Design (6.1)

He doesn't know about visibility of class members. So, I explain to him about what is a visibility of class member, such as, + mean public, - mean private, # mean protected and ~ mean package. Also, I explain about relationship arrow. I said there are many types of relationship, such as, association, inheritance, implementation, dependency, aggregation and composition. So, association is a common relationship between 2 classes. Inheritance is a type of relationship where one linked class is a child class by virtue of assuming the same information and functionalities of parent class. Implementation is a relationship between two classes where one model element executes the behavior that another model element identifies. Dependency is relationship between two classes that each class have dependency with another.

Student Name: Marina Liu

Task: Moving the player (3.3)

Marina can't get put player stay on window. She struggles with how to put player stay on window when player move in the window. I also found this one hard to put player stay on window. With using Lecture's feedback, I suggest her that she can use gameWindow .Width in the condition. Also, I suggest her that she can use gameWindow.Height for check if player hit bottom of window. In addition, she needs to consider gap of window

So, the concept of code will like this if the player wants to stay on window:

```
public void StayOnWindow(Window gameWindow)
{
const int GAP = 10;
// X is from public player
// Y is from public player
        if (X < GAP)
        {
        X = GAP;
        }
        if (X > gameWindow.Width - Width - GAP)
        {
        X = gameWindow.Width - Width - GAP;
        }
        if (Y < GAP)
        {
        Y = GAP;
       if (Y > gameWindow.Height - Height - GAP)
       {
       Y = gameWindow.Height - Height - GAP;
       }
}
```

After I explain about stay on window method for the player, she found the player stay on window. With the code that I showed to her, she can move the player to edge to window and still stay on window.

Task: Messy code (4.2)

In this task, I realized that this task about code indentation and code refactoring. It is because the source code from this task is very messy and hard to find it. Also, I found there are some variable that hard to understand. In the instruction, we must tidy up the code format and fix the issue. I suggest her to make better code formatting with using visual studio code. I suggest type "***Shift+Alt+f"*** to make code looks nicer than before. After, she can fix another problem like change variable to easy read and understand. In the end, she doesn't confuse to read this task code and she can run the code.

Task: Concept Visualization 1 (4.4)

She struggles with this task because she doesn't understand about control flow sequence. So, I told her that control flow sequence is known as basic control flow. I suggest her to use if statement. The reason why I suggest her to use IF statement because this statement is very basic statement that used in the C# programming language to make value is true or false. My explanation is very useful to her to understanding sequence control flow in this task.

Student Name: Melissa Martina Aranha

Task: Moving the player (3.3)

She can't handle the player when the player hit the bottom. So, I told her that with using gamewindow height, player's height and GAP. So, the concept of code will like this:

```
if (Y > gameWindow.Height - Height - GAP) { Y = gameWindow.Height - Height - GAP; }
```

After I explain to her, she solves the problem.

Task: Messy Code (4.2)

She also found same problem with Marina. She can't read the messy code in this task. I always to suggest to do code indentation when they found code that hard to read and understand. I suggest type "***Shift+Alt+f"*** to make code looks nicer than before. After, she can fix another problem like change variable to easy read and understand. In the end, she doesn't confuse to read this task code and she can run the code. After that, she realizes that bullet shot the rocket. I suggest find human error that type wrong section. Then she found it. I told her that there is some variable that are hard to read and put in the wrong place.

Student Name: Luke Sciberras

Task: Moving the player (3.3)

He also struggles with player stay on window when player move to left and bottom. I also found difficult to make player stay on window. So, I also fix this problem with another my classmate. Marina, Melissa and Luke are face same problem in this task. I remember that lecture give an advice to how to fix the problem. For Luke, he can use gameWindow .Width in the condition. Also, I suggest him that he can use gameWindow.Height for check if player hit bottom of window. In addition, he needs to consider gap of window.

The concept of code will be like this:

```
if (X > gameWindow.Width - Width - GAP){X = gameWindow.Width - Width - GAP;}
```

```
if (Y > gameWindow.Height - Height - GAP) 119 { 120 Y = gameWindow.Height - Height - GAP; }
```

In the end, Luke fix the problem after I explain how use gamewindow.Height, gamewindow.Width, player height and width and gap can check the edge of game window.

Task: Concept Visualization 1 (4.4)

Luke confuse about selection control flow and repetition control flow. For my example, I use switch statement for selection control flow because switch statement is a statement that can handle many cases in the program. I also give real example for switch case. I said that switch case usually uses for menu option function in the C# programming language. For repetition control flow, I said repetition control flow is like while statement. While statement used to execute statement while Boolean expression value is true. After short explain that I gave to Luke, he now can give example for his control flow and now he also more understands about selection and repetition control flow.