CPT306 Individual Project
Coursework Assignment Specification

2021/22 Semester 2
Bachelor Degree – Year 4

Module Code	Module Leaders	Module Title
CPT306	**Hai-Ning Liang** **Nan Xiang**	**Principles of Computer Games Design**

Coursework Assignment Number: **1 of 3**

Method of Working: **Individual**

Coursework Title: **Creating a 2D Game**

Percentage (%) Weighting: **25% of the overall module marks**

Date and time of publication: **10:00 am on Sunday, 28 February, Week 2**

Date and time for submission: **11:59 pm on Sunday, 3 April, Week 6**

General Instructions

1. One copy of this assignment should be handed via the module **Learning Mall** page at http://learningmall.xjtlu.edu.cn no later than the time and date shown above, unless an extension has been authorized by the module leader.

2. Before submission, each student must complete module coursework submission form obtainable from the module **Learning Mall** page. This assignment is being marked by student name and id, please ensure that you complete the correct coursework submission form.

3. Format of the coursework assignment submission:
   A **ZIP** file submitted via the **Learning Mall** module page containing the deliverables outlined in the "**What to Submit**" section of the coursework assignment specification.

4. Use of unfair means:
   You are reminded of the University's Code of Practice on the Use of Unfair Means and that the work you submit for assignment should contain no section copied in whole or in part from any other source unless where explicitly acknowledged by means of proper citation.

5. Late penalties:
   For work submitted late the penalty is loss of **5%** marks per day. Work that is **5** or more days late will automatically be graded as **FAIL**, and no re-submission will be allowed.

The story so far…

In the year of 3040, you are in a Space Fighter, positioned in the geostationary orbit, assigned a job of collecting and grouping ionised particles into the solar power system of the international space station. Unexpectedly, space debris is mixed with ionised particles.

Scenario

You have been tasked with destroying the space debris and to group the associated ionised particles into the storage area. The complete implementation of the game should look like the following illustrative image (see Figure 1).

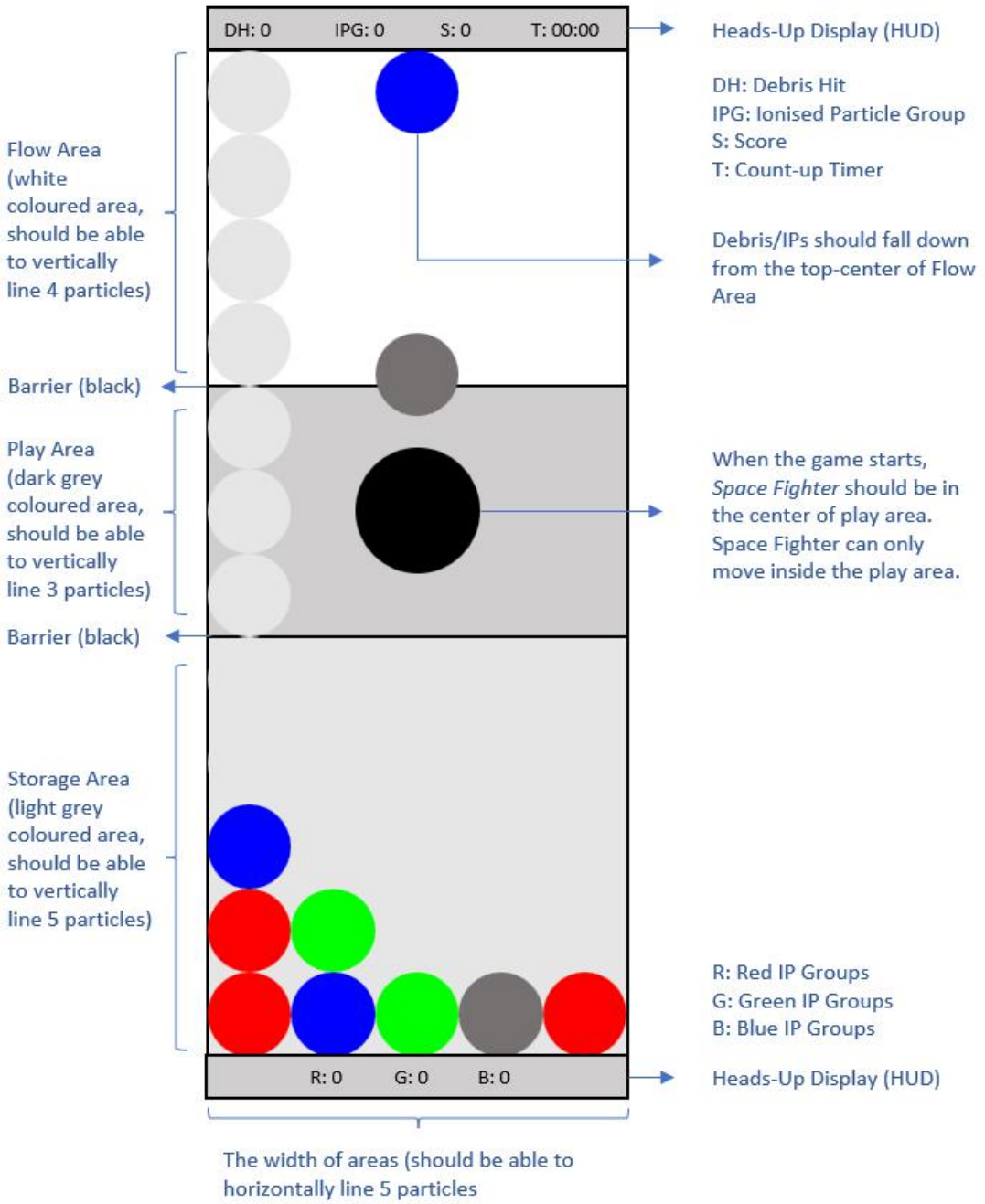

Figure 1: Illustrative image of the complete implementation of the game including details of the game elements/components.

Game Play

The player should be able to direct the *Space Fighter* by easily moving in the dark grey coloured Play Area (See Figure 1 above). The black barriers should block the motion of the *Space Fighter* towards both the flow and storage areas. Thus, a collision with the black barriers will stop the motion of the *Space Fighter*.

Ionised particles (red, green, and blue) and debris (grey) should fall down towards the play area at a random sequence. The player should be able to destroy the debris and manipulate the ionised particles by pushing each one sideways using the *Space Fighter*, with the aim of creating a vertical line or a horizontal line of three same-coloured ionised particles as shown in the Figure 2. When such a same-coloured vertical or horizontal line is created, it disappears, and any particles or debris above the disappeared line should fall down. At the same time, the player gets certain scores.

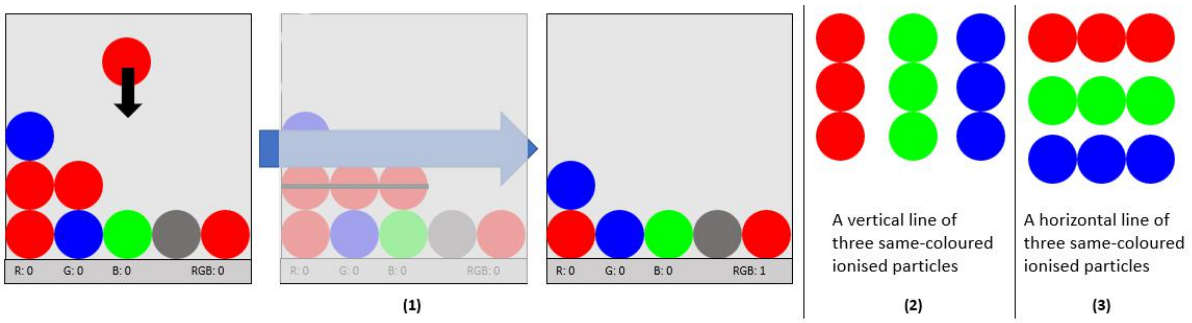

Figure 2: (1) An example of creating a horizontal line of three same-coloured ionised particle. (2) Aimed vertical lines. (3) Aimed horizontal lines.

The left-click on the *Start* button (should be implemented by you, explain more in Game Menu section) begins the game and a timer starts counting-up from 0.

The player will win the game when s/he successfully reaches 100 points. As shown in Figure 3, the player will lose when (1) the storage area is filled with non-lined ionised particles, (2) the storage area is filled with more than 5 debris, or (3) *space fighter* is destroyed because two particles/debris are overlaying it.

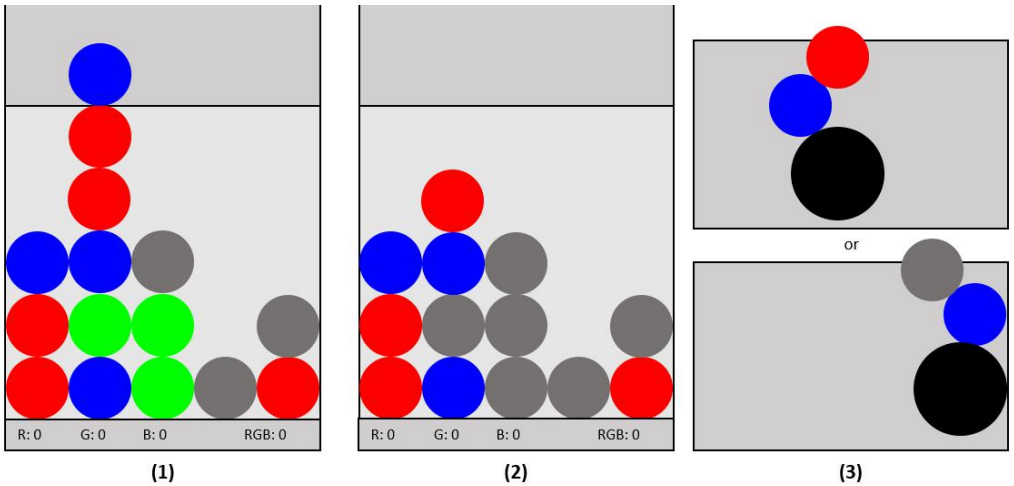

Figure 3: Examples of the situations when the player loses the game.

Implementation

You have been asked to write your own code to implement the game using Unity and C#. The resources for the game should be created by you, including the *ionised particles*, *debris*, *space fighter*,

notification messages, and *user interfaces*. However, your implementation should follow the specification detailed below.

Play, Flow and Storage Areas

The following measurements should be used to implement the Flow, Play, and Storage Areas in your game (see Figure 1).

Location	Height (Capable of storing)	Width (Capable of storing)
Play Area	3 particles (vertically lined)	5 particles (horizontally lined)
Storage Area	5 particles (vertically lined)	5 particles (horizontally lined)
Flow Area	4 particles (vertically lined)	5 particles (horizontally lined)

Table 1: Specification of Flow, Play, and Storage Areas

Space Fighter

The *Space Fighter* should appear at the centre of the Play Area when the game starts (see Figure 1). The black barriers must block the movement of the *Space Fighter*. The *Space Fighter* must be a circle and the size must be 1.5 times the particle size. Your own design of the *Space Fighter* is also required. For example, you may attach a text symbol or a picture on the *Space Fighter* or add some visual effects to the *Space Fighter*'s movement.

Control

The player should be able to play the game using the following keyboard and mouse controls:

Control	Function
Left-click (mouse)	Begins the game and starts the timer.
Right arrow	Moves *Space Fighter* towards the right side
Left arrow	Moves *Space Fighter* towards the left side
Up arrow	Moves *Space Fighter* upwards.
Down arrow	Moves *Space Fighter* downwards.
Spacebar	Pause/resumes the entire game.

Table 2: Specification of Player Controls

Ionised Particles and Debris

The ionised particles are represented in red, green, and blue, while the debris is characterized in grey using the RGB codes listed in Table 3.

Name	Colour	RGB
Debris	Grey	128, 128, 128
Ionised Particle Red	Red	255, 0, 0
Ionised Particle Green	Green	0, 255, 0
Ionised Particle Blue	Blue	0, 0, 255

Table 3: Specification of Debris and Ionised Particles

The ionised particles/debris should start falling from the top-centre (as shown in the Figure 1) of the flow area in a random sequence when the game is started. One important condition is that the same-coloured particles should not be immediately following. The particles/debris should fall down at a delay of 2 seconds. In other words, for every 2 seconds new particles/debris should start to fall down in the mentioned location. The ratio between the move speed of *Space Fighter* (for all directions) and

the falling speed of the ionized particles/debris is 3:2. Please adjust the speeds to make the game in a moderate difficulty.

The ionised particles should be pushed freely by the *Space Fighter*. The upper 1/4 surface of the *space fighter* can be used to destroy debris, while other areas will only push the debris (see Figure 4). You are encouraged to specify this area on the *Space Fighter*.

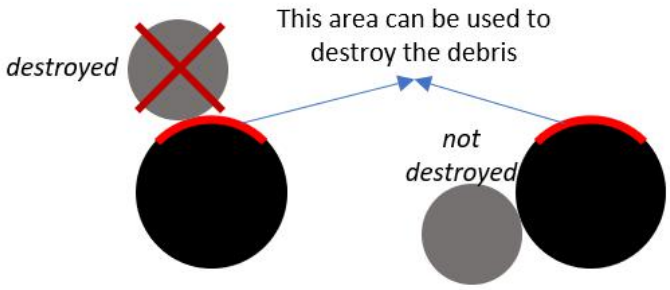

Figure 4: To destroy the debris.

Scoring

The scoring formula for the game should be built by counting the number of debris destroyed and by grouping same-coloured ionised particles together.

Description	Scores
Destroy a single debris	5 points
Group 3 same-coloured particles in a vertical or horizontal line	10 points
Group a vertical line and a horizontal line at the same time.	30 points

Table 4: Specification of Scoring

Heads-up display (HUD)

There is at least one, at most two heads-up display (HUD) areas to be implemented to provide the following required information listed in Table 5. The HUD(s) must be at the top or at the bottom of the interfaces, that is, above the flow area or below the storage area. Figure 1 shows an example of having two HUDs. You can decide the number of HUDs and its (their) position, but make sure the required elements are included.

HUD	Description
HUD	DH: the number of debris hit
	IPG: the number of ionized particle groups allocated
	S: scores achieved
	T: playtime
	R: the number of red particle groups allocated (vertical/horizontal lines)
	G: the number of green particle groups allocated (vertical/horizontal lines)
	B: the number of blue particle groups allocated (vertical/horizontal lines)

Table 5: Specification of the elements in heads-up displays

Notification Messages

The win and lose messages should be displayed as a pop-up display message while freezing the game window (flow, play, storage areas and heads-up display area). Table 6 shows the four different

messages should be implemented with the specified text colour. Except the message and the text colour, you can have your own design on the pop-up display.

Reason	Message	Text Colour
To Win	Congratulations! You Won!	Yellow
To Lose – by filling with non-lined ionised particles	Opps! Ionised particles are not successfully lined in the storage!	Orange
To Lose – by filling with more than 5 debris	Opps! Debris are filled in the storage!	Black
To Lose – *Space Fighter* has been destroyed because 2 particles/debris are overlaying it	Opps! Space Fighter has been destroyed!	Red

Table 6: Specification of the elements in heads-up displays

Game Menu

A *Start* button is required at the beginning of the game. Below requirements are completely optional/not marked.

- You can have a menu before the game starts and put the *Start* button in the menu. The menu can have other elements, such as a *Help* button (shows the game rule), an *Acknowledge* button (shows who developed the game), or a *Quit* button (quits the game).
- When you pause the game, it is good to have an interface to show the current pause status. For this interface, you can have a *Resume* button to resume the game and a *Restart* button to restart the game.
- It is also good to have a *Restart* button after the game finished.

Frequent Marks Reduction

- If the particles do not line up perfectly within the storage area, marks will be reduced.
- If the particles escape the game region marks will be reduced.

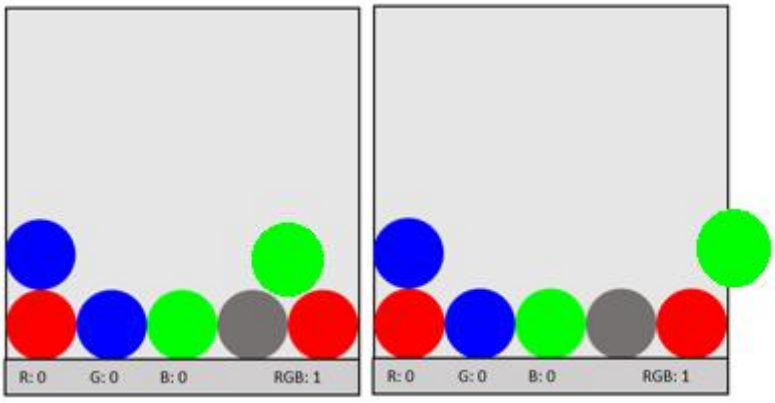

Figure 5: Example of particles not perfectly aligned or escaping the play area.

What to Submit

Your **ZIP** file should include these files:

- Submission form (.pdf). The submission form should be properly completed with your signature. It is available on **Learning Mall**. Submission form with incorrect information certainly will affect your marks, so carefully complete the submission form. The submission form should be properly named as mentioned below.

- Exported Unity package of your game (.unitypackage). Please export all the required source files in the package.

- Executable file of your game (.exe). Please build your game. Note that, the target platform should be **Windows**.

- Game specification (.txt). This document should indicate which version of Unity and the resolution setup you used to develop the game. If you have any special issues about the game, you can also write down here. Please make this document concise.

Please make sure all the required files can be opened and run properly on a **Windows** computer. Please use your **First Name**, **Last Name** and **Student Number** to name above mentioned files and the **ZIP** file—for example **Haining_Liang_999999** will be the name of the files module leader would be submitting, with 999999 being as his student number. Any submissions with improper or incomplete file names certainly will affect your marks, so carefully name your files.

The end of the document

***