DEPARTMENT OF COMPUTER SCIENCE
ASSESSMENT DESCRIPTION 2013/14 (EXAM TESTS AND COURSEWORK)

MODULE DETAILS:

Module Number:	08120	Semester:	2
Module Title:	Programming 2		
Lecturer:	RSM		

COURSEWORK DETAILS:

Assessment Number:	2	of		2
Title of Assessment:	Software Development Project			
Format:	Program	Demonstration		Report
Method of Working:	Individual			
Workload Guidance:	Typically, you should expect to spend between	4	and	20	hours on this assessment
Length of Submission:	This assessment should be **no** more than: *(over length submissions **will be** penalised as per University policy)*	N/A - coding exercise **words** *(excluding diagrams, appendices, references, code)*		

PUBLICATION:

Date of issue:	6th March 2014

SUBMISSION:

ONE copy of this assessment should be handed in via:	E-Bridge		If Other (state method)	
Time and date for submission:	**Time**	9:15	**Date**	Monday 5th of May
If **multiple hand–ins** please provide details*:*				
Will submission be scanned via TurnitIn?	No	If this requires a separate TurnItIn submission, please provide instructions:		
*Late submissions **will be** penalised as per university policy*				

The assessment must be submitted **no later** than the time and date shown above, unless an extension has been authorised on a *Request for an Extension for an Assessment* form which is available from the Departmental Office (RB-308) or http://intra.net.dcs.hull.ac.uk/student/exam/Advice%20regarding%20resits%20in%20modules%20passed%20by%20compe/Forms/AllItems.aspx.

If Turnitin is taking a long time to produce its analysis you must still submit your work, albeit initially without the Turnitin analysis. A delay at Turnitin cannot be used to excuse a late submission.

MARKING:

Marking will be by:	Student Name

COURSEWORK COVERSHEET:

BEFORE submission, you **must** ensure you complete the **correct** departmental ACW cover sheet (if required) and attach it to your work. The coversheets are available from: http://intra.net.dcs.hull.ac.uk/student/ACW%20Cover%20Sheets/Forms/AllItems.aspx	NO coversheet required as E-Bridge submission

ASSESSMENT:

The assessment is marked out of:	100	and is worth	40	% of the module marks

N.B If multiple hand-ins please indicate the marks and % apportioned to each stage above (i.e. Stage 1 – 50, Stage 2 – 50). It is these marks that will be presented to the exam board.

ASSESSMENT STRATEGY AND LEARNING OUTCOMES:

The overall assessment strategy is designed to evaluate the student's achievement of the module learning outcomes, and is subdivided as follows:

LO	Learning Outcome	Method of Assessment *{e.g. report, demo}*
1	*Demonstrate knowledge and understanding of problem analysis, data types and objects*	Program/Presentation/Report
2	*Make use of appropriate tools to create and deploy maintainable programs in an object oriented programming language. Analyse the approach and solution to the problem.*	Program/Presentation/Report
3	*Use abstraction for problem solving and process design. Analyse this approach and the solutions to the problem.*	Program/Presentation/Report
4	*Apply knowledge of the syntax and semantics of an object oriented programming language.*	Program/Presentation/Report

Assessment Criteria	Contributes to Learning Outcome	Mark
Application functionality	1,2,3,4	40%
Appropriate use of Library Framework	3,4	10%
Additional Features	3,4	20%
Appropriate Programmer Documentation	3,4	10%
Appropriate User Documentation	3,4	10%
Evidence of Good Design	1,2,3,4	10%

FEEDBACK

Feedback will be given via:	Verbal (via demonstration)	Feedback will be given via:	Feedback Sheet
Exemption (staff to explain why)			

Feedback will be provided no later than 4 'teaching weeks' after the submission date.

This assessment is set in the context of the learning outcomes for the module and does not by itself constitute a definitive specification of the assessment. If you are in any doubt as to the relationship between what you have been asked to do and the module content you should take this matter up with the member of staff who set the assessment as soon as possible.

You are advised to read the **NOTES** regarding late penalties, over-length assignments, unfair means and quality assurance in your student handbook, also available on the department's student intranet at:
- http://intra.net.dcs.hull.ac.uk/student/ug/Handbooks/Forms/AllItems.aspx (for undergraduate students)
- http://intra.net.dcs.hull.ac.uk/student/pgt/Student%20Handbook/Forms/AllItems.aspx (for postgraduate taught students).

In particular, please be aware that:
- Your work will be awarded zero if submitted more than 7 days after the published deadline.
- The overlength penalty applies to your written report (which includes bullet points, and lists of text you have disguised as a table. It does not include contents page, graphs, data tables and appendices). Your mark will be awarded zero if you exceed the word count by more than 10%.

Please be reminded that you are responsible for reading the University Code of Practice on the use of Unfair means (http://student.hull.ac.uk/handbook/academic/unfair.html) and must understand that unfair means is defined as any conduct by a candidate which may gain an illegitimate advantage or benefit for him/herself or another which may create a disadvantage or loss for another. You must therefore be certain that the work you are submitting contains no section copied in whole or in part from any other source unless where explicitly acknowledged by means of proper citation. In addition, **please note** that if one student gives their solution to another student who submits it as their own work, **BOTH** students are breaking the unfair means regulations, and will be investigated.

In case of any subsequent dispute, query, or appeal regarding your coursework, you are reminded that it is your responsibility, not the Department's, to produce the assignment in question.

Department of Computer Science

08120 Programming 2
ACW 2 2013/2014

You must select **one** of the two deliverables and write a program which has the required behaviours.

Banjos4Hire

The shop is the only banjo hiring shop in the country. Customers can register with the shop and they can hire up to five banjos at any given time. Each banjo has a particular value. A customer is only allowed to hire the most expensive banjos (which are worth more than 200 pounds) when they have had three successful hires of lower priced banjos.

Banjos are hired for up to a week by the customer. When they are returned they are inspected. If they are found to be damaged the customer is charged a fine of ten pounds in addition to the repair costs. If a customer damages more than three banjos their membership is revoked and they are not allowed to hire any more. If the customer does not return a banjo on time they are sent a letter and charged ten pounds for every day they keep the banjo. If a customer is late returning a banjo on more than ten occasions their membership is revoked.

The shop want to track their customers so that warning letters are sent on time and customers are banned when they have broken the rules. They are also thinking of starting a sister shop "Accordion Rental" and they want an easy way of identifying their biggest banjo customers who may be interested in renting accordions.

Data Storage

The program must store the following information

- The stock held in the shop as a collection of banjos
- The customers of the shop, held as a collection of customers
- All the rentals

For each banjo the program must store:

- A description of the banjo as a string of text
- The price per day of hiring that banjo
- The value of the banjo
- The state of the banjo (either in the shop, being repaired or out on hire)
- A unique banjo ID

For each customer of the shop the program must store:

- The name of the customer
- The collection of rentals that were made by that customer
- The number of banjos that have been returned late
- The number of banjos that have been returned damaged
- The number of banjos that are presently out on hire to that customer
- A unique customer ID

For each rental that was made by the customer the program must store:

- The ID of the banjo that was hired
- The date the banjo was hired
- The date the banjo was due back

- The state of the hire (awaiting pickup, out on hire, late being returned, returned)
- A unique rental ID

The information must be stored in a file and retrieved from the file automatically when the program starts. The first time the program runs it will create an empty file.

Program Functions

The set of behaviours that the customer has asked for is as follows:

- **Banjos:** Add a banjo to the store, find a banjo by id, edit the rental price of a banjo
- **Customers:** Add a customer to the store, find a customer by id, create a rental for a customer, find the rental status of a customer
- **Management:** Produce a list of currently active rentals

The system should detect invalid entries where appropriate. It should also store all the information in a file so that it can be recovered when the program is restarted.

User Interface

The system should provide a Windows Presentation Foundation based user interface which will allow the above system behaviours to be accessed. The program should use a single page which will contain an appropriate set of components.

Additional Application Features

Extra marks can be gained by implementing additional features. Some suggested enhancements are:

- **Finding by Name:** This enhancement allows the users to search for names of customers and banjos. If there is a name clash the system should show a list of matching names and addresses and allow the user to pick the required item. This enhancement is worth 20%.
- **Most Popular Banjos:** This enhancement allows the system to determine which have been the most popular banjos that have been hired by the shop. This enhancement is worth 10%.
- **Customer mailshot:** This enhancement allows the manager to identify customers who have not rented a banjo in the last three months. The system will prepare a list of these customers that can then be used to trigger a mailshot to them. This enhancement is worth 10%.
- **Calculate Total Stock Value:** The system should display the total value of the banjos in the store. This enhancement is worth 5%.

Note that in the assessment criteria, a maximum of 20% is allocated for additional features.

Solution Structure

The solution should be structured using classes to hold the required data. The system should be able to hold at least 50 different banjos and manage at least 100 customers.

Alien Banjo Attackers from Space
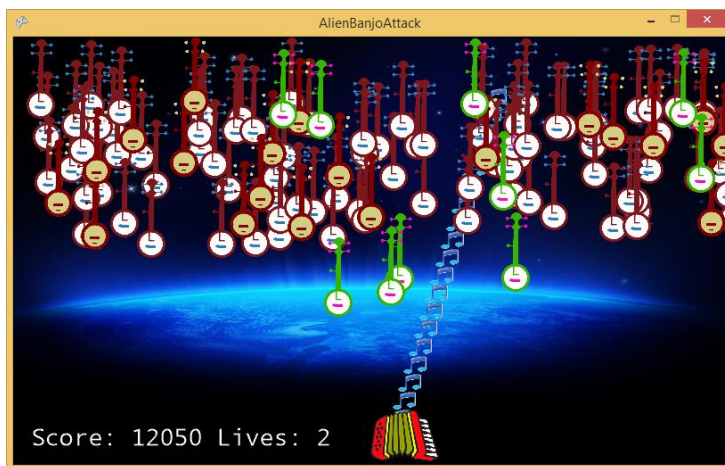
For this deliverable you are required to create an XNA program that plays the game of "Alien Banjo Attackers from Space". In this action packed space shooter the player must control the "Battleship Accordion" in its fight to save the planet from an attack of deadly space banjos. The player must shoot the banjos using musical notes and prevent them reaching the bottom of the screen. There are three types of killer banjo alien, if any of them touches the player they lose a life. After the player loses three lives the game is over.

The screenshot shows the view of the game in play. The ship can be controlled by the direction keys on the keyboard and the player presses the space bar to fire notes at the attackers. The player can fire a large number of simultaneous notes which move up the screen until they collide with a banjo or leave the top. The notes move vertically up the screen from the x position of the player where they were fired.

The aim of the game is to destroy all the banjos and last as long as possible.

The score and lives left are displayed at the bottom of the screen.

There are three types of enemy banjo, the "Plain", the "Hunter" and the "Deadly Strummer".

	Plain Banjo This alien moves from left to right. When it reaches the edge of the screen it drops down and then moves back. If it reaches the bottom of the screen the game is over. It takes one hit to destroy and is worth 10 points. If it collides with the player it is destroyed and the player loses a life.
	Hunter Banjo The Hunter banjo will behave like a plain banjo for five seconds and will then move towards the player. If it reaches the bottom of the screen the game is over. It takes one hit to destroy and is worth 20 points. If it collides with the player it is destroyed and the player loses a life.
	Deadly Strummer This moves towards the player as soon as it appears. It moves faster than the Hunter banjo. It takes two hits to destroy and is worth 50 points. If it collides with the player it is destroyed and the player loses a life.

Required Gameplay Features

The game must implement the above gameplay and have the following features:

- **Play Mode:** The system should run as an XNA game on a Windows PC, Xbox 360 or Windows Phone. It should be controlled by either the keyboard, a gamepad or by using the Windows Phone touch screen or accelerometer. It must have an "attract" mode which displays some gameplay and the highest score achieved so far.
- **Enemy banjos:** The enemy must appear in particular positions at the start of each level
- **Scoring of the game:** The game must display the score during the gameplay and track the highest score during a session.
- **Game Save:** At any point during the game the player **must** be able to press the S key or a button on the gamepad to save the entire game state including the positions of all the sprites and music notes along with the game score. The next time the game is started it should automatically load this state and continue the game from that point.
- **Graphics:** The game must display a backdrop and an image for each of the types of ship. The artwork assets above will be available for download. You can use your own artwork if you wish, but remember that this is not a graphic design assignment so great graphics will not attract extra marks.
- **Movement:** The player should be able to direct their ship and fire missiles by using the arrow keys on a Windows PC keyboard, Xbox 360 gamepad or by tipping the Windows Phone.
- **Storage:** The player should be able to stop playing the game at any point and return to it later. The game must store its state in the file store of the host computer and then when the game is resumed it should place all the objects back on the screen in the correct positions.

Additional Game Features

Extra marks can be gained by implementing additional features. Some suggested enhancements are:

- **Explosions:** When an enemy banjo is destroyed it can just disappear. You can add an explosion animation to the game when this happens. This enhancement is worth 15%.
- **Sound effects:** Use the sound playback features of XNA to produce noises when the banjos are displayed and notes fired. This enhancement is worth 5%.
- **"StrumFire":** Add the ability of the "Deadly Strummer" to fire notes back at the player. The Deadly Strummer will fire notes which are aimed at the position of the player at the time the note is fired. This enhancement is worth 10%.
- **"Windows 8 or Windows Phone":** You can make a version for the Windows 8 or Windows Phone platform using the MonoGame XNA framework. If you demonstrate your game running on these platforms the enhancement is worth 20%.

Note that in the assessment criteria, a maximum of 20% is allocated for additional features.

You are welcome to publish your games based on this mechanic, please let me know if you do this.

Solution Structure

The solution should be structured using Sprite class and a Playfield class that holds all the items to be drawn on the playfield. The game should be able to display at least 100 different sprite elements.

Important Note

You must complete the required features of your solutions **before** adding any extra ones.

Marks for extra features **will not be counted** if any of the required features are not provided by your solution.

Department of Computer Science Coursework Assessment Sheet

Module Number: 08120   Title: Program Development – Alien Banjo Attackers

Student Name: _____

Application Functionality – 40%

Background display, player movement	
Collision detection and scoring	
Enemy object behaviour	
Game storage and recovery	
"Attract Mode"	

Appropriate use of Library Framework– 10%

Correct use of LoadContent, Update and Draw	

Additional Features– 20%

Explosion Animations – 15%	
Sound Effects - 5%	
"Strumfire" – 10%	
Windows 8 or Windows Phone – 20%	

Appropriate Programmer Documentation – 10%

Inline comments conforming to C# convention	
Appropriate level of detail	

Appropriate User Documentation– 10%

Two page document with screenshots	

Evidence of Good Design– 10%

Sprite and Playfield classes	
Good game state management	

_____

_____

Date: _____ Student: _____ Marker: _____

Module Number: 08120   Title: Program Development – "Banjos4Hire"

Student Name: _____

Program Works Correctly – 40%

Customer creation and recovery	
Banjo creation and recovery	
Hiring of a banjo	
Listing active rentals	
Data storage and retrieval	

Appropriate use of Library Framework– 10%

Use of Forms and Components for display	

Additional Features– 20%

Finding by name - 20%	
Most Popular Banjos – 10%	
Customer Mailshot – 10%	
Total Stock Value – 5%	

Appropriate Programmer Documentation – 10%

Inline comments conforming to C# convention	
Appropriate level of detail	

User Documentation – 10%

Shows how program is started and features used	

Evidence of Good Design – 10%

Customer, Banjo and Rental classes	
Appropriate variable names and structure	

_____

_____

Date: _____ Student: _____ Marker: _____