UNIVERSITY OF Hull

DEPARTMENT OF COMPUTER SCIENCE
ASSESSMENT DESCRIPTION 2014/15 (EXAM TESTS AND COURSEWORK)

MODULE DETAILS:

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

COURSEWORK DETAILS:

Assessment Number:	1	of	1		
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:	20th February 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**	Tuesday 5th May
If **multiple hand–ins** please provide details:				
Will submission be scanned via TurnitinUK?	No	If submission is via TurnitinUK within E-Bridge students MUST only submit Word, RTF or PDF files. Students MUST NOT submit ZIP or other archive formats.		

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.

A student who has submitted the wrong file to E-Bridge will still incur a late penalty if their resubmission is made after the coursework deadline.

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 1 2014/2015

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

"Gallery X"

"Gallery X" is an upmarket art gallery based in a large city in the UK. It sells artwork in the form of paintings and sculptures. Artists apply to the gallery to be allowed to display their works. If the artist is accepted the gallery will exhibit their artwork for sale. An artist can bring up to 5 artworks for display in the gallery. The artist will set the price of an artwork.

If the artwork sells the gallery will receive 20% of the sale price and the artist will get the rest. The gallery never has more than 50 items on display at any time.

Artworks are displayed for up to two weeks after which they are returned to the artist if they have not been sold.

Customers can visit the gallery and reserve artworks that they wish to buy. An artwork can be reserved for two days, during which time it is not available for sale to anyone else. When a reservation expires the artwork is again available for sale. *Note that artwork reservation is an optional behavior (see below).*

Data Storage

The program must store the following information

- The stock held in the gallery as a collection of stock items.
- The customers of the gallery, held as a collection of customers
- The artists associated with the gallery, held as a collection of artists
- The reservations that have been made
- Sale records of any artwork that has been sold

For each stock item (piece of artwork) the program must store:

- A description of the item as a string of text
- Whether the item is a painting or a sculpture
- The sale price of the item.
- The date that the item was first displayed in the gallery
- The state of the item (either in the gallery, reduced in price, sold or returned to the artist)
- A unique stock item ID

For each customer of the gallery the program must store:

- The name of the customer
- The collection of sales that were made to that customer
- A unique customer ID

For each artist exhibiting in the gallery the program must store:

- The name of the artist
- The collection of stock items that have been exhibited by that artist
- A unique artist ID

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

- The ID of the stock item that was purchased
- The date the purchase took place
- A unique purchase 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 must create an empty file.

Program Functions

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

- **Artists:** Add an artist to the gallery, find an artist, find the stock items that the artist has exhibited
- **Stock Items:** Add an item to the gallery, select an item from a list of items and edit the item details, sell an item, find items that have been in the gallery more than two weeks
- **Customers:** Add a customer to the store, find a customer by id, create a sale for a customer, find out what items a customer has purchased
- **Management:** Produce a list of stock items currently on display in the gallery

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:

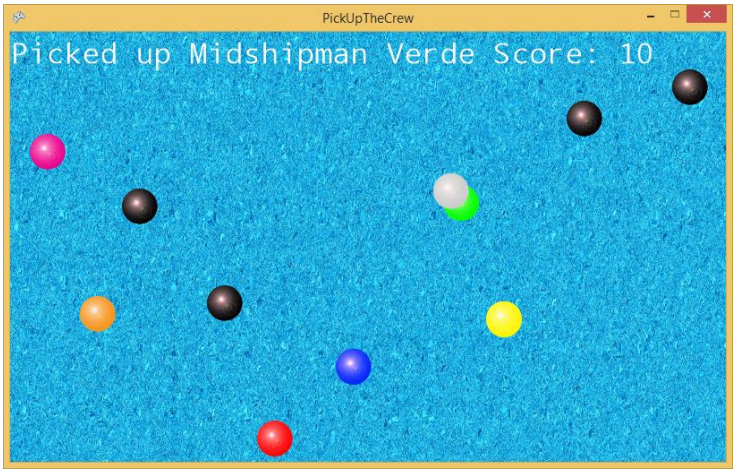
- **Price Reduction:** If an artwork is not sold after one week the gallery will contact the artist and suggest that the price be reduced for the second week. This enhancement is worth 20%.
- **Artwork Reservation:** This enhancement allows customers to reserve artworks that they wish to buy. An artwork can be reserved for two days, during which time it is not available for sale to anyone else. When a reservation expires the artwork is again available for sale. This enhancement is worth 10%.
- **Customer mailshot:** This enhancement allows the manager to identify customers who have not bought any artwork 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%.
- **Artwork Images:** The gallery will be able to attach an image to each stock item. The image will be displayed when the item is being edited or viewed. 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 100 different stock items and manage at least 100 customers.

Pick Up the Crew



For this deliverable you are required to create an XNA program that plays the game of “Pick up the Crew”.

The player controls "Captain White", of the good ship "Sinkeasy", which turns out to have been a rather poor choice of name. The player must go around and pick up all his crew, who were dumped into the sea when the boat went down. Each player is represented by a different coloured disk. The captain picks crew members just by touching them. The crew member then disappears from the sea and the score is updated

Also in the sea are a number of hungry sharks. The good news is that normally the sharks are not paying attention to anyone. The bad news is that if the captain or one of the crew get too close to a shark it will hear them and start chasing the captain.

- If a shark touches the captain or any of the crew the sharks win and the game is over.
- If the captain manages to pick up all the crew and escape from the sea the captain wins and the level is over. The game then continues with a new level, containing one additional shark. When the sharks and crew are placed on the screen it is important that they do not overlap, otherwise the game will be over before it starts.

You will have to adjust the hearing range of the sharks and the speeds at which they move to make the game as exciting and skillful as you can

The score of the game is displayed at the top of the screen, along with the name of the player who has been most recently picked up. The crew are as follows: "Lieutenant Sky","Midshipman Verde","Cabinboy Crimson","Stoker Orange","Able Seaman Shocking-Pink" and "Commander Custard".

Required Gameplay Features

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

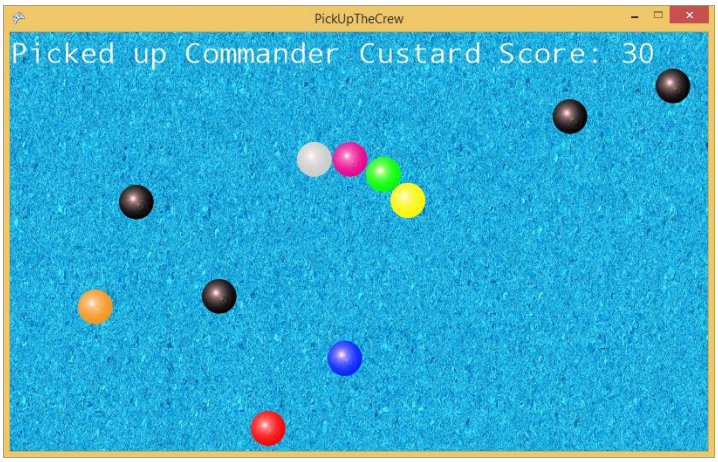
- **Play Mode:** The system should run as an XNA game on a Windows PC, Xbox 360 or Windows Phone. It should be controlled by either the keyboard, a gamepad or by using the Windows Phone touch screen or accelerometer. It must have an "attract" mode which displays some gameplay and the highest score achieved so far.
- **Crew:** The crew 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 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 crew. The artwork assets 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 the captain and pick up crew by using the arrow keys on a Windows PC keyboard, Xbox 360 gamepad or by tipping the Windows Phone or tapping the screen.
- **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:

Crew Snake

In the simplest form of the game the captain picks crew members just by touching them. The crew member then disappears and the score is updated. A more advanced version of the game has the crew following the captain in a "snake" after each one is picked up, as shown on the right. The captain must maneuver the snake to pick up each crew member in turn. This version of the game is more exciting because the crew becomes progressively harder to manage and the sharks have a bigger target each time a crew member is picked up. This enhancement is worth 15%.



Sound Effects

Use the sound playback features of XNA to produce noises when the crew are picked up and the sharks catch a crew member. This enhancement is worth 5%.

Heavy Crew

As the captain picks up more and more crew his movement speed should reduce making it easier for the sharks to catch up with him. 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.

Publishing Games

You are welcome to publish your games based on this mechanic, please let me know if you do this. The domain name pickupthecrew.com has been purchased, the student submitting the best game will be able to use this to help publish their result

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 – Pick up the Crew

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%

Crew Snake – 15%	
Sound Effects - 5%	
Heavy Crew – 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: _____

Department of Computer Science Coursework Assessment Sheet

Module Number: 08120 Title: Program Development – "Gallery X"

Student Name: _____

Program Works Correctly – 40%

Customer creation and recovery	
Artist creation and recovery	
Submission of artwork item	
Selling artwork item	
Data storage and retrieval	

Appropriate use of Library Framework– 10%

Use of Forms and Components for display	

Additional Features– 20%

Price Reduction - 20%	
Stock Reservation – 10%	
Customer Mailshot – 10%	
Stock Images – 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, Artist and StockItem classes	
Appropriate variable names and structure	

_____

_____

Date: _____ Student: _____ Marker: _____