ICS 3M0 Video Game Project Details

Description

Each student in this course will develop his/her own video game in Visual Basic (VB). Students are discouraged from developing "shoot 'em up" games because VB has poor graphics support. Obtaining satisfactory results in VB for such games requires calls to APIs (Application Programming Interfaces) such as Windows GDI, DirectX and Open GL. Learning to use APIs may require far more time than is available in a one semester course. Students are encouraged to develop strategy games, puzzle games or any games that do not require a great deal of graphical processing power.

Notes

1. Students who have a strong desire to develop "shoot 'em up" games or other games requiring a great deal of graphical processing may still be able to do so without using APIs. Certain clever tricks can be employed to improve the performance of such games. Extremely motivated and hard-working students may even be able to learn to use APIs in the short time that we have.
2. Students who are not interested in developing a video game may choose a different project of comparable difficulty.

Due Date

Your game will be due some time in the ***last two weeks of classes***.

Details

Evaluation Criteria

The game that you develop will be judged according to the following criteria:

1. Coding Practices (Style)

(a) The code should be logical, tidy and constructed according to the general guidelines learned throughout the course (i.e. proper indentation, comments for major blocks of code and abstruse code, meaningful identifier names, etc).

(b) The code should be as short as possible.
Duplicate code should be eliminated by using subs and functions.
Needlessly long and messy code should be shortened by using loops, arrays, control arrays, etc.

2. Difficulty of Coding

(a) Games that are difficult to code will be given more credit than those that are easy to code.

(b) The code should include a large number of the programming techniques learned in ICS3M0.

3. User Interface

The user interface should be attractive, well organized and user-friendly.

4. How Interesting and Challenging is the Game?

Games that are interesting and challenging will receive more credit than those that are not.

Evaluation Guide (Rubric)

A detailed evaluation guide will be given in a separate document.

Game Ideas

Card Games – Blackjack, Euchre, Hearts, Crazy Eights, Go Fish,…

Board Games – Monopoly, Snakes and Ladders, Battleship, MahJongg,…

Puzzle/Strategy Games – Tic-Tac-Toe (very easy), Connect Four, Othello (Reversi), Minesweeper, Towers of Hanoi, …

Word Games – Anagrams, Boggler, Word Jumble, Word Morph, …

Role Playing Games (RPGs) – There are many possibilities in this category!

How to find other Game Ideas

- Visit game sites such as "Yahoo® Games"
- With a search engine such as "Google®," use search phrases such as "games," "board games," "strategy games," "role playing games" and "puzzles"
- Talk to friends/classmates/teachers/parents
- Visit a store that sells board games and/or other types of games.

Before you Begin…

Please, please ask me for advice before you begin. I need to confirm that the game that you have chosen is appropriate to your skill level and our time constraints. ***In addition, do not begin coding until you have a sound overall plan and you have chosen suitable algorithms!***

ICS3M0 Final Culminating Activity (Game Project) Evaluation Guide

Final Evaluation Summary	
Game Project: 10%	Final Examination: 20%

Victim: _____

Categories	*Criteria*	*Descriptors*					*Level*	*Average*
		Level 4	Level 3	Level 2	Level 1	Level 0		
Knowledge and Understanding (KU)	**Understanding of ICS3M0 Programming Concepts** To what level has the student coded the solution in a manner that takes advantage of the programming concepts and constructs learned in ICS3M0? To what extent has the student included ICS3M0 concepts and constructs in the solution?	Very High	High	Moderate	Minimal	Insufficient		/20
Application (APP)	**Stability and Correctness** To what degree does the video game behave correctly? To what extent is run-time error handling implemented (to prevent crashes)?	Very High	High	Moderate	Minimal	Insufficient		/40
	Avoidance of Code Duplication To what degree has the student used subs, functions, arrays, control arrays and other constructs to avoid duplication of code?	Very High	High	Moderate	Minimal	Insufficient		
	Debugging To what extent has the student employed a logical, thorough and organized debugging method?	Very High	High	Moderate	Minimal	Insufficient		
	Completeness of Implementation To what degree has the student completed the implementation of the software design?	Very High	High	Moderate	Minimal	Insufficient		
Thinking, Inquiry and Problem Solving (TIPS)	**Level of Difficulty** How challenging was the development of the chosen video game software? How challenging was it for the student to select, design and implement the algorithms? To what extent was it necessary to include advanced programming concepts learned in ICS3M0?	Very High	High	Moderate	Minimal	Insufficient		/30
	Research When problems are encountered during the design, implementation and validation phases, to what degree has the student consulted available resources ***before*** asking for help?	Very High	High	Moderate	Minimal	Insufficient		
	Algorithm/Implementation Efficiency To what level does the algorithm use resources (memory, processor time, etc) efficiently? (This includes choosing appropriate data types for variables.)	Very High	High	Moderate	Minimal	Insufficient		
Communication (COM)	**Indentation of Code** **Insertion of Blank Lines in Strategic Places** (to make code easier to read)	Very Few or no Errors	A Few Minor Errors	Moderate Number of Errors	Large Number of Errors	Very Large Number of Errors		/50
	Comments (Internal Documentation) • Effectiveness of explaining abstruse (difficult-to-understand) code • Effectiveness of introducing major blocks of code • Avoidance of comments for self-explanatory code	Very High	High	Moderate	Minimal	Insufficient		
	Descriptiveness of Identifier Names Variables, Constants, Objects, Functions, Subs, etc **Sub and Function Design** Subs and functions are self-contained (can be used in other programs without modification); parameters and, return types are logical **Inclusion of Property Names with Object Names** (e.g. 'txtName.Text' instead of 'txtName' alone) **Clarity of Code** How easy is it to understand, modify and debug the code? **Adherence to Naming Conventions** (e.g. use "txt" for text boxes, "lbl" for labels, etc.)	Masterful	Good	Adequate	Passable	Insufficient		
	User Interface To what degree is the user interface • well designed? • logical? • attractive? • user-friendly?	Very High	High	Moderate	Minimal	Insufficient		
	User Help Documentation (If Applicable) To what degree has the student designed "help" documents that are • logically organized? • easy to understand and relevant? • grammatically correct (including spelling)?	Very High	High	Moderate	Minimal	Insufficient		