ICS2O – Programming Assignment 2

Topic: Everything…

Marking:

We will be looking for a number of things for this assignment. **Remember that just because your program works does not mean you will get full marks; this is actually only a portion of the final mark for a question.**

- **Style**
  - Documentation (Header, variables, internal comments, flow charts)
  - Structure (Indentation, white space, code location)
  - Decisions (Variable Naming, documentation descriptions, data types)
- **Correctness** (Your program produces the expected results)

Academic Integrity:

Plagiarism and other acts of academic dishonesty will be dealt with harshly in accordance with the school's rules. Talking about the questions is allowed, this is called collaboration. **Copying/Using/Giving code from/to friends or other sources is considered cheating.** To ensure you do not cross this line; when talking about a question with someone do so away from a computer with a paper and pen in hand and write out IDEAS, not code. Then when you are finished talking return to your computer and implement the ***ideas*** you have come up with.

Follow the instructions below to get started…

- Create a folder on your USB Key named **Assignments**, inside that folder create another folder named **<lastname><firstInitial>_PASS2**. E.g. **SquarepantsB_PASS2**

Time Management:

This program will focus your knowledge in an assignment in which you will need to use everything you have learned this semester. There is only one question, but there is no time to spare. This assignment will challenge you. Are you up for the challenge?

This assignment is designed to force you to use the fundamentals of everything we have done in the class thus far. It will serve to get you ready to be a productive unit as part of a team on your final assignment. Be sure to pay close attention to the use of subprograms, loops, arrays and flow charts.

Hangman

Task:

Your job will be to create the game hangman. The rules are simple; the computer will randomly choose a phrase that has an associated category. There are four things that are always visible to the guesser:

- The category for the chosen phrase
- The alphabet (all guessed letters are not visible or shown to be guessed in some way)
- The phrase is presented to the user as a series of underscores and spaces (if needed) where each underscore represents a letter from the phrase.
- A wooden platform with a wooden post with a horizontal wooden beam that has a hangman's rope (Neuse) hanging from it. At the beginning of the game the Neuse is empty

The rest of the game relies on the skills of deduction of the guesser. Each turn the guesser has two choices, they can attempt to solve the puzzle or guess another letter.

If they try to solve the puzzle but are incorrect they are punished. If they guess correct the round is over and they are awarded points and the next round will begin with a new phrase and category if required.

If they guess a letter and it does not exist or one that has already been guessed they are punished. If they guess a new existing letter then each letter in the phrase that matches that letter is revealed. The turn then starts again with the two choices.

NOTE: To simplify things, use all upper case letters in phrases that are all the same length. Choose a maximum number of characters you want in your phrases and stick with it for every single one. You may wish to use extra spaces at the end of a phrase that is shorter. Talk to your teacher if you are unsure.

The game lasts 3 rounds, and the guesser's score for each round is added up to get their final score.

Punishment:

This is the game called hangman; it is called this because if you are punished enough times you will be "hung". Each time the player is punished another body part is added to the Neuse. There are a total of 6 body parts in this order:

- Head
- Torso
- Left arm
- Right arm
- Left leg
- Right leg.

If the guesser has all body parts added they are hung and they lose that round with no points.

Scoring:

Scoring is pretty simple, the guesser scores 10 points for each body part not added. That means they can win a maximum of 60 points in the round.

Phrases:

Your game must have a string array of 20 phrases. It will also have a string array of 20 category descriptions. These arrays should work together. What this means is that when a random phrase is chosen, say for example you randomly choose the number 14, the current phrase will be the element 14 from the phrase array and the category will be element 14 from the category array. You may decide your own phrases and categories, but you must have between 3 and 5 different categories, no more no less. See the simple example below: (Do not use these phrases)

	Phrase Array		Category Array
1	Despicable Me	1	Movie Titles
2	Super Mario Bros	2	Video Games
3	Arnold Schwarzenegger	3	Famous People
4	Ready Player One	4	Book Titles
5	Call Me Maybe	5	Song Titles

Flowchart:

Before beginning programming you must complete a flowchart in Microsoft Word or Draw.io and submit it before programming may begin. Your flowchart must demonstrate the flow of **ONE** complete Round of play by the guesser.

ENHANCEMENTS (Want to earn that level 4+ … this is the only way):

An enhancement is an improvement in the program that improves the overall experience. This can include many things; the creativity, quantity, quality and complexity of these enhancements is how you will be awarded marks. Below is a list of possible enhancements, do not feel limited by this list. If you have other ideas, USE THEM! Do you have to do all of these enhancements to get perfect? Not necessarily, the quality of the enhancement itself can sometimes be enough.

- Multiplayer (Instead of the computer choosing the phrase, have another player entering a phrase and category)
- Fully functional Top 10 score board (Array's will help here)
- Animations
- sounds
- Ensure no phrase is ever chosen more than once in a single game

NOTES:

Subprograms, Arrays & Constants

You program needs to make good appropriate use of subprograms you create as well as event subprograms. To do this remember subprograms should be used for both readability and reusability purposes. Any collections of related data should use an array. It is expected that there will be no magic numbers with the common and small +1 exceptions.

BE VERY CAREFUL OF PLAGIARISM IN THIS PORTION OF THE ASSIGNMENT, YES YOU MAY RESEARCH ON THE INTERNET AND LEARN HOW THINGS ARE DONE, BUT DO NOT SIMPLY CUT-AND-PASTE CODE FROM THE INTERNET. THIS IS CHEATING AND WILL BE DEALT WITH ACCORDINGLY.

Programming Assignment 2: Hangman

The following rubric acts as a checklist, after grading each box will be in one of the three states shown above. These states will be used to determine your level for that achievement category.

NAME:

Achievement Category	Mark /10
Knowledge & Understanding Structure a. Indentation b. White Space c. Code Location d. Variable Blocking e. Logic Blocking Subprograms and arrays effectively done a. Definitions b. Calling/Accessing c. Procedure vs. Function	
Thinking & Inquiry Decisions a. Naming b. Documentation Descriptions c. Data Types d. Appropriate use of subprograms and arrays	

Achievement Category	Mark /10
Communication Documentation a. Headers b. Variables c. Blocks d. Subprograms	
Application Software Correctness a. Game is fully playable as expected b. All required data is displayed as necessary c. Control works as expected (Letters/Guessing) d. Drawings completed successfully e. Punishments are in the game with consequences f. Letters are generated properly g. UI is aesthetically pleasing h. End-of-game works as described in all scenarios	

Level 1 (50 – 59%)	Level 2 (60 – 69%)	Level 3 (70 – 79%)	Level 4 (80 – 100%)
- shows little understanding of … - rarely adheres to …	- shows some understanding of … - mostly adheres to …	- shows understanding of … - adheres to …	- shows considerable understanding of … - completely adheres to … - goes above and beyond in showing …