==================================================================

Student Name：彭敬樺 Assignment：2

Student ID：0416106

Student Email: wilbert.phen@gmail.com

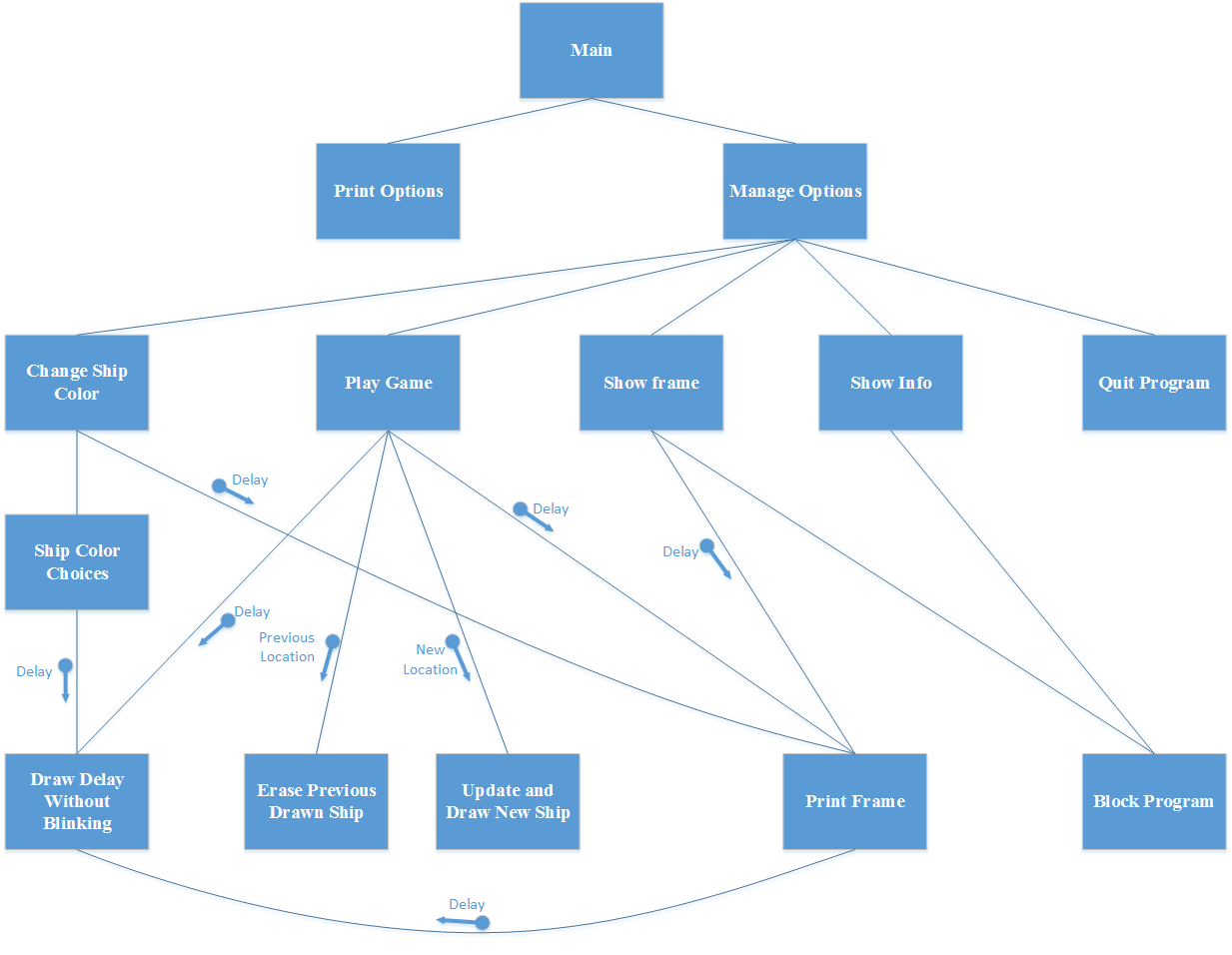
**Claim: I worked on my own. ( YES / NO )**

**Introduction**

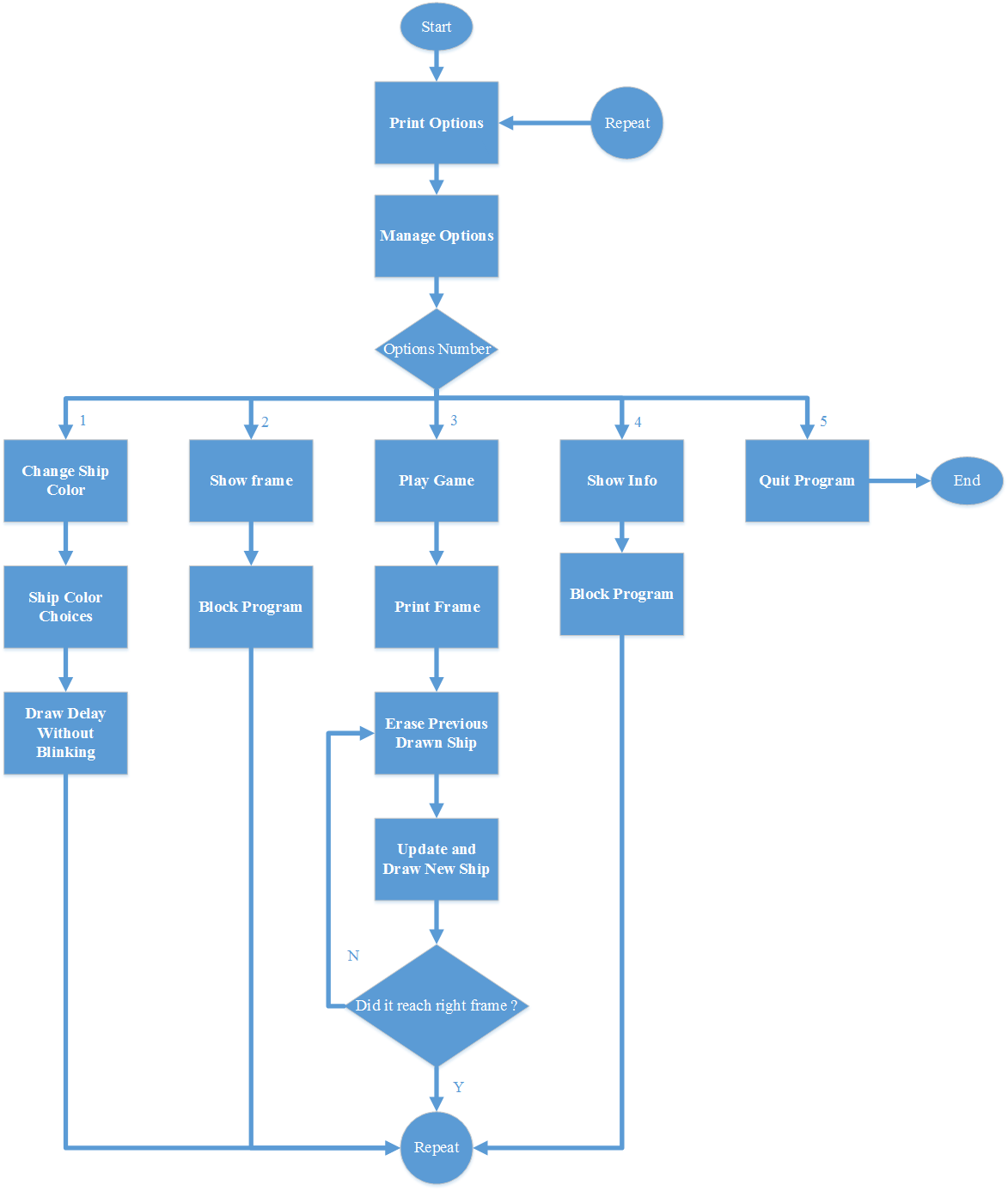
Word Count: 142 words

Assignment 2 consists of program to play a simple spaceship game. There will not be any obstacle or complicated structure and knowledge to know beforehand. Every instruction is given explicitly and try to avoid ambiguity. In the beginning, there are 5 options to be chosen to continue throughout the program.

The ship’s color can be changed through the first option. The second choices will provide viewer the process of frame printing onto the console interface. To play the game, the third alternative is the one to choose. Next, information about the author of this program can be easily accessed by the fourth number. The fifth choice provides way to quit the program properly because there will be infinite looping for the options to be chosen. This program’s purpose is to ensure understanding towards more detailed command and consequences of each instruction involved.

**Structure Chart**

**Flowchart**

****

**System Architecture**

Word Count: 146 words

This program consists of 13 functions: 1 main functions and 12 supporting functions. Options will be laid out repeatedly as long as the “quit” option is not selected. This program runs by calling the appropriate function that will have the required job done. This function is segregated from the other for some reasoning. The first reason to split the function into several one is to increase readability, which allows reader to understand it faster. The second reason is due to specification and specialization requirement. Some parts of the code could be done repeatedly if this is not implemented. The isolated function can also do some error checking before adjusting crucial data.

Before any functions started, this program has included two library to assist the process: Irvine32.inc and Macros.inc. Many instruction used in the code will be declared and deeply implemented in this parts of the code.

**The Approach**

Word Count: 355 words

Writing a simple game is not as easy as what the name stated. Using assembly language, program could become more complicated and entangled. To prevent and reduce abnormality in the result, the code is separated into smaller function that will start running from the main function.

Due to the number of function, some constant expression are declared on top of the data and code. This writing style will enable the programmer to edit their program more easily as it is as chained towards the value of the constant. Some important value such as starting coordinate, default color, or default delay are included in this category.

Next, some data is declared to assist some variable exchanging and calculation in each function. In this particular code, only one data is used: ship\_color. This variable is used throughout the entire project. Therefore, declaring it as global data will decrease time consumption from data eviction.

Some function such as “block\_program”, “draw\_delay\_without\_blink”, and “print\_frame” is often called by variety of function. Because of its ubiquitous existence, some value of register used by the function will first be saved into the stack to prevent data loss or modification by such trivial function. By restoring their former value before exiting the function, this function will not be destructive towards others.

There are one function which will alter the wanted result without us recognizing it. This function in my program is called “print\_frame”. If not carefully implemented, this function will change the color of text due to the drawing of frame using some different color. Saving and restoring register’s value will not be any help here. The only way is to first recover current text color and applied it back before the function exiting.

In this particular implementation, every input will be read by the “readkey” function that will not block the program although no input is received. This deemed to be useful when the ship have to keep moving even though no key is pressed. In some occasion, the input buffer must be flushed beforehand to prevent the “readkey” function to read from existing buffer, which could be inputted long before read.

**Discussion/ Experiments**

Word Count: 282 words

After completing the entire program writing, I as the author start to experiment with the program. The first and foremost is to ensure the looping process could run successfully and ensure that all data is re-initialized into its correct starting value.

Next thing is to check whether the entire parts of the program has the same frame color before any different color has been implemented. In this parts, I encounter some characteristics of the function that I have used for a long time but have not been able to realize it. One thing to be realizes is that “clrscr” is not literally clear screen. The actual action done is as if the entire screen is printed with space character. This condition meant that the current background text color could affect the result of “clrscr”.

Label that is followed by only 1 colon is a local label, however the one followed by 2 colon is the global label. The difference between this two is that global label can be accesed from anywhere in the program, while local one cannot. This is quite useful in the implementation where the function which decides whether the program has to quit or not is not the main function of the program. The other function will need to have access to the last last to exit the program from the main. The disadvantages of this is that we are jumping between function without using the usual process. This is quite dangerous if the label we jump to is not ending the program. The value in the stack after jumping between function could be not as expected. That is why that this method is advised not to be used.

**Conclusion**

Word Count: 196 words

This assignment provides a good tunnel to understanding each instruction in a more detailed fashion. By experiencing it directly, we can determine what is this function actually done to our program. Some function could have some unexpected result or could have change some register value that is very important for the flow of program.

Stack is one of the most interesting and a good way to pair your program with. This capability enables us to store and grab the wanted value in a more controlled way. Although stack could be seem as an endless memories. There should be noted that stack memories is actually limited. The reason we still think it as though it is infinite is that because our program size is still in a tiny scale compared to the one other professional programmers created.

The thing I learnt from this homework is quite much. The one worth mentioning is that we should have better understanding towards what the function that we call will do. Will the function alter the register? This should be first understood. In my experience, many value broke just because that value is in register an being altered by another function.