1. The following exercise is to familiarise yourself with PCSpim. Note: If you use QtSpim, your exercise here will be based on QtSpim.  
     
   a) Make a hand drawing of the PCSpim user interface [0.5 marks]  
   Note: Your hand drawing doesn’t need to copy all the information on the user interface; just pick up a couple of lines from each area (pane).  
     
   b) Explain in writing what is the role of four main horizontal areas (panes) in the PCSpim user interface [0.5marks]

The **text segment** contains the machine language code for instructions. These are usually stored in its text segment.

The **data segment** contains a binary representation of initialized data in the source file (arrays, strings and other data types). The data also may be incomplete because of unresolved references to labels in other files. This data is stored in memory.

-Floating Register – holds floating point information in register

-Int Register - Holds register for integer values

1. **Question (0.3):** Answer the question in writing. List a couple of lines of code statements from PCSpim. Point out which are the *original text* format of the instructions, the *assembled* version, and the *machine code* version. Briefly explain main differences in between.

***Assembly language is the symbolic representation of a computer’s binary encoding—machine language. Assembly language is more readable than machine language because it uses symbols instead of bits. Original text may have pseudo-instructions; this type of instruction may not be actually supported by the MIPs architecture. You will not see these in the assembled version, and this might cause some differences between the original text and the assembled version. The assembled instruction are easier to interpret; for the user(primitive format). The machine code version is in binary or hexadecimal (machine readable).***

***Memory Machine AssemblyCode*** ***Instruction***

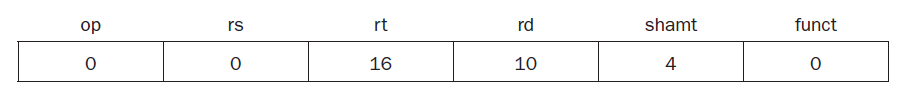


1. **Question (0.2):** What is the effect of clearing registers? Why simulator provides this operation?

***Clearing registers, clears the contents and addresses which are located in them. Clearing the registers allow for errors to be reduced, and allows the programmer using the software to have a clear understanding of what is going on in the program that they are testing. The computer could be pointed to the wrong address and incorrectly calculate things if the registers are not cleared.***

1. **Questions (0.2):** Observe what’s happening when running the program. What was the effect from this code modification? Explain why.

***You need three operands for the sub operation to work; this has only two operands, and therefore a syntax error occurs. You need rs, rt, and rd.***



1. **Questions (0.3):** When running *helloimproved.*s, after the output text is printed to the console, why the cursor is placed in the next (blank) line? What is the role of instruction ‘syscall’? How syscall *services* and *arguments* are selected and specified in the code?

* ***Because of “/n”gives it a new line before it executes the command.***

***SPIM provides a small set of operating-system-like services through the MIPS system call (syscall) instruction. To request a service, a program loads the system call code (see Table below) into register $v0 and the arguments into registers $a0, ..., $a3 (or $f12 for floating point values). A value is loaded immediately into $v0, with the value of 4 (which specifies that it wants to print a string); then the string that it is going to print, its address (label1) is loaded into $a0. The service that it provides is dependent on the value that is storied in $v0, and the arguments address is loaded in to $a0, and whatever is in the argument is printed.***

