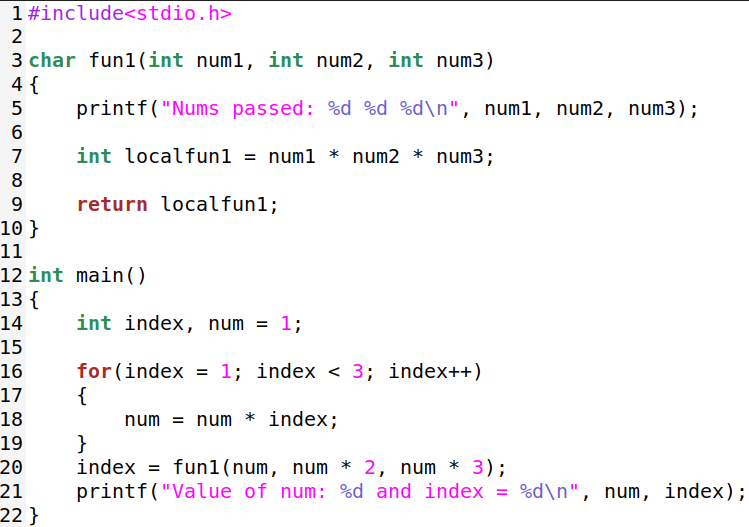
1. Create assembly code that perform the following:
   1. Create a data section with two dd values 0x70000000 label n1 and 0x90000006 label n2
   2. Store n1 in eax register and n2 in edx register
   3. Add the two values. The result should be in eax register
   4. In a loop subtracts 4 from eax (Register that contains addition result)
   5. If the result of the subtraction is Not Zero keep on looping
   6. If result is Zero use the **XOR** instruction to make edx contain the value 0
   7. Use system call to exit the program
   8. ( 5 marks) Attach screen capture with assembly code
   9. ( 2 marks) Attach screen capture that demo the respective programs you used to generate the executable
2. Use gdb to debug the program and the respective commands to verify every instruction and registers values. Provide screen captures that demo the following:
   1. ( 1 mark) Registers values after addition
   2. ( 3 marks) Registers values when eflag register changes. If flag changed, provide the flag(s) name
   3. ( 1 mark) Registers values after subtraction (within the loop)
   4. ( 1 mark) Registers values after executing xor instruction
   5. ( 3 marks) The opcode of the loop in Intel format

Type and compile the following C code



1. **In order to see the symbols remember to compile the program with the appropriate switch/option.**
2. **Commands that may be helpful: info frame, break, info register, info locals, x/16x $esp**
3. ( 3 marks) Provide a screen captures of C code and results after compilation
4. Use the debugger to analyze the stack. Use the respective debugger commands to:
   * ( 4 marks ) disassembly the function called **fun1**  identify the prologue and epilogue
   * ( 5 marks) Analyze the stack  when calling function **fun1**. Provide screen capture to demo by highlighting the following:
     + function arguments
     + rsp, rbp, old rbp ( saved frame pointer)
     + ret addresses for this function **fun1**