Practical 09

Assembly Language

THIS IS A PROCTORED PRACTICAL

YOU MUST SHARE YOUR SCREEN SO YOUR PARTICIPATION IN THIS PRACTICAL CAN FULLY INVIGILATED

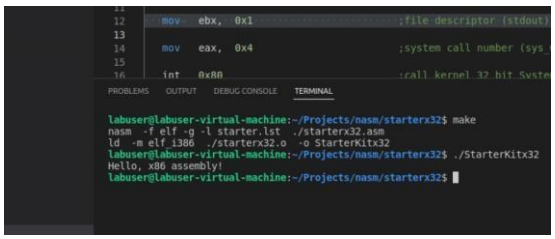
1. Create a Github repository "Assembly_and_C"
2. Create a sub directory PRACTICAL_##
3. Add Github link to CA Spreadsheet
   e.g https://STUDENTID.github.com/Assembly_and_c/PRACTICAL_##
4. Invite Lab Supervisors including **MuddyGames** as a collaborators
5. Go to designated group to complete practical
6. Upload completed Practical files to Github repository

Create a unique file ***e.g. practical_##_part#.c*** or ***practical_##_part#.asm*** for each practical section below.

Clone https://bitbucket.org/MuddyGames/assembly-and-c-x86_64/src/master/

Linux VM https://comp-vcentre.itcarlow.ie

Objective Understand and utilise x84 Assembly Instructions

1	Create a new directory and name ***practical_09_part1.*** This is a clone of **starterx32** directory Program, edit compile and execute code to perform activities =>	 1. Open terminal in Visual Studio Code 2. Perform a make 3. Run binary produced 4. Modify output string so that it outputs "Assembly and C" 5. Note registers used
2	Create a new directory and name ***practical_09_part2.*** This is a clone of **starterx64** directory Program, edit compile and execute code to perform activities =>	1. Open terminal in Visual Studio Code 2. Perform a make 3. Run binary produced 4. Modify output string so that it outputs "Assembly and C" 5. Note registers used

3	Create a new directory and	

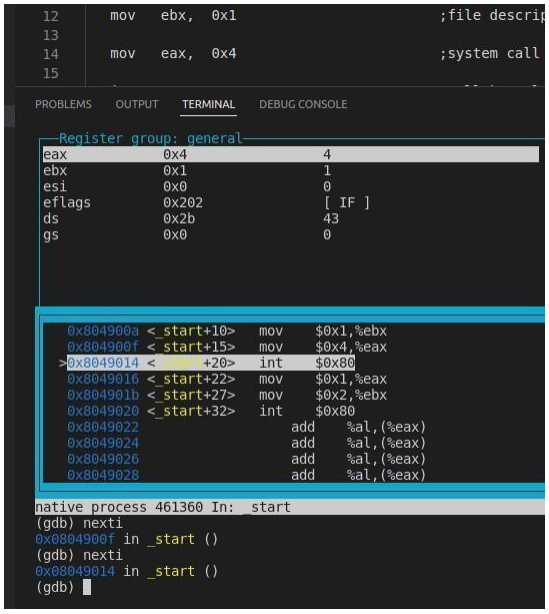
name ***practical_09_part3.***

This is a clone of **starterx32** directory

Debug binary produced in ***practical_08_part1***

Program, edit compile and execute code to perform activities =>

For full list of [GDB Commands]



1. Open terminal and issue following command

   `gdb StarterKitx32 -tui`

2. Set a breakpoint

   `break _start`

3. Run binary

   `run`

4. Step through using

   `nexti`

   AND

   `next`

   commands, observe difference

5. Rerun StarterKitx32 using

   `run` command

6. Examine register values using

   `layout reg` command

7. Examine register values using e.g.

		`info registers eax` **`i r eax`** **`info all-registers`** commands

		8. Take screenshots of each step and add to ***practical_09_part3*** directory

4	Create a new directory and name ***practical_09_part4.***
	
	This is a clone of **integrationx32** directory
	
	Program, edit compile and execute code to perform activities =>
		integrationx32 > add.asm
		`global add`
		
		`section .data`
		`section .text`
		`add:`
		`  mov  eax, [esp+4]  ; argument 1`
		`  add  eax, [esp+8]  ; argument 2`
		`  ret`
		
		1. Modify main.c and add.asm so that the method takes 3 arguments
		
		`extern int add(int a,int b,int c);`
		
		2. Compile run as check validity of code
		
		3. Create and new assembly file which subtracts one number from another
		
		`extern int sub(int a, int b);`
		
		4. Modify make file to include the new assembly file sub.asm
		
		5. Compile run as check validity of code

5	Complete Practical Quiz which will be provided by Lab Supervisor	

Demonstrate completed assembly files at the end of the LAB and ensure it has been checked

Student Name	**Brandon Jaroszczak**	**Student Number**	**C00296052**
Date	**31/3/2025**	**Checked**	