

## Practical 09

### Assembly Language

THIS IS A PROCTORED PRACTICAL

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

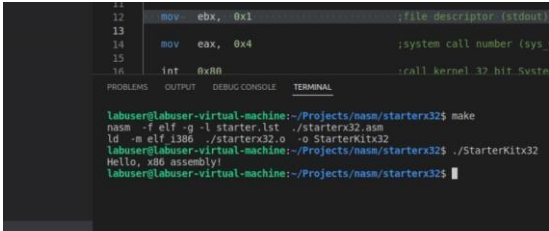
1. Create a Github repository "Assembly\_and\_C"
2. Create a sub directory PRACTICAL\_##
3. Add Github link to CA Spreadsheet  
e.g [https://STUDENTID.github.com/Assembly\\_and\\_c/PRACTICAL\\_##](https://STUDENTID.github.com/Assembly_and_c/PRACTICAL_##)
4. Invite Lab Supervisors including **MuddyGames** 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/](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

<p><b>1</b></p>	<p>Create a new directory and name <b>practical_09_part1</b>.</p> <p>This is a clone of <b>starterx32</b> directory</p> <p>Program, edit compile and execute code to perform activities =&gt;</p>	 <ol style="list-style-type: none"> <li>1. Open terminal in Visual Studio Code</li> <li>2. Perform a make</li> <li>3. Run binary produced</li> <li>4. Modify output string so that it outputs "Assembly and C"</li> <li>5. Note registers used</li> </ol>
<p><b>2</b></p>	<p>Create a new directory and name <b>practical_09_part2</b>.</p> <p>This is a clone of <b>starterx64</b> directory</p> <p>Program, edit compile and execute code to perform activities =&gt;</p>	<ol style="list-style-type: none"> <li>1. Open terminal in Visual Studio Code</li> <li>2. Perform a make</li> <li>3. Run binary produced</li> <li>4. Modify output string so that it outputs "Assembly and C"</li> <li>5. Note registers used</li> </ol>
<p><b>3</b></p>	<p>Create a new directory and</p>	

## Practical 09

### Assembly Language

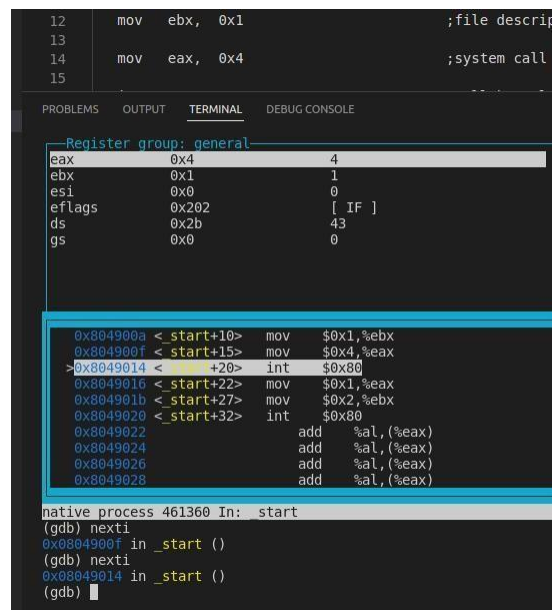
name ***practical\_09\_part3.***

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](#)



The screenshot shows a GDB terminal window with the following content:

```
12      mov     ebx, 0x1                ;file descrip
13
14      mov     eax, 0x4                ;system call
15
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
--Register group: general--
eax      0x4      4
ebx      0x1      1
esi      0x0      0
eflags   0x202    [ IF ]
ds       0x2b     43
gs       0x0      0

0x0804900a <_start+10> mov     $0x1,%ebx
0x0804900f <_start+15> mov     $0x4,%eax
>0x08049014 <_start+20> int     $0x80
0x08049016 <_start+22> mov     $0x1,%eax
0x0804901b <_start+27> mov     $0x2,%ebx
0x08049020 <_start+32> int     $0x80
0x08049022                add     %al,(%eax)
0x08049024                add     %al,(%eax)
0x08049026                add     %al,(%eax)
0x08049028                add     %al,(%eax)

native process 461360 In: _start
(gdb) nexti
0x0804900f in _start ()
(gdb) nexti
0x08049014 in _start ()
(gdb) |
```

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.

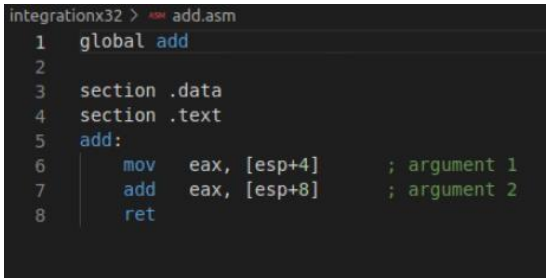
## Practical 09

### Assembly Language

		<pre>info registers eax i r eax info all-registers commands</pre>
--	--	---

## Practical 09

### Assembly Language

		8. Take screenshots of each step and add to <b><i>practical_09_part3</i></b> directory
4	<p>Create a new directory and name <b><i>practical_09_part4</i></b>.</p> <p>This is a clone of <b>integrationx32</b> directory</p> <p>Program, edit compile and execute code to perform activities =&gt;</p>	 <pre> integrationx32 &gt; .\add.asm 1  global add 2 3  section .data 4  section .text 5  add: 6      mov     eax, [esp+4]      ; argument 1 7      add     eax, [esp+8]      ; argument 2 8      ret </pre> <ol style="list-style-type: none"> <li>1. Modify main.c and add.asm so that the method takes 3 arguments</li> </ol> <pre>extern int add(int a,int b,int c);</pre> <ol style="list-style-type: none"> <li>2. Compile run as check validity of code</li> <li>3. Create and new assembly file which subtracts one number from another</li> </ol> <pre>extern int sub(int a, int b);</pre> <ol style="list-style-type: none"> <li>4. Modify make file to include the new assembly file sub.asm</li> <li>5. Compile run as check validity of code</li> </ol>
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	