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_#)

1. Invite Lab Supervisors including **MuddyGames** as a collaborators
2. Go to designated group to complete practical
3. Upload completed Practical files to Github repository

NOTE: Use of EASy68K editor and emulator allowed, use of internet allowed, use of slide deck(s) allowed. Installer located here <http://www.easy68k.com/>

Create a unique file ***e.g. practical\_##\_part#.X68*** for each practical section below.

**Objective** Understand and utilise Conditional Branches and Control Structures**:**

|  |  |  |
| --- | --- | --- |
| **1** | Create a new 68K project and name the file ***practical\_06\_part1.X68***    Edit compile and execute the code across and observe while debugging and contents of Data and Address Registers. | [**Source Code Image (click here)**](https://1drv.ms/i/s!Au3XD_Li32ZenpATXokfoxgKGK96eA?e=aZsjC0) |
| **2** | Create a new 68K project and name the file ***practical\_06\_part2.X68***    Complete code for [Trap Tasks as listed here.](https://1drv.ms/x/s!Au3XD_Li32ZenpAUeN5_huWAKHKzgA?e=fg2pgn) | [**Trap Codes (click here)**](https://1drv.ms/x/s!Au3XD_Li32ZenpAUeN5_huWAKHKzgA?e=fg2pgn) |
| **3** | Create a new 68K project and name the file ***practical\_06\_part3.X68***    Edit compile and execute the code across and observe while debugging and contents of memory, data registers and address registers.    Review questions, what is the purpose of Address Register A7?  A7 is the Stack Pointer which keeps track of the top stack. It is there for the push and pop feature of a stack | [**Source Code Image (click here)**](https://1drv.ms/i/s!Au3XD_Li32ZenpAs71_nGigwe3RRvQ?e=xLCcKt) |
| **4** | Create a new 68K project and name the file ***practical\_06\_part4.X68***    Edit compile and execute the code across and observe while debugging and contents of memory, data registers and address registers.    Review questions, what is the purpose of Address Register SP (Stack Pointer)?    Unlike A7 SP register is designed specifically to manage the stack AND point to the top of the stack (track the top) | [**Source Code Image (click here)**](https://1drv.ms/i/s!Au3XD_Li32ZenpAth-Ic5mMDzg_C4w?e=btC7VT) |

|  |  |  |
| --- | --- | --- |
| **5** | Create a new 68K project and designate the file as ***practical\_06\_part5.X68***.    Review questions parameters to subroutines can be passed through the stack what other functions can be achieved through the stack, what observations have you made when opening VIEW | STACK ?  Most of the stack has unused memory? (FF FF FF FF) | [**Source Code Image (click here)**](https://1drv.ms/i/s!Au3XD_Li32ZenpECnvnekT5bSV6_Hw?e=FPCOAs) |

|  |  |  |
| --- | --- | --- |
| **6** | Create a new 68K project and designate the file as ***practical\_06\_part6.X68***.    Review questions moving around stack can be achieved by what means other than push and pop, what problems could this cause?    Direct addressing with offset or not, but that will cause stack misalignment if done incorrectly. The data can and will be 100% corrupted | [**Source Code Image (click here)**](https://1drv.ms/i/s!Au3XD_Li32ZenpEDym_n5UP-oEgAaQ?e=ib9054) |
| **7** | 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** | **Ariel Fajimiyo** | **Student Number** | **C00300811** |
| **Date** | **25/02/25** | **Checked** |  |