Practical 06

Assembly Language

THIS IS A PROCTORED PRACTICAL

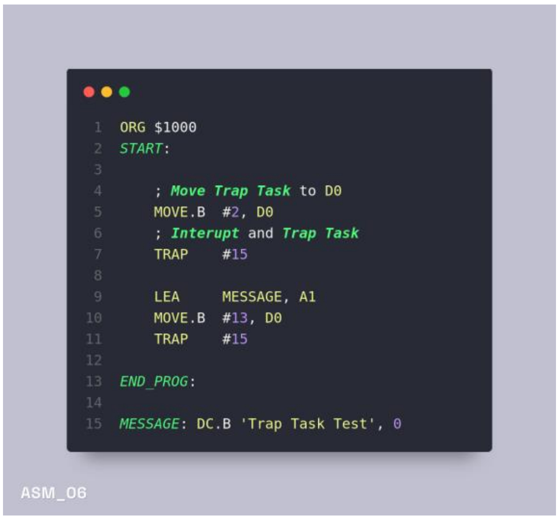
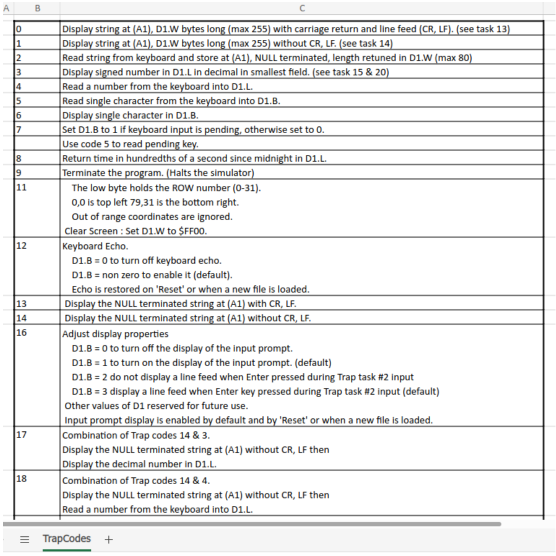
YOU MUST SHARE YOUR SCREEN SO YOUR PARTICIPATION IN THIS PRACTICAL CAN FULLY INVIGILATED

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

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/

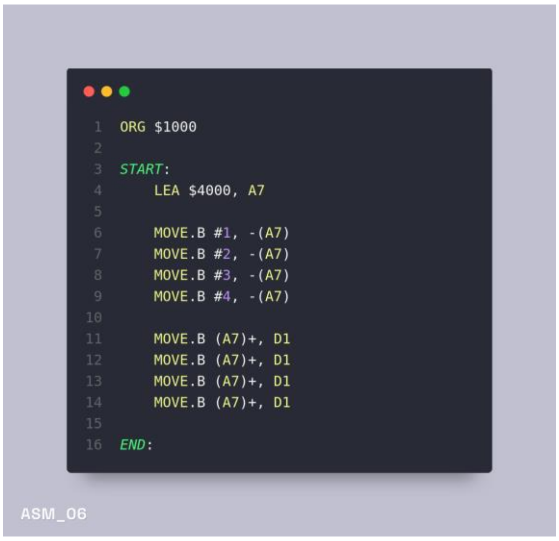
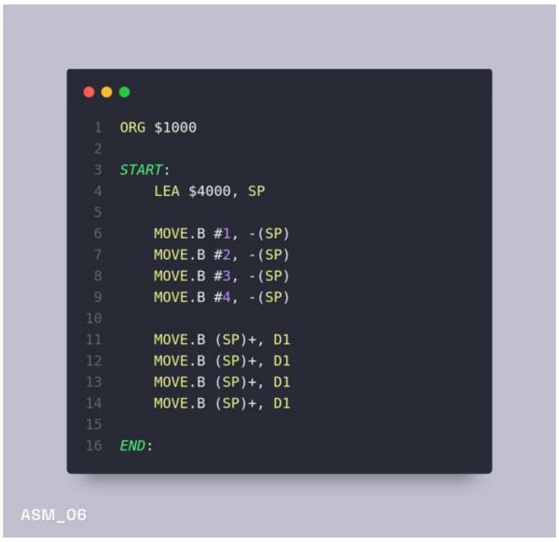
Create a unique file ***e.g. practical_##_part#.X68*** for each practical section below.

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)
2	Create a new 68K project and name the file ***practical_06_part2.X68*** Complete code for Trap Tasks as listed here.	 Trap Codes (click here)

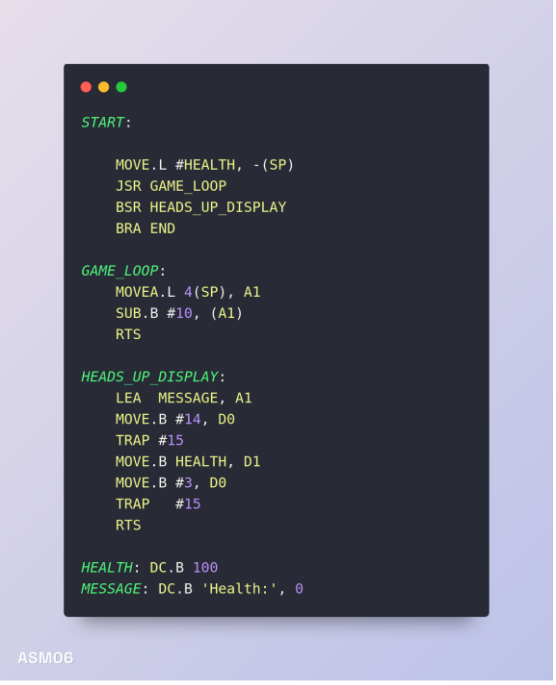
```
ORG $1000
START:

    ; Move Trap Task to D0
    MOVE.B  #2, D0
    ; Interupt and Trap Task
    TRAP    #15

    LEA     MESSAGE, A1
    MOVE.B  #13, D0
    TRAP    #15

END_PROG:

MESSAGE: DC.B 'Trap Task Test', 0
```

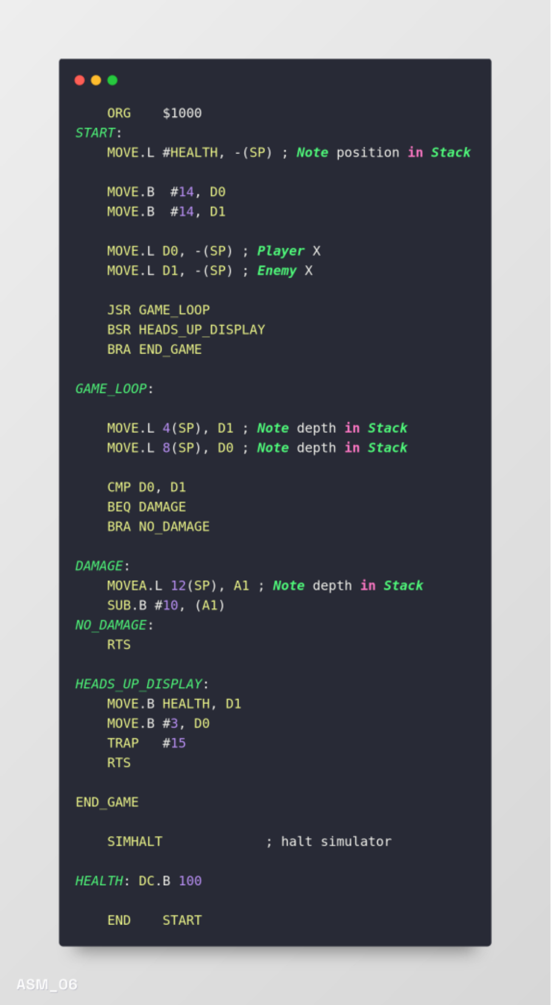
B	C
0	Display string at (A1), D1.W bytes long (max 255) with carriage return and line feed (CR, LF). (see task 13)
1	Display string at (A1), D1.W bytes long (max 255) without CR, LF. (see task 14)
2	Read string from keyboard and store at (A1), NULL terminated, length retuned in D1.W (max 80)
3	Display signed number in D1.L in decimal in smallest field. (see task 15 & 20)
4	Read a number from the keyboard into D1.L.
5	Read single character from the keyboard into D1.B.
6	Display single character in D1.B.
7	Set D1.B to 1 if keyboard input is pending, otherwise set to 0.
	Use code 5 to read pending key.
8	Return time in hundredths of a second since midnight in D1.L.
9	Terminate the program. (Halts the simulator)
11	The low byte holds the ROW number (0-31). 0,0 is top left 79,31 is the bottom right. Out of range coordinates are ignored. Clear Screen : Set D1.W to $FF00.
12	Keyboard Echo. D1.B = 0 to turn off keyboard echo. D1.B = non zero to enable it (default). Echo is restored on 'Reset' or when a new file is loaded.
13	Display the NULL terminated string at (A1) with CR, LF.
14	Display the NULL terminated string at (A1) without CR, LF.
16	Adjust display properties D1.B = 0 to turn off the display of the input prompt. D1.B = 1 to turn on the display of the input prompt. (default) D1.B = 2 do not display a line feed when Enter pressed during Trap task #2 input D1.B = 3 display a line feed when Enter key pressed during Trap task #2 input (default) Other values of D1 reserved for future use. Input prompt display is enabled by default and by 'Reset' or when a new file is loaded.
17	Combination of Trap codes 14 & 3. Display the NULL terminated string at (A1) without CR, LF then Display the decimal number in D1.L.
18	Combination of Trap codes 14 & 4. Display the NULL terminated string at (A1) without CR, LF then Read a number from the keyboard into D1.L.

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?	 ORG $1000 START: LEA $4000, A7 MOVE.B #1, -(A7) MOVE.B #2, -(A7) MOVE.B #3, -(A7) MOVE.B #4, -(A7) MOVE.B (A7)+, D1 MOVE.B (A7)+, D1 MOVE.B (A7)+, D1 MOVE.B (A7)+, D1 END: Source Code Image (click here)
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)?	 ORG $1000 START: LEA $4000, SP MOVE.B #1, -(SP) MOVE.B #2, -(SP) MOVE.B #3, -(SP) MOVE.B #4, -(SP) MOVE.B (SP)+, D1 MOVE.B (SP)+, D1 MOVE.B (SP)+, D1 MOVE.B (SP)+, D1 END: Source Code Image (click here)
5	Create a new 68K project and designate the file as	

Practical 06

Assembly Language

	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 ?
		
		START:
		    MOVE.L #HEALTH, -(SP)
		    JSR GAME_LOOP
		    BSR HEADS_UP_DISPLAY
		    BRA END
		GAME_LOOP:
		    MOVEA.L 4(SP), A1
		    SUB.B #10, (A1)
		    RTS
		HEADS_UP_DISPLAY:
		    LEA MESSAGE, A1
		    MOVE.B #14, D0
		    TRAP #15
		    MOVE.B HEALTH, D1
		    MOVE.B #3, D0
		    TRAP #15
		    RTS
		HEALTH: DC.B 100
		MESSAGE: DC.B 'Health:', 0
		Source Code Image (click here)

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?
		
		    ORG $1000
		START:
		    MOVE.L #HEALTH, -(SP) ; Note position in Stack
		    MOVE.B #14, D0
		    MOVE.B #14, D1
		    MOVE.L D0, -(SP) ; Player X
		    MOVE.L D1, -(SP) ; Enemy X
		    JSR GAME_LOOP
		    BSR HEADS_UP_DISPLAY
		    BRA END_GAME
		GAME_LOOP:
		    MOVE.L 4(SP), D1 ; Note depth in Stack
		    MOVE.L 8(SP), D0 ; Note depth in Stack
		    CMP D0, D1
		    BEQ DAMAGE
		    BRA NO_DAMAGE
		DAMAGE:
		    MOVEA.L 12(SP), A1 ; Note depth in Stack
		    SUB.B #10, (A1)
		NO_DAMAGE:
		    RTS
		HEADS_UP_DISPLAY:
		    MOVE.B HEALTH, D1
		    MOVE.B #3, D0
		    TRAP #15
		    RTS
		END_GAME
		    SIMHALT ; halt simulator
		HEALTH: DC.B 100
		    END START
		Source Code Image (click here)

7	Complete Practical Quiz which will be provided by Lab Supervisor	

Practical 06

Assembly Language

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

Student Name		**Student Number**	
Date		**Checked**	