NAME : MIGUEL CARLOS V. GONZALEZ **DATE : _____**
SUBJ/SEC : COMPUTER ARCHITECTURE AND ORGANIZATION / BSCOE 4-6 **PROFESSOR : Engr. Rolito L. Mahaguay, *MSE-CpE, PCpE***

LABORATORY EXPERIMENT #2
INTRODUCTION TO TURBO ASSEMBLER

OBJECTIVE (S):

1. to be able to learn the basics of turbo assembler.
2. To properly assemble and link a program using tasm and tlink.
3. To be able to create an assembly language program using turbo assembler with SK as editor.

REQUIREMENTS:

1 personal computer
1 system disk
1 data disk
- Assembler – **TASM.EXE**
- Loader - **TLINK.EXE**
- Editor - **SK**

DISCUSSION:

Writing a program in Assembly Language follows the same procedures as in high-level languages such as Pascal.
1. Type the program instructions into the computer using a text editor, then save the program on the disk.
2. Translate the text file into a machine language program using an assembler. If the assembler finds errors, correct them with the text editor and reassemble the program.
3. Convert the assembler output to an executable ***run module*** using the loader.
4. Execute the program.
5. Check the results. If they differ from what you expected, you must find the error or ***bugs***, that is, you must ***debug*** the program.

Text_Editor

A text editor is a program that allows you to enter and prepare your program from the ordinary keyboard into a computer-readable form. It also allows you to save this file on the disk for later use. The assembler and loader program requires the inputs to be saved files from the disk.

A text editor can be any popular work processor or edit program that can produce pure ***ASCII*** text.

As a standard ;

Assembler

A program written in assembly language is translated to machine code by an assembler. Assemblers and the corresponding assembly language mnemonics are generally limited to use with one particular microprocessor, which limits their portability, or use on other machines. Today's assemblers do much more than translate assembly language mnemonics into binary code.

Loader

Before the microprocessor can execute any machine instruction, it must first be loaded into the memory accessible to it. The loader is the program that actually takes the machine instructions (object code) and places it in memory at the specified starting address for execution. Loaders range from very simple to very complex.

PROCEDURES:

1. Write the following program using EDIT as your text editor. Assign the filename your ***nickname.asm*** (Take note that filename sholud be a maximum of 8 characters only)

```
                .model small
                .code
                org 100h

start           mov ah,2
                mov dl, 'R'
                int 21h
                mov dl, 'E'
                int 21h
                mov dl, 'D'
                int 21h
                int 20h
end start
```

2. Save the program by pressing F2.

3. Go to the DOS prompt and then assemble the program by using the command below:

   C:\>tasm nickname ***ex: C:\>tasm tolitz***

```
C:\TASM>tasm nickname
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   nickname.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  476k
```

4. If there are warnings and errors in the assembling process, go back to the source code and fix those errors. If no error occurs proceed to the next step.

5. Link the program by using the command :

   C:\>tlink/t nickname ***ex: C:\>tlink/t tolitz***

   There should be no BAD OBJECT FILE message in the linking process if a *bad object file occurs* assemble the program once again.

```
C:\TASM>tlink/t nickname
Turbo Link  Version 2.0  Copyright (c) 1987, 1988 Borland International
```

6. Execute the program by simply typing the filename on the prompt. What is the output of the given program?

```
C:\TASM>nickname
RED
```

7. Modify exer2.asm. The output should be:

   R
   E
   D

8. Write down the modified program in the space below.

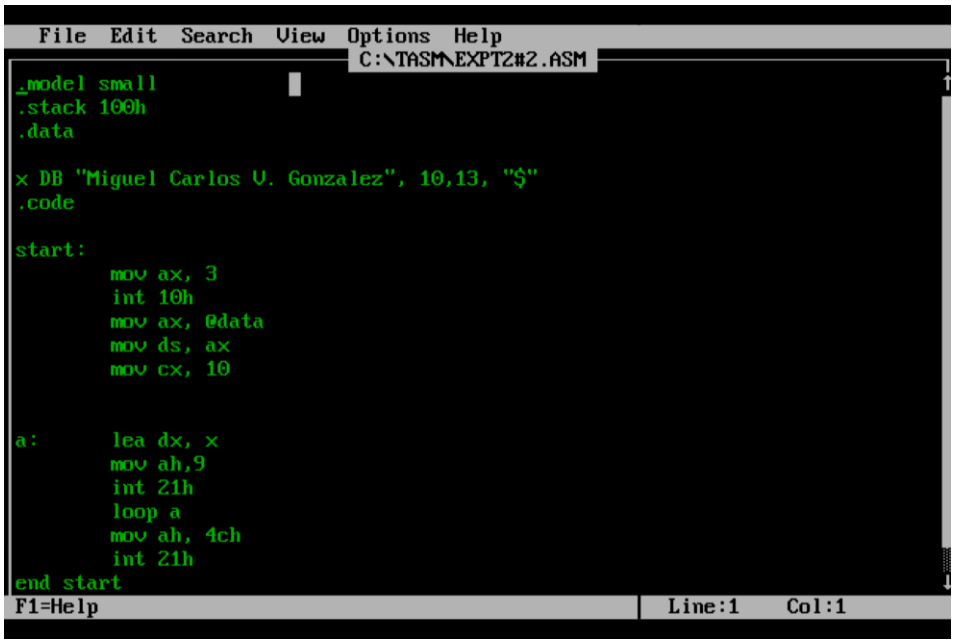
```
                .model small
                .code
                org 100h

start:
                mov ah,2
                mov dl, 'R'
                int 21h
                mov dl, 10
                int 21h
                mov dl, 'E'
                int 21h
                mov dl, 10
                int 21h
                mov dl, 'D'
                int 21h
                mov dl, 10
                int 21h
                int 20h
end start
```

9. Copy the program to your data disk.

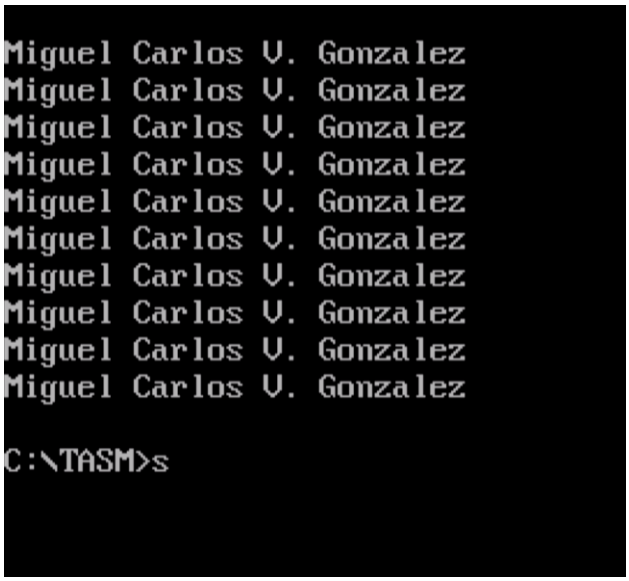
EXERCISES:

1. Write down the required program output from your instructor/professor.

2. Encode the program.



3. Ask your instructor to check your work, then copy it to your data disk.



4. How many files are created using the turbo assembler? **2 FILES**

5. What are those files?

```
C:\TASM>DIR EXPT2#2
Directory of C:\TASM\.
EXPT2#2  ASM                323 16-11-2023  1:24
EXPT2#2  OBJ                284 16-11-2023  1:29
    2 File(s)               607 Bytes.
    0 Dir(s)        262,111,744 Bytes free.
```

6. Which files are created in the assembling process?

- The OBJ file was created during the assembling process.

7. Which files are created in the linking process?

```
C:\TASM>DIR EXPT2#2
Directory of C:\TASM\.
EXPT2#2  ASM                323 16-11-2023  1:24
EXPT2#2  EXE                566 16-11-2023  1:30
EXPT2#2  MAP                232 16-11-2023  1:30
EXPT2#2  OBJ                284 16-11-2023  1:29
    4 File(s)             1,405 Bytes.
    0 Dir(s)        262,111,744 Bytes free.
```

8. What is the purpose of /t in the linking process?

- Without adding /t in the linking process Turbo Link will not be able to compile or link the program due to the written program not having a defined stack.

9. Compare DEBUG and TURBO ASSEMBLER use in assembly language programming.

- DEBUG is a debugging tool for inspecting and manipulating assembly programs, but has limited capabilities. TASM, on the other hand, is a full-featured assembly language assembler used for writing, assembling, and generating executable programs from assembly source code.

10. What is the purpose of org 100h in your program?

- The org 100h in assembly language sets the program's starting address to 0x100 in memory.

SUMMARY & CONCLUSION:

In this experiment, I have learned how to initially write a loop in assembly, by using the loop function to repeat my name, a number of times. Aside from that I have also learned what happens when I use TASM and TLINK to compile the programs I write using the EDIT functionality. Additionally, using special characters, I have learned how to modify the output of a certain program