COMP 3050 Computer Architecture
Homework #3 February 5, 2026

- This assignment is due on **Thursday, February 19.**

- **All** of your submissions must include a minimum of **four** separate files:
- **File 1:** A short **write-up** that **first** specifies what you think your **degree of success** with a project is (**from 0% to 100%**), followed by a brief discussion of your approach to the project along with a **detailed description** of any problems that you were **not** able to resolve for this project. **Failure to specifically provide this information will result in a 0 grade** on your assignment. If you do **not disclose** problems in your write-up and problems are detected when your program is tested, you will receive a grade of 0. **Make sure that you include your email address in your write-up so that the corrector can email you your grade.**
- **File(s) 2(a, b, c, …)**: For this assignment you need only one file which will include your **complete source code**, in the form of an **.asm** file as discussed in class.
- **File 3**: A simple text file that includes the **commands necessary** to build and run your code (the **masm** command, and the appropriate **mic1** command).
- **File 4:** A file that includes your **resulting output** run(s) from your project. This is a simple text file that shows your output, but make sure that you **annotate it** so that it is self descriptive and that all detailed output is well identified. For mic1 executions, use the **script** command from the shell to capture all screen output to a file.

- The files described above should be the only files placed in one of your subdirectories, and this subdirectory should be the target of your submit command (see the on-line file **Assignment_Submit_Details.pdf** for specific directions).

- This assignment will require you to write a program in Mic1 macro-assembly language to calculate **Fibonacci numbers**. This program will require that you build a function to calculate the Fibonacci number for the value of the argument N using the following definition of Fibonacci numbers:

```
Fibonacci(0) = 0

Fibonacci(1) = 1

Fibonacci(N) = Fibonacci(N-1) +  Fibonacci(N-2)

The sequence of numbers looks like:
N          0 1   2   3   4    5   6   7
Fib(N)     0 1   1   2   3    5   8   13
```

- The function you will build must compute the required Fibonacci number using a **recursive implementation**, and should present a signature of:

```
int Fib (int argument)
```

- Your program should set up **data locations** for the following values to be used as **arguments to Fib:**

  3
  9
  18
  23
  25

- Your program must also set up five **corresponding locations** for the Fibonacci numbers of the argument values shown above.

- Your main routine should iteratively call your Fib function, passing each argument value (on the stack) and storing each return value (in the AC) into the location reserved for it. You can use the **halt** command between each call to your subroutine and from the debugger interface, print out the return value that you've stored from the previous call during debugging, but you probably will want only a single halt at the end of the program to show your results once you have the program running correctly.

- Remember, you must indicate **clearly** on your short write-up, how much of this assignment you feel you were able to do correctly, and what, if anything, you were not able to do. **Failure to specifically provide this information will result in a 0 grade** on your assignment. If you do not disclose problems in your write-up and problems are detected when your program is tested, you will receive a grade of 0.

Assuming that your recursive **Fib** function is called **5 times** (once with each of the above arguments) and assuming that the **5 arguments and their corresponding Fibonacci numbers** are stored in memory from locations **100 – 109** respectively, your results from the debugger should look like this:

EXPECTED OUTPUT SHOWN BELOW FROM A FINAL HALT

```
-bash-3.00$ ./masm < rfib.asm >rfib.obj
-bash-3.00$ ./mic1 prom.dat rfib.obj 0 2048

Read in 81 micro instructions
Read in 116 machine instructions
Starting PC is : 0000000000000000  base 10:       0
Starting SP is : 0000100000000000  base 10:    2048


ProgramCounter : 0000000000100100  base 10:      36
Accumulator    : 0000000000000000  base 10:       0
InstructionReg : 1111111100000000  base 10:   65280
TempInstr      : 1000000000000000  base 10:   32768
StackPointer   : 0000100000000000  base 10:    2048
ARegister      : 0000000000000001  base 10:       1
BRegister      : 0000000000000000  base 10:       0
CRegister      : 0000000000000000  base 10:       0
DRegister      : 0000000000000000  base 10:       0
ERegister      : 0000000000000000  base 10:       0
FRegister      : 0000000000000000  base 10:       0

Total cycles   : 35109454

Type decimal address to view memory,  q  to quit or  c  to continue: 99
     the location   99 has value 1111111111111111 , or 65535  or signed     -1
Type  <Enter>  to continue debugging
Type        q  to quit
Type        f for forward range
Type        b for backward range: f
Type the number of forward locations to dump: 10
     the location  100 has value 0000000000000011 , or     3  or signed      3
     the location  101 has value 0000000000001001 , or     9  or signed      9
     the location  102 has value 0000000000010010 , or    18  or signed     18
     the location  103 has value 0000000000010111 , or    23  or signed     23
     the location  104 has value 0000000000011001 , or    25  or signed     25
     the location  105 has value 0000000000000010 , or     2  or signed      2
     the location  106 has value 0000000000100010 , or    34  or signed     34
     the location  107 has value 0000101000011000 , or  2584  or signed   2584
     the location  108 has value 0110111111110001 , or 28657  or signed  28657
     the location  109 has value 0010010100010001 , or  9489  or signed   9489
```

The arguments are shown in locations 100 – 104 in RED, the function results are shown in locations 105 – 109 in PURPLE.