1. (35 points) Perform the following calculations, assuming a 24-bit word size. Indicate the presence or absence of overflow. Assume base 16 for all parts.

a) (5 points) Assume unsigned addition: OV No OV (circle one)

\[
\begin{array}{c}
3 \ 4 \ 8 \ 2 \ 6 \ 3 \\
+ \ 7 \ 3 \ C \ 6 \ 9 \ 8
\end{array}
\]

b) (10 points) Assume unsigned subtraction: OV No OV (circle one)

\[
\begin{array}{c}
9 \ 8 \ E \ A \ 7 \ F \\
- \ 8 \ 9 \ F \ F \ 6 \ 3
\end{array}
\]

c) (5 points) Assume two's complement addition: OV No OV (circle one)

\[
\begin{array}{c}
3 \ 9 \ 6 \ F \ E \ 2 \\
+ \ 5 \ 3 \ 8 \ A \ A \ A
\end{array}
\]

d) (15 points) Assume two's complement subtraction: OV No OV (circle one)

\[
\begin{array}{c}
A \ 6 \ 8 \ 9 \ B \ E \\
- \ B \ 8 \ F \ D \ E \ 3
\end{array}
\]

2. Floating Point Data Representation (20 points)
Provide the hexadecimal value for the CUSP floating point representation for the decimal \(2347.1875\). Showing your work facilitates partial credit. CUSP has an 8-1-15 representation with a 128 exponent bias.

3. CPU Design (20 points) Explain why CISC was the preferred instruction set format in the early years of computers, and what changes occurred that made RISC the preferred format now. (This can be done in two sentences!)

4 (30 points) Fill in the values for the following registers after the program halts. Also circle the state of each flag after the program halts. Assume the initial PC is set to $100, ACCUM = 0, and XR = 0.

Opcodes: LDA = 00, LDX = 01, STA = 04, ADA = 10, CMA = 20, JMP = 40, JLT = 4A
ASCII: space = $20, ‘ = $27, A = $41, I = $49, M = $4D, a = $61, d = $64, m = $6D

```
.BLKW  4, $100020
ADA # PromptLen
HLT
.EQU @, $0FF
LDX # 53
LDA $105
CMA $106
JLT $001
JMP $002
.CHAR 'Madam I'm Adam', PromptLen
.WORD $101
```

ACCUM _________  PC _________  XR __________

LT: SET CLEAR  OV: SET CLEAR  EQ: SET CLEAR
5. (46 points) CUSP Programming. Write a CUSP program that repeatedly prompts the user for a number between 1 and 100 until the user enters a zero, and then prints out the average of the numbers. The program should start execution at location PC location $20$. MINI_OS: PUT_NUM $E00$; GET_NUM $E01$; PUT_STR $E05$ Remember to allow for MINI_OS.OBJ. PROCEDURE PUT_STR(NumChar: INTEGER; VAR PutStr: CHAR_STRING);

Here is a sample run of the program.

Please enter a number (1-100): 30
Please enter a number (1-100): 45
Please enter a number (1-100): 63
Please enter a number (1-100): 0
46