Name: ________________________
Last, First

Pawprint ID: ________________________

Show all work on this exam paper. **No** notes or calculators. You may use the Appendix of your book!

Tally Sheet:

Problem	Max Score	Your Score
1	20	
2	25	
3	30	
4	25	
Total	100	

1(A.) (10 points) Suppose we have the following instructions.

```
LDAA        #$FF
CMPA        #$7F
BGT         AHEAD
```

Will the program take the branch? Give a Reason ________NO____________________________

BGT -> signed numbers -> $FF = -1, $7F = 127; -1 < 127-> branch NOT taken

1(B.) (10 points) Suppose we have the following instructions.

```
LDAA        # $FF
CMPA        # $7F
BHI         AHEAD
```

Will the program take the branch? Give a Reason _______________YES____________________

BHI-> unsigned numbers -> $FF = 255; $7F = 127; 255 > 127 -> branch TAKEN

For problem 1, please give a convincing reason. For example, just stating ‘the branch will be taken because the number is greater’ is insufficient. Rather, come up with the decimal value of the numbers to show that a number is greater than the other. A correct branch prediction with incorrect, missing, or insufficient reason will result in 0 points.

2. (25 points) For each of the following questions, assume that the registers are initialized to the following values (i.e., each part is independent of the others). Note: ML stands for memory location. Give the contents of the registers/memory locations after each instruction is fetched and executed. All numbers shown are HEX numbers except when noted otherwise. DO NOT leave any box empty (even if the contents don't change) - empty boxes will be counted wrong! Also, do not use a DASH "-" in a box and do not forget the $-sign for hex numbers.

A. SUBA #$FE (5 points)

	PC	A	X	Y	ML($5000)	ML($5001)	ML($5002)
Initial	$C000	$FF	$4FF0	$5000	$80	$40	$20
After	$C002	$01	$4FF0	$5000	$80	$40	$20

B. ADDA 1,Y (5 points)

	PC	A	X	Y	ML($5000)	ML($5001)	ML($5002)
Initial	$C000	$FF	$4FFF	$5000	$80	$40	$20
After	$C003	$3F	$4FFF	$5000	$80	$40	$20

C. STAA 3,X (5 points)

	PC	A	X	Y	ML($5000)	ML($5001)	ML($5002)
Initial	$C000	$FF	$4FFF	$5000	$80	$40	$20
After	$C002	$FF	$4FFF	$5000	$80	$40	$FF

D. STX 1,X (5 points)

	PC	A	X	Y	ML($5000)	ML($5001)	ML($5002)
Initial	$C000	$FF	$4FFF	$5000	$80	$40	$20
After	$C002	$FF	$4FFF	$5000	$4F	$FF	$20

E. LDY 2,X (5 points)

	PC	A	X	Y	ML($5000)	ML($5001)	ML($5002)
Initial	$C000	$FF	$4FFF	$5000	$80	$40	$20
After	$C003	$FF	$4FFF	$4020	$80	$40	$20

3. (30 points) Write a complete assembly program to implement the following pseudo code IN THE TABLE PROVIDED ON THIS PAGE. Note: START, COUNT and RESULT are **signed** numbers/variables. Please use meaningful labels such as IF, THEN, ELSE, ENDIF, DO, UNTIL, etc. You**r** program should match the pseudo code 1-to-1 using the correct implementation of the if-then-else and do-until structures.

```
COUNT = START;
DO {
    IF ( RESULT >= 20 ) RESULT=0;
    ELSE  RESULT ++;
   COUNT++;
} UNTIL (COUNT > 5)
```

You may assume the following declarations:

```
           ORG     $ B000
START      EQU     -5
COUNT      RMB     1
RESULT     FCB     10
```

Labels	Operation/Instruction	Operands
	ORG	$C000
	LDAA	#START
	STAA	COUNT
DO		
IF	LDAA	RESULT
	CMPA	#20
	BLT	ELSE
THEN	CLR	RESULT
	BRA	ENDIF
ELSE	INC	RESULT
ENDIF	INC	COUNT
UNTIL	LDAA	COUNT
	CMPA	#5
	BLE	DO

4. (25 points) Consider the following assembly language program:

```
            ORG         $B000
DATA1       FCB         97, 100, 98, 120, 32, 41, 42, 0

            ORG         $B100
DATA2       FCB         120, 32, 41, 42, 0


*start of your program

            ORG         $C000
            LDX         #DATA1
LABEL       LDAA        0,X
            BEQ         LABEL1
            INX
            BRA         LABEL
LABEL1      LDY         #DATA2
LABEL2      LDAA        0,Y
            BEQ         LABEL3
            STAA        0,X
            INX
            INY
            BRA         LABEL2
LABEL3      CLR         0,X
            STOP
```

Come up with the pseudo code for this program (**please include the data section**). The pseudo code should match the program 1-to-1 (only use if-then, if-then-else, while or do-until structures). **Note: use pointers in your pseudocode. Assume that DATA1 and DATA2 store unsigned numbers.**

```
unsigned int DATA1[];
unsigned int DATA2[];
unsigned int *POINTX, *POINTY;

        POINTX=&DATA1[0];
        WHILE( *POINTX != 0) POINTX++;

        POINTY=&DATA2[0];
        WHILE( *POINTY != 0){
                *POINTX = *POINTY;
                POINTY++;
                POINTX++;
        }
        *POINTX=0;
```