# Exercise 2 Problems

1. Convert the following assembly code fragments into machine code using the most efficient addressing mode available (i.e. if possible, use direct instead of extended). Also, this problem intentionally uses some instructions that have not shown up in lectures yet, since their operation has no effect on the problem.
   1. LDAB $2000

ADDB #10

STAB $2000

* 1. LDY #$3000

LDD 0,Y

* 1. LDAA $20

LDAB #20

ABA

STAA $21

* 1. LDAA $2500

CMPA $2501

BRA $50

* 1. LDD $40

SUBD #1

BRA -100

1. Write the assembly code and the machine code to load the specified register with the given address using indexed addressing off of X. Assume that register X already contains $32F0, and use the most efficient offset (i.e. don’t use a 9 bits if a 5 bits can be used).
   1. A, $32F5
   2. A, $32E0
   3. B, $32D0
   4. B, $3255
   5. X, $3200
   6. D, $32AA
   7. D, $3353
   8. Y, $3361
   9. S, $3300
   10. S, $3232
2. The arrays below represent bytes of machine code. Beginning with the first byte, convert one machine code instruction into assembly code. Note that arrays may have extra bytes, and the extra bytes should be ignored. It also intentionally uses some instructions that have not been discussed in lecture yet, since their operation has no bearing on this problem.
   1. 86 45 96 11
   2. 20 21 22 23
   3. 72 19 00 26
   4. AB E8 CC 97
   5. 83 20 00 00
   6. 63 00 01 02
   7. 09 90 04 40
   8. 02 27 A0 FF
   9. 18 27 10 00
   10. 18 06 20 45
3. Calculate the answers to the following hexadecimal problems. All numbers are unsigned. Note that one assumes the answer must fit in the same number of digits as the longest number in the problem.
   1. $25 + $25
   2. $37 + $48
   3. $A2 + $25
   4. $7E + $55
   5. $C080 + $0080
   6. $DFFF + $1045
   7. $C140 + $82
   8. $B0FE + $26
   9. $3456 + $FF
   10. $40 – $25
   11. $57 – $48
   12. $A2 – $25
   13. $7E – $55
   14. $C080 – $0081
   15. $DFFF – $1045
   16. $C140 – $82
   17. $B0FE – $26
   18. $3456 – $FF
4. Calculate the answers to the following hexadecimal problems. All numbers are signed. Assume the answer is limited to 4 hexadecimal digits.
   1. $D024 + $88
   2. $8176 + $79
   3. $2D9D + $CD
   4. $C080 + $80
   5. $DFFF + $45
   6. $C140 + $82
   7. $B0FE + $26
   8. $3456 + $FF
5. Calculate the branch destination using the supplied offset if the program counter given is pointing to the given address AFTER the instruction has been fetched.
   1. PC=$2027, offset= $79
   2. PC=$21DC, offset= $88
   3. PC=$3F7B, offset= $3F
   4. PC=$1EFF, offset=$AC
   5. PC=$3011, offset=$FD78
   6. PC=$12DD, offset=$13BA
6. Repeat Problem 6 assuming that the value of the program counter points to the beginning of a BRA or LBRA, and that the more efficient branch is used.
7. Use the BRA or LBRA instructions to write a line of assembly code that begins in memory location $2450 to branch to the address given using the more efficient branch (i.e. don’t use LBRA if BRA will reach). Hint: Note that the offset is calculated from the next memory address after the branch instruction. A BRA is 2 bytes long, occupying addresses $2450 and $2451, and the offsets are calculated from $2452. The LBRA instruction is 4 bytes long, and the offsets are calculated from $2454.
   1. $2460
   2. $2440
   3. $2503
   4. $2403
   5. $3000
   6. $1000
   7. $24AE
   8. $23F7
   9. $3585
   10. $1234