Name:_____

20
12
2

Problem 1 (10 points)

What is the output of this Linux command executed in the Bash shell?

```
for i in `seq 20 -2 0`; do echo "$i" ; done | grep 2 | tail -n 2
```

Problem 2 (10 points)

What is the output of this Linux command executed in the Bash shell?

```
for i in 1 2 3 ; do
for j in 1 2 3 ; do
echo -n "$i "
done
echo ""
done
```

1 1 1
2 2 2
3 3 3

Name:_____

Problem 3 (10 points)

What is 0x1111 in binary?

0001 0001 0001 0001

Convert 65 from decimal to binary

1 0 0 0 0 0 0 1

Convert this 8-bit value, 1000 0001, to its 2's complement decimal equivalent.

$-128+1 = -127$

Perform the following addition and report the result in hexadecimal:

0xAABB + 0x7733

$$\begin{array}{r} {\scriptstyle 1}\\ AABB\\ 7733\\ \hline 121EE \end{array}$$

Problem 4 (10 points)

Reconstruct the definitions of A, B, C, D given the bytes stored in memory. Memory addresses increase in the up direction. The first byte of A in memory contains 0x78.

```
          section .data
A         dw   0x5678
B         db   0x34
C         dw   0x4312
D         db   42
E         dd   0x00102241
```

	0x99
	0x00
	0x10
	0x22
	0x41
	0x42
	0x43
	0x12
	0x34
	0x56
A →	0x78

Name:_____

Problem 5 (10 points)

What is left in **eax**, **ebx**, **ecx**, and **edx** after these instructions have executed.

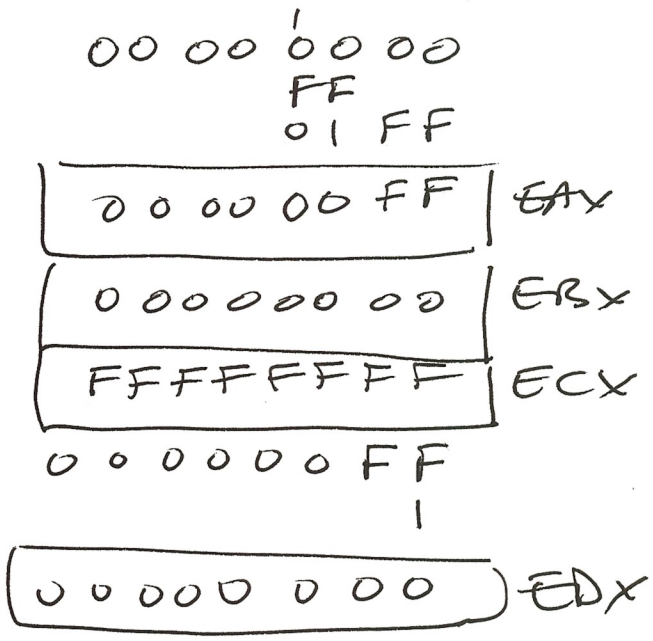
```
mov     eax, 0
mov     ah, 0xff
add     ax, 0x1ff

mov     ebx, -1
add     ebx,  1

mov     ecx, 0
dec     ecx

mov     edx, 255
add     dl, 1
```

eax: _____

ebx: _____

ecx: _____

edx: _____



Problem 6 (10 points)

What is left in **eax**, **ebx**, **ecx**, and **edx** after these instructions have executed.

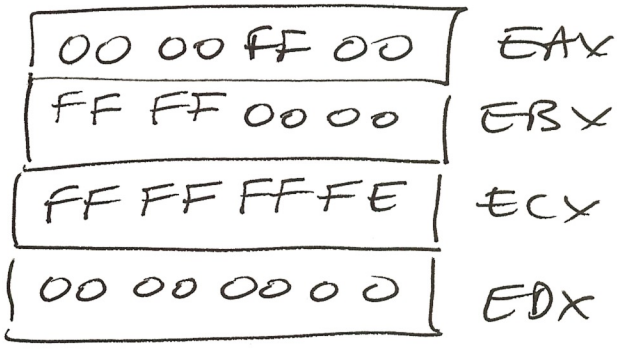
```
mov     eax, 0
or      ah, 0xff

mov     ebx, 0xffffffff
and     bx,0

mov     ecx, 0x1
xor     ecx, 0xffffffff

xor     edx, edx
```

eax: _____

ebx: _____

ecx: _____

edx: _____



Name:_____

Problem 7 (10 points)

Study this Java program:

```
public class Looping {

    public static void main( String[] args ) {
        int x = 1;
        for ( int i=0; i<2000000000; i++ ) {
            System.out.println( x );
            x = x * 2;

            if ( x== 0 ) break;           // case 1
            if ( x== -1 ) break;          // case 2
            if ( x== -2 ) break;          // case 3
            if ( x== 0x80000000 ) break; // case 4
            if ( x== 0x7fffffff ) break; // case 5

        }
    }
}
```

Question 1: How many values of x will it print? Check the correct answer:

- ☐ none
- ☒ approximately 30
- ☐ approximately 2,000,000,000
- ☐ approximately 4,000,000,000
- ☐ it's an endless loop: it will print values forever

Question 2: Which of the break statements is executed? Check the appropriate box.

- ☐ case 1
- ☐ case 2
- ☐ case 3
- ☒ case 4
- ☐ case 5
- ☐ None; it's an endless loop
- ☐ None; the loop never runs because its upper bound is negative

Name:_____

Problem 8 (10 points)

Given the following declarations:

```
        section .data
a       dw      1
b       dw      2
c       dw      3
sum     dw      0
```

Write the assembly language instructions that are needed to compute

sum = (a+b)*2 – c

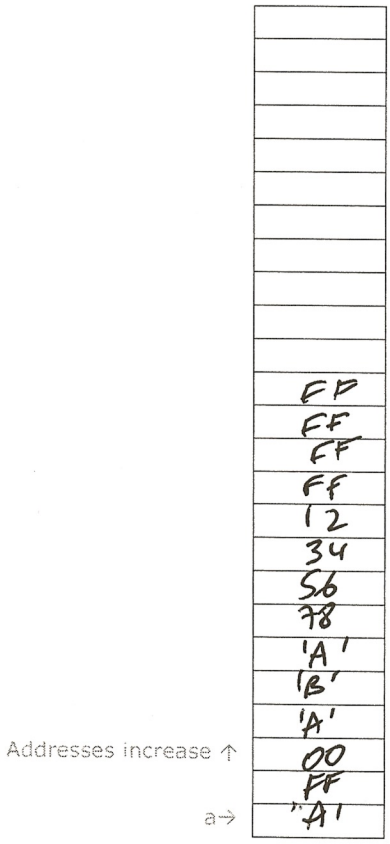
```
mov     ax, word [a]
add     ax, word [b]
add     ax, ax          ; 2(a+b)
sub     ax, word [c]
mov     word [sum], ax
```

Name:_____

Problem 9 (10 points)

Show the contents of the memory with whatever format is appropriate once the variables shown below have been loaded into it. (The Ascii code for 'A' is 0x41)

```
            section     .data
a           db          'A'
b           dw          0xff
c           db          'ABA'
d           dd          0x12345678
e           dd          -1
```



Name:_____

Problem 10 (10 points)

This question refers to byte integers.

What are the smallest and largest *unsigned* integers that can be stored in 8 bits?

smallest: 0 largest: $2^8 - 1 = 255$

What are the smallest (most negative) and largest (most positive) *signed* integers that can be stored in 8 bits?

smallest: $-2^{8-1} = -128$ largest: $2^{8-1} - 1 = 127$

Problem 11 (10 points)

What is one major difference between compiled and interpreted languages when it comes to integer arithmetic?

integer arithmetic with compiled languages can overflow and program will not indicate errors.
Interpreted languages can change the storage allocated to variables and make it grow as needed.