Question 1. The bigger the better. (18, 3 pts each)

1. What is the largest number that can be represented by a 7 bit floating point number (say with the same rules as IEEE 754 floating point), with a 1 bit sign, 3 bit exponent, and 3 bit significand (bias=3)?$1.875 * 2^3 = 15$
2. In C, what's the largest `int` plus one? -2^{31}
3. Consider an n-bit signed number, what's the largest one? $2^{(n-1)} - 1$
4. In C, what's the smallest `unsigned int` minus one? $2^{32} - 1$
5. Which can represent the largest number in C, the largest `float` or the largest `signed long` or largest `unisgned int`? lagest float
6. Which integer type in C is large enough to store a pointer without loss of precision?uint64_t, long,..

Question 2. Matchmaker (8 Pts, 1 pts each)

Pretend to be a compiler.

You are free to assign registers to variables however you choose. Assume x and y are of type int. Remember, the compiler(me) may have done some optimizations.

(g)	y=x+y	(a)	`shl $ 5 %edi`
(a)	x=x*32	(b)	`xorl %edi %edi`
(f)	x=x*5+3	(c)	`shr $ 31 %edi`
(c)	x=(x < 0) ? -1 : 0	(d)	`movl $1 %eax`
(d)	x=1	(e)	`imul %edi %edx`
(h)	x=x*3+5	(f)	`leaq 3(%edi,%edi,4)`
(b)	x=0	(g)	`addl %edi %edi`
(e)	x=x*y	(h)	`leaq 5(%edi,%edi,2)`

3.1

65

6c697665

3.2

Multiples of four

4.1

24

240

4.2 This is one way to draw it:

a	-	-	-	b	b	b	b	c	-	-	-	-	-	-	-	d	d	d	d	d	d	d	d

4.3

movb $0,248(%rdi)

4.4

rearrange the memory values in the struct (size depends on how you rearrange)

Question 5. I can puzzle, (15 Pts, 2 pts each)

Answer these true false puzzles. Assume the following setup:

```
int x = foo();
int y = bar();
unsigned ux = x;
unsigned uy = y;
```

True `-x == ~x+1`

False `x >> 2 == x / 4`

False `x > 0 && y > 0 ⟹ x + y > 0`

False `5*ux > ux`

False `x < 100 ⟹ 10*ux > ux`

7.1 this answer assumes fib(4) was called

Return addr of caller	
Old rpb	fib(4)
Old rbx	
<stack space>	
0x400572	
Old rbp	fib(3)
Old rbx (4)	
<stack space>	
0x400572	
Old rbp	fib(2)
Old rbx (3)	
<stack space>	<-%rsp

7.2 2 (0,1 acceptable to have as well)
7.3 40 (0,1 acceptable to have as well)
7.4 0,1
7.5 44, 45
7.6 13, 23
7.7 35 and 31 (i allowed either)

8. Insertion Sort (http://quiz.geeksforgeeks.org/insertion-sort/)

```
void insertionSort(int arr[], int n)
{
   int i, key, j;
   for (i = 1; i < n; i++)
   {
       key = arr[i];
       j = i-1;

       /* Move elements of arr[0..i-1], that are
          greater than key, to one position ahead
          of their current position */
       while (j >= 0 && arr[j] > key)
       {
           arr[j+1] = arr[j];
           j = j-1;
       }
       arr[j+1] = key;
   }
}
```