Name: ______________________________

Directions: **Work only on this sheet** (on both sides, if needed). MAKE SURE TO COPY YOUR ANSWERS TO A SEPARATE SHEET FOR SENDING ME AN ELECTRONIC COPY LATER.

1. (20) Fill in the blank: A ________________ is located either in or near the CPU, and contains a copy of part of memory.

2. (20) What will be printed out?

```
char z = 'L';
printf("%c",z + 0x11);
```

3. (20) Suppose we are a 32-bit Intel machine. Say **z** is declared as **int**, **&z = 200** and **z** contains $1 \cdot 16^5 + 2 \cdot 16^4 + 3 \cdot 16^3 + 4 \cdot 16^2 + 5 \cdot 16 + 6$. State the contents (as a base-10 number) of Byte 202.

4. The function below finds the sum of the elements in column **colnum** of a two-dimensional array **x** having **nrow** rows and **ncol** columns.

```
int colsum(int *x, int colnum, int nrow, int ncol)
{
   int sum = 0, m, *p;
   p = ______________________;  // (a)
   for (m = 0; m < nrow; m++) {
      sum += *p;
      p += ______________________;  // (b)
   }
   return sum;
}
```

State what code should go in the blanks:

(a) (10) Blank (a).

(b) (10) Blank (b).

5. (20) Consider a machine with 24-bit words and addresses. We will be storing numbers in the range 0,1,...,15; I'll call such numbers *glonks*. We want to store as many glonks as possible, so of course we will store multiple glonks per word. How many can we store in all of memory? Your answer must consist of an R expression. Assume that our machine has as much memory as possible, and that the operating system etc. take up only negligible space.

Solutions:

1. cache

2. ']'

3. c(**z**) = 0x123456, and we are on a little-endian machine, so 0x56 is in Byte 200, 0x34 in Byte 201 and 0x12 = 18 is in Byte 202.

4.a x + colnum

4.b ncol

5. `2 * 2^24`