HOMEWORK-1
Total Points: 40

1. [5 points] Consider the following declaration of a 2-D array.

   ```
   int a[4][3];
   ```

   List all elements (i.e., `a[0][0]`, `a[0][1]`, …, etc.) in the array. If the starting address of the array is 1000, compute the address of each element in the array. Which address will be written when you try to execute the statement "`a[1][7] = 100`"?

2. [5 points] Write a definition of the `strcmp` routine that takes two character arrays containing '\0' terminated strings. It returns 0 if the character strings are equal; otherwise, it returns 1. The prototype of the `strcmp` routine is as follows:
   ```
   int strcmp(char str1[], char str2[])
   ```

3. [20 points] Consider the definition of the power function below, as discussed in class.

   ```
   int power(int x, int n) {
     int pow_h;
     if (n == 0)                    // one comparison operation
       return 1;
     if ((n % 2) == 0) {            // one comparison operation
       pow_h = power(x, n/2);
       return pow_h * pow_h;        // one multiplication operation
     }
     else {
       pow_h = power(x, (n-1)/2);
       return x * pow_h * pow_h;  // two multiplication operations
     }
   }
   ```

   How many comparisons, multiplications, and function calls will occur during the computation of `power(3, 23)`?

   Now rewrite the definition of power that roughly computes `x^(n/3)` instead of `x^(n/2)` during the recursive step, and add more conditions to compute the correct value of `x^n`, where `^` is the power operator.

   How many comparisons, multiplications, and function calls will occur during the computation of `power(3, 23)` in your modified implementation?

fib(6), fib(5), fib(4), fib(3), fib(2), fib(1), fib(0), fib(1), fib(2), fib(1), fib(0), fib(3), fib(2), fib(1), fib(0), fib(1), fib(4), fib(3), fib(2), fib(1), fib(0), fib(1), fib(2), fib(1), fib(0)