CSC 362 Program #4
Due Date: Tuesday, November 10

Goldbach's conjecture is that every even integer greater than 2 is the sum of 2 prime numbers. For instance, 4 = 2 + 2, 6 = 3 + 3, 8 = 3 + 5, 10 = 5 + 5 (or 7 + 3), 12 = 7 + 5, 14 = 7 + 7, etc. You will prove this for the numbers between 6 and 100 inclusive (skip 4 since it is an exception in that it is the only even number whose prime sums are both even). You will solve this by writing an assembly language program. Your program will be embedded in C where you will do all variable declarations and output in C, but all other code in assembly. Here is the basic structure of your program:

```
#include <stdio.h>
void main( ) {
        // declare your variables in C and initialize them (if you need to)
        __asm { // enter assembly code, from here on, code is assembled, not compiled
                // assembly code here to iterate for sum = 6 to 100
                        // get the first x (3) and generate the first y (sum – x)
                        // have a loop to determine if x is prime by dividing some denominator
                        //      into x (starting at denominator = 2) until either you find one that
                        //     divides into x (x is not prime) or denominator reaches x (x is prime)
                        // if x is not prime, get next x/y pair (add 2 to x, compute y = sum – x)
                        //     otherwise determine if y is prime
                        // if you find a prime x/y pair, exit assembly code to output this, as in
                        //       8 = 3 + 5, return to assembly, increment sum by 2, repeat for loop
                        // otherwise if you increment x and it reaches sum, then no x/y pair is
                        //       found, exit assembly, output no solution found and exit program
                }
        }
```

Exiting the assembly code is done just by using }. You can return to assembly code using another __asm {…}. For instance, to output sum = x + y, you might have code like this:

```
… // in assembly code here
}  // exit assembly code
printf("%d = %d + %d\n", sum, x, y);
__asm { // re-enter assembly code
   …
 }
```

Note that printf makes use of edx. If you are using the edx for some temporary storage, you will wipe it out with the printf statement. So move the value in edx somewhere else (for instance a variable in memory) before exiting assembly code, and reload it into the edx after re-entering the assembly code.

Although it is repetitious, I suggest that you do not use a separate function to determine if x or y are prime. Instead, place the prime number code right into the places listed above. Yes, you are mostly duplicating code but by doing this, you don't have to worry about exiting the assembly code, calling a function, and then in the function having assembly code. This helps prevent wiping out registers like what happens when you use printf. Repeating code like this is not a good programming practice but it will hopefully simplify the program a little. If you do write the prime number code in a separate function, all of the code aside from the function's header, variable declarations, and return statement must be in assembly code.

NOTE: it is recommended that you write this program in C first to make sure you know how to solve it, and then convert the C code into assembly.

Run your program and obtain the output. Hand in your program (well commented) and your output.