Lab session (15th Dec)

Part 1:

- Declare an `int` variable 'var' and initialize to a value of your choice. Print "Even" if var is even, and "Odd" if var is odd without using the '%' operator! *(hint: integer division)*
- Declare, and initialize, an `int` variable 'sum' to some positive integer. Your program should then print, in seperate lines, every pair of positive integers a, b such that $a + b = sum$. *(Remark: you do not need nested loops for this!)*
- Declare, and initialize, an `int` variable 'product' to some positive integer. Your program should then print, in seperate lines, every pair of positive integers a, b such that $a \times b = product$. *(Remark: modify your previous program carefully)*

Part 2:

- Declare, and initialize, an `int` variable 'n'. Then print n copies of * in the first line, n-1 *s in the 2nd line, and so on till the last line that has a single *. Eg: If n = 4, then your output should be:

```
****
***
**
*
```

- The fibonacci sequence is defined as follows: The first two fibonacci numbers are $fib(1) = 0$ and $fib(2) = 1$. Then, $\forall n > 2$, the n'th fibonacci number is defined as $fib(n) = fib(n-1) + fib(n-2)$. Write a program where you first declare, and initizalize and `int` variable 'n', and then print $fib(1)\ \ fib(2)\ \ \cdots\ \ fib(n)$ each separated by a space. Eg: If n = 8, your output should be:

```
0 1 1 2 3 5 8 13 21
```

Part 3:

Learn by experiments. Some examples:

- See what happens if you use an `int` variable as the test condition for an `if` construct. Figure out exactly when the test is considered `true` and when it is considered `false`. Extend this to understanding how logical operators, such as `!`, `&&` etc. work with `int`, `float` operands.
- Learn the precedence of arithmetic, and logical operators in the C language by constructing arithmetic, and logical expressions, and testing their evaluations to understand in what order they were evaluated. [Hint: You can find precedence and associativity tables of C language online.)]
- Write a loop that does not terminate and see what happens.