CIIC 4030/ICOM 3046 Programming Languages
Assignment #5

In this assignment you will be using Go-Routines. Suppose that *Result* is a type representing a single result with its index from a slice. The structure definition can be written as:

```
type Result struct {
   index  int
   result string
}
```

Implement a *ConcurrentRetry* that runs all the tasks concurrently and sends the output in a *Result* channel. The parameter *concurrent* is the limit on the number of tasks running in parallel. Your solution must not run more than `concurrent` number of tasks in parallel. The parameter *retry* is the number of times that the task should be attempted. If a task returns an error, the function should be retried immediately up to `retry` times. Only send the results of a task into the output channel if it does not error.

Multiple instances of *ConcurrentRetry* should be able to run simultaneously without interfering with one another, so global variables should not be used. The function must return the channel without waiting for the tasks to execute, and all results should be sent on the output channel. Once all tasks have been completed, close the channel. The definition of the function must be as follows:

```
func ConcurrentRetry(tasks []func() (string, error), concurrent int, retry int) <-chan Result {
        // TODO
        return nil
}
```