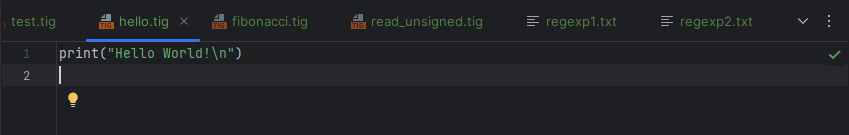
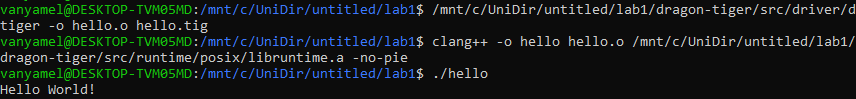
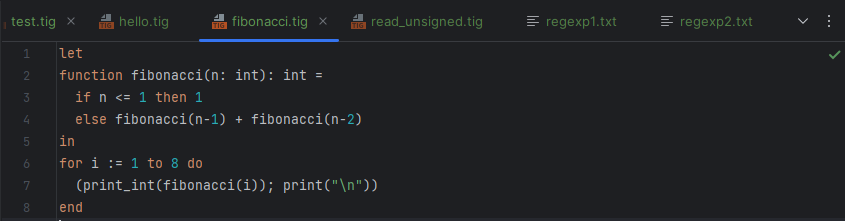
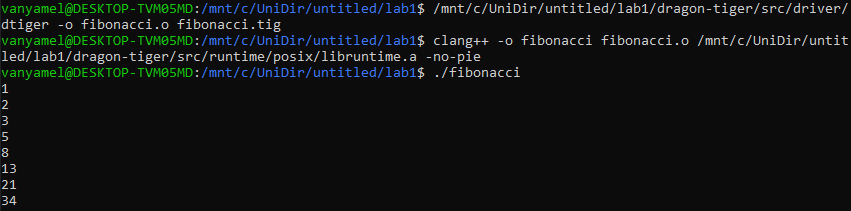
1. [ hello.tig ] Write a Tiger program, hello.tig , that prints the string “Hello World!” followed by a new line character to the standard output.



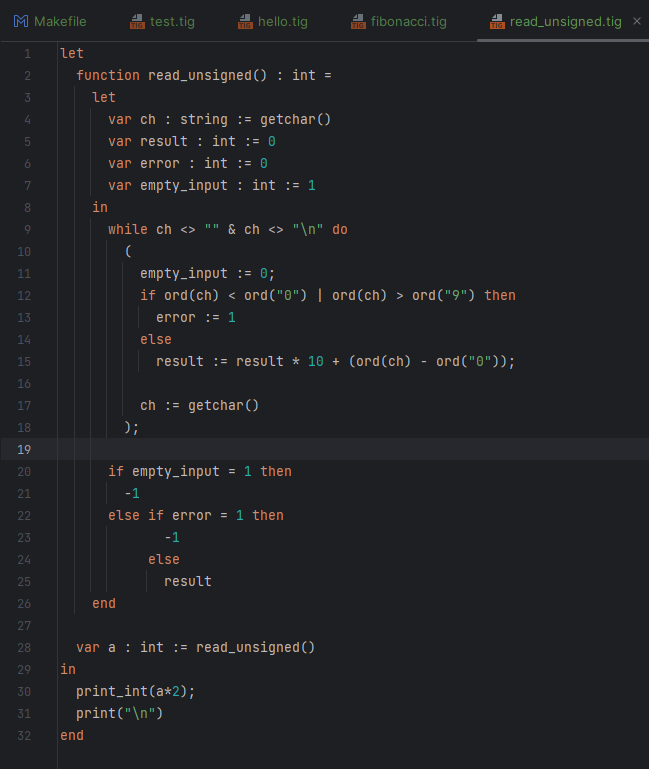


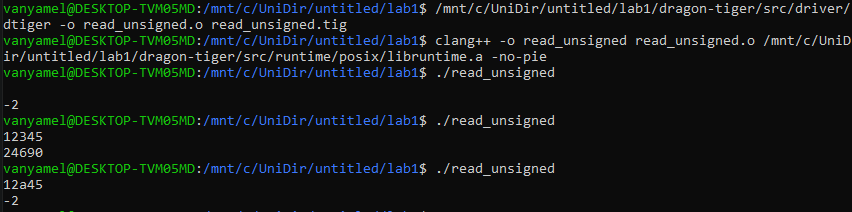
1. [ fibonacci.tig ] Complete the following program and save it in a fibonacci.tig file. The function fibonacci should return the \(n^{th}\) term of the Fibonacci sequence which is defined recursively as follows, \[f\_0 = 1, f\_1 = 1 \\ f\_{n+2} = f\_{n} + f\_{n+1}\]





1. [ read\_unsigned.tig ] Complete the following program and save it in read\_unsigned.tig file. The function read\_unsigned reads a line from the standard input ( stdin ). If the line contains only numerical characters it returns the number as a positive base-ten integer. Otherwise, it returns \(-1\). You should correctlyhandle the line termination character \n .





1. Regular Expressions and Finite Automata

Give a regular expression and an automata for each of the following languages in \(\Sigma = \{a, b\}\):

[ regexp1.txt ] words in \(\Sigma^{\*}\) for which the first a (if it exist) precedes the first b (if it exists).

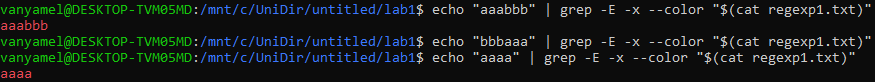
[ regexp2.txt ] words in \(\Sigma^{\*}\) for which the number of a is even (0 is considered even).

You will write regular expressions in the format accepted by grep -E . The two committed files will contain nothing but the regular expression. You can test

what a regular expression matches like this:

echo "aaaaab" | grep -E -x --color "$(cat regexp2.txt)"

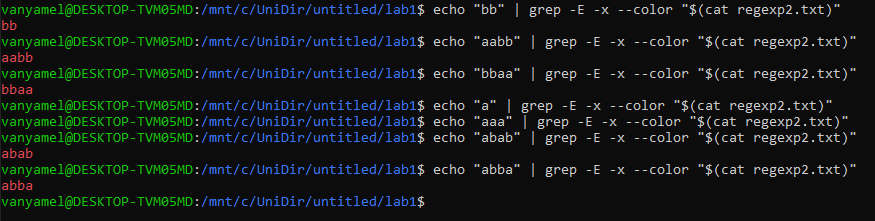
regexp1.txt  


Tests  


regexp2.txt



Tests



1. Lexer  
   