B4 - Functional Programming

B-FUN-400

Wolfram

Elementary Cellular Automaton

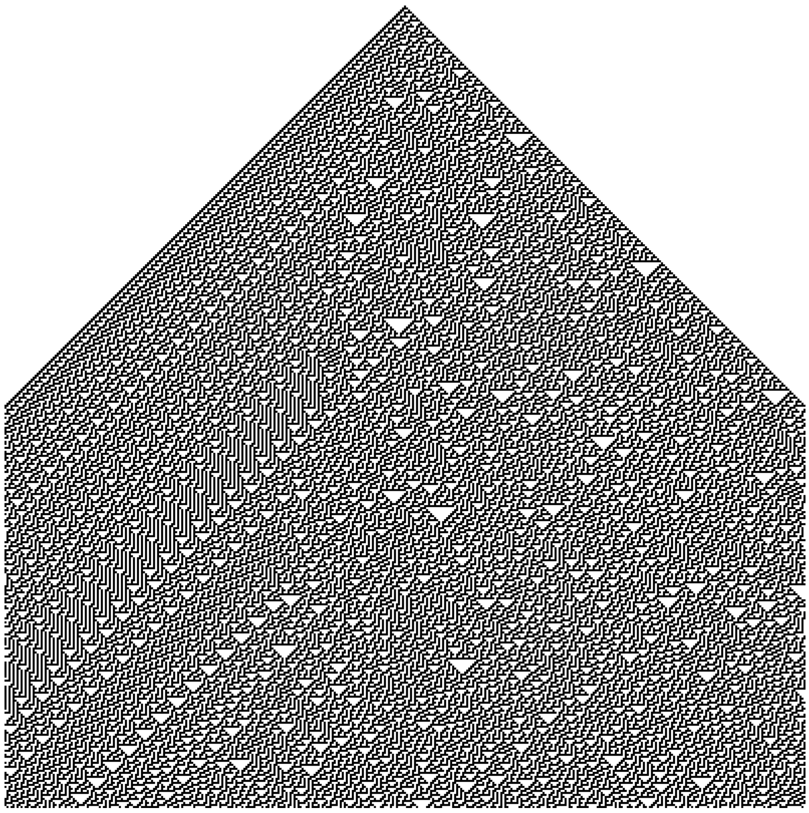
1.0

Wolfram

binary name:	wolfram
language:	Haskell
compilation:	stack wrapped in a Makefile *(see below)*

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Elementary Cellular Automaton



The goal of this project is to implement Wolfram's elementary cellular automaton in the terminal.

https://en.wikipedia.org/wiki/Elementary_cellular_automaton

You only have to implement rule 30, rule 90 and rule 110. The other rules are considered a bonus.

The space your cellular automaton are living in is infinite (to the left, right, and bottom).
This means the parts not shown on screen can still have an effect on future generations.

You have to handle the handling of the arguments yourself. Getopt is forbidden.

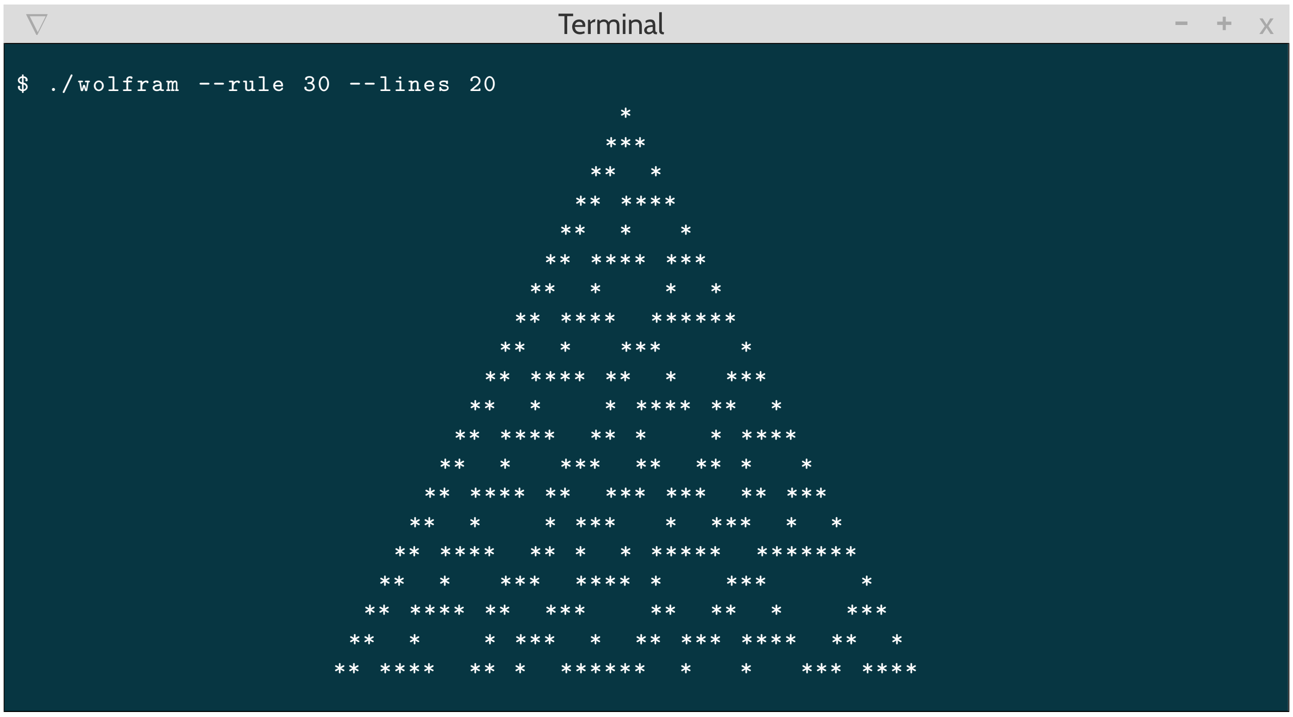
Invocation

Your program must support the following options:

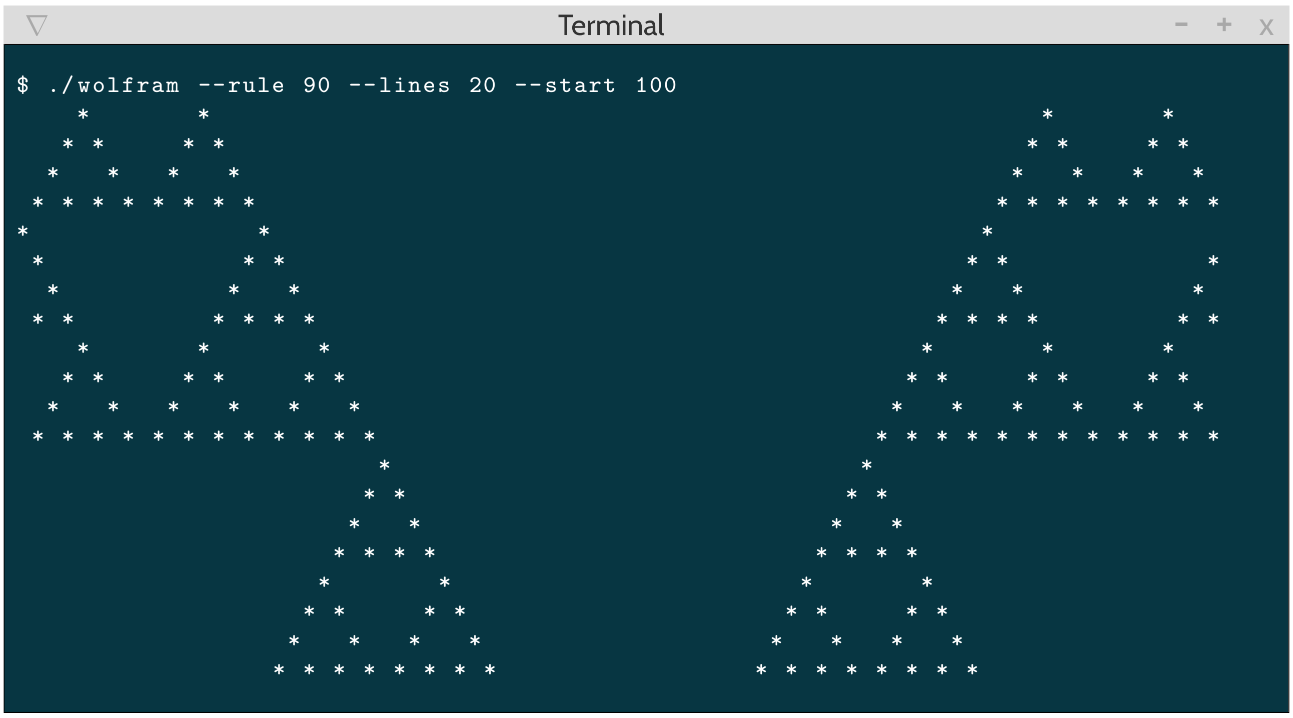
- **--rule** : the ruleset to use (no default value, mandatory)
- **--start** : the generation number at which to start the display. The default value is 0.
- **--lines** : the number of lines to display. When homited, the program never stops.
- **--window** : the number of cells to display on each line (line width). If even, the central cell is displayed in the next cell on the right. The default value is 80.
- **--move** : a translation to apply on the window. If negative, the window is translated to the left. If positive, it's translated to the right.

If no option or invalid options are provided your program must return 84 and display a usage message (eventually accompagned with an explicite error message of your choice)

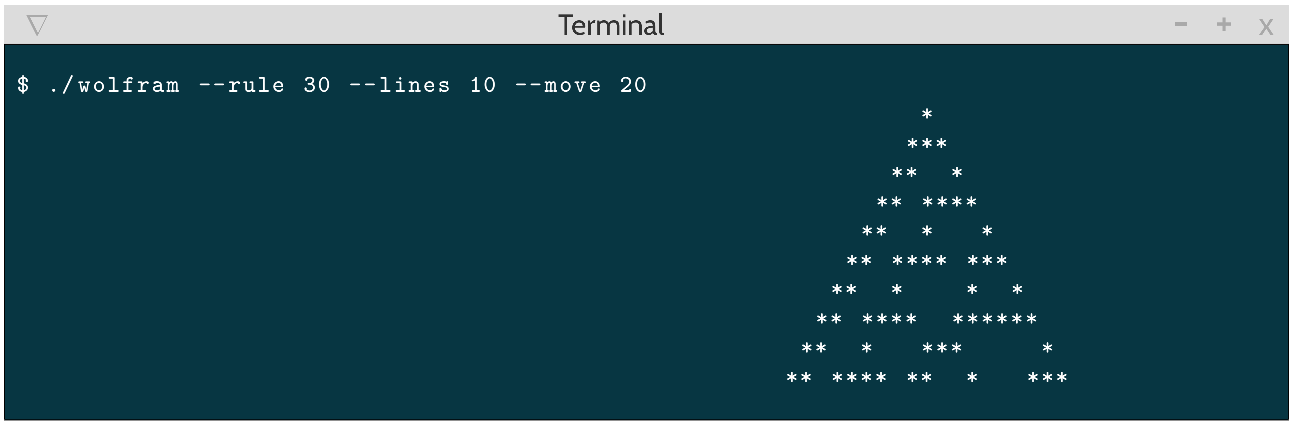
Examples



```
$ ./wolfram --rule 30 --lines 20
                                        *
                                       ***
                                      **  *
                                     ** ****
                                    **  *   *
                                   ** **** ***
                                  **  *    *  *
                                 ** ****  ******
                                **  *   ***     *
                               ** **** **  *   ***
                              **  *    * **** **  *
                             ** ****  ** *    * ****
                            **  *   ***  **  ** *   *
                           ** **** **  *** ***  ** ***
                          **  *    * ***   *  ***  *  *
                         ** ****  ** *  * *****  *******
                        **  *   ***  **** *    ***      *
                       ** **** **  ***    **  **  *    ***
                      **  *    * ***  *  ** *** ****  **  *
                     ** ****  ** *  ******  *   *   *** ****
```



```
$ ./wolfram --rule 90 --lines 20 --start 100
    *       *                                                       *       *
   * *     * *                                                     * *     * *
  *   *   *   *                                                   *   *   *   *
 * * * * * * * *                                                 * * * * * * * *
*               *                                               *
 *             * *                                             * *             *
  *           *   *                                           *   *           *
 * *         * * * *                                         * * * *         * *
    *       *       *                                       *       *       *
   * *     * *     * *                                     * *     * *     * *
  *   *   *   *   *   *                                   *   *   *   *   *   *
 * * * * * * * * * * * *                                 * * * * * * * * * * * *
                        *                               *
                       * *                             * *
                      *   *                           *   *
                     * * * *                         * * * *
                    *       *                       *       *
                   * *     * *                     * *     * *
                  *   *   *   *                   *   *   *   *
                 * * * * * * * *                 * * * * * * * *
```



```
$ ./wolfram --rule 30 --lines 10 --move 20
                                                            *
                                                           ***
                                                          **  *
                                                         ** ****
                                                        **  *   *
                                                       ** **** ***
                                                      **  *    *  *
                                                     ** ****  ******
                                                    **  *   ***     *
                                                   ** **** **  *   ***
```

Hints

The main purpose of this project is to let you discover Haskell and it's syntax.

Considering it's simplicity, you should (and are expected) to produce the cleanest code possible. This project is a great opportunity to use some of the great features of Haskell, such as pattern matching and guards, partial application and currying of functions, higher order functions and closures, isolation of side effect ridden code from pure code.

And as always, testing your code is the surest way to not introduce new bugs to your code base…

Bonus

- Support all the 256 possible rules.
- add a -generation argument, in which case the program reads a single line from stdin and use it as a first generation.
- output bitmap images.
- display your automaton in graphic mode in real time.

BUILD WITH STACK

Stack is a convenient build tool/package manager for Haskell.
Its use is required for this project, with **version 2.1.3 at least**.

It wraps a build tool, either **Cabal** or **hpack**.
You are required to use the hpack variant (package.yaml file in your project, autogenerated .cabal file).

This is what stack generates by default with `stack new`.

Stack is based on a package repository, **stackage**, that provides consistent snapshots of packages.
The version you use must be in the **LTS 16** series (`resolver: 'lts-16.16'` in stack.yaml).

In `stack.yaml`, extra-dependencies cannot be used.

base is the only dependency allowed in the `lib` and `executable` sections of your project (package.yaml).
There is no restriction on the dependencies of the `tests` sections.

You must provide a **Makefile** that builds your stack project (i.e. it should at some point call 'stack build').

'stack build' puts your executable in a directory that is **system-dependent**, which you may want to copy.
A useful command to learn this path in a system-**independent** way is:
`stack path --local-install-root`.

The automatic test system expects to find the file stack.yaml of your project at the root of your repository