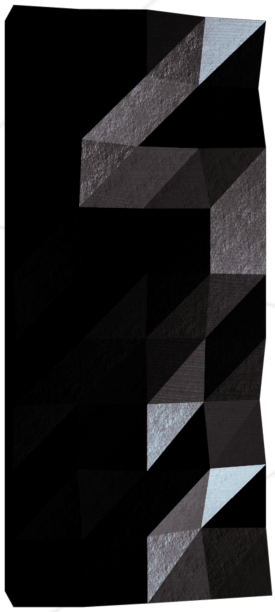

AlCu

A rush on algorithms

Summary: This rush will make you work on efficient programming methods and algorithms.

Version: 1.1

Contents

I Preamble **2**

II Introduction **3**

III General rules **4**

IV Project instructions **5**

IV.1 Game instructions 5

IV.2 Program instructions 5

V Example **7**

VI Bonus part **8**

VII Turn-in and peer-evaluation **9**

Chapter I

Preamble

A marble is a small spherical object often made from glass, clay, steel, plastic, or agate. Marbles can be used for a variety of games called marbles.
The games have been played in many countries, but the rules are made up by the players, and there are many variations. One standard idea is to have a target marble. Players flick their marbles with their thumbnails and try to hit the target.
Another version involves players trying to hit each other's marbles out of a target zone.

Chapter II

Introduction

This project invites you to create a turn-based strategy game. The board consists of N heaps containing any number of items. Each player alternates by taking between 1 to 3 objects from the last heap. The goal is to be the last player to take an object.
Chance does not intervene. Be a `winner`.

Chapter III

General rules

- Your assignment must be written in `C`.
- No norm requirements.
- `CC` is used as the compiler.
- You have to compile your program with the following flags: `-Wall -Wextra -Werror`
- You have to provide a `Makefile` to compile your source files. It must not relink.
- Your `Makefile` must at least contain the rules: `NAME`, `all`, `clean`, `fclean`, and `re`.
- Your program should not quit unexpectedly (segmentation fault, bus error, double free, and so forth) except for undefined behaviors. If it happens, your project will be considered non-functional, and your grade will be `0`.
- Within the mandatory part, you're allowed to use the following functions:
  - `read`
  - `write`
  - `open`
  - `close`
  - `malloc`
  - `free`

Chapter IV

Project instructions

IV.1 Game instructions

- The game will be played on a board. For example, here is a board:

```
   |||||||
    |||||
     |||
      |
```

- There is a fixed number of heaps.
- There are 2 players, one of whom is an AI.
- Players take turns, and the AI starts first.
- On each player's turn, they can take between 1 and 3 items from a single heap.
- The player who takes the last item from the last available heap loses.

IV.2 Program instructions

- The board will be read from a file or standard input, with an empty line signaling the end of input if no arguments are given.
- The board will be formatted according to the following rules:
  - Each line indicates the number of items present on that line, followed by a newline.
  - This number must be between 1 and 10000.
  - If the board is incorrect, you will write `ERROR` to standard error, followed by a newline.
- The board must be displayed after each turn.

- On a player's turn, you will ask them how many items they want to remove.
  - Items are removed from the last heap.
  - This must be a valid number.
  - In case of error, you will ask again.
- Your AI must try to win.
- At the end of the game, the winner will be announced.

`A part of your grade will be based on your AI algorithm.`

Chapter V

Example

```
$> cat -e alcu.map
8$
5$
3$
2$
1$
$
$> ./alum1 < alcu.map
 | | | | | | | |
   | | | | |
     | | |
      | |
       |
AI took 1
 | | | | | | | |
   | | | | |
     | | |
      | |
Please choose between 1 and 3 items
3
3 - Invalid choice
Please choose between 1 and 3 items
2
 | | | | | | | |
   | | | | |
     | | |
...
You are the winner! Congratulations!
```

Chapter VI

Bonus part

The only bonus is to provide some sort of visualizer. You're free to use any library you want; here are some:

- Termcap
- ncurses
- MiniLibX
- SDL
- Metal
- OpenGL
- ...

The bonus part will only be accessible if the mandatory part is PERFECT. "Perfect" means the mandatory part has been completed in full and works without any malfunctions. If you have not passed ALL the mandatory requirements, your bonus part will not be evaluated at all.

Chapter VII

Turn-in and peer-evaluation

As usual, turn in your work on your Git repository. Only the work included in your repository will be reviewed during the evaluation.

Good luck.