The University of Queensland
School of Information Technology and Electrical Engineering
Semester Two, 2016
CSSE2310 / CSSE7231 - Assignment 3

Introduction

In this assignment, you will write two types of programs `C99` programs to play a game "King of St Lucia". Rules for the game were supplied in a separate document. The first type of program (players) will listen on their `stdin` for information about the game and give their moves to `stdout`. These types of program will send information about the state of the game to `stderr`. The second program (`stlucia` — also known as the hub) will start a number of processes to be players and communicate with them via pipes. The stlucia will be responsible for running the game (sending information to players; processing their moves and determining the score). The stlucia will also ensure that, information sent to `stderr` by the players, is discarded.

The idea is that you will produce a number of player programs (with different names) which implement different playing strategies. However all players will probably share a large amount of common code (with the other players you write).

Your programs must not create any files on disk not mentioned in this specification or in command line arguments. Your assignment submission must comply with the C style guide (version 2.0.3) available on the course blackboard area.

Invocation - players

The parameters to start a player are: the number of players (in total) and the label of this particular player (starting at A). For example:
`./player 3 B`

The maximum number of players permitted in a game will be 26.
After argument checking has completed successfully, players will output a single exclamation mark (!) to `stdout`.

Invocation - stlucia

The parameters to start the hub are:

- The name of a file containing a list of dice rolls to use.
- The number of points to play to.
- The names of programs to run as players (one for each player). There must be at least two players in the game.

For example, executing: `./stlucia ex.rolls 15 ./typeA ./typeA ./typeA`
would start a game with 3 players (each running `./typeA`) and using the rolls contained in `ex.rolls`. The game will end when at least one player has 15 or more points.

Representations

Whenever rolls are represented as strings, use the following symbols: 1, 2, 3, H, A, P. For ordering purposes: $1 < 2 < 3 < H < A < P$. Roll files will contain the above characters and newlines in any combination (your implementation should skip over newlines). Any other character in a roll file is an error. Your program should read the entire roll file into memory when it starts rather than trying to detect problems later. When all rolls are exhausted, go back to the beginning.

How your players will work

EAIT

Rerolls: If they have more than 2 of the same number (1,2 or 3), they will keep them. If they have less than 6 health, they will keep any H they roll. Apart from that, they will reroll as many dice as possible as many times as possible.

Retreat: This player will retreat from StLucia if they have less than 5 health points, otherwise they will stay.

SCIENCE

Rerolls: If they have less than 5 health, they will keep any H they get and reroll everything else. If they have 5 or more health, they will keep any A they get and reroll everthing else.

Retreat: This player will retreat from StLucia immediately.

HABS

Rerolls: If they have less than 5 health, then they will reroll any As they have. Apart from that, they will not reroll.

Retreat: This player will not retreat unless they have less than 4 health. If there is only one other player left, they will never retreat.

HASS

Rerolls: If they are in StLucia, they will never reroll Ps. They will reroll everything else. If they are not in StLucia, they will reroll As unless the As they have rolled are enough to eliminate the player in StLucia. They will reroll everything else.

Retreat: This player will not retreat from StLucia.

MABS

Rerolls: If they are not in StLucia, they will always keep Hs and 3s. Apart from this, they will reroll everything else. If they are in St Lucia, they will keep 3s and As. Apart from this, they will reroll everything else.

Retreat: This player will retreat from StLucia immediately.

Messages

Messages from stlucia to player

Message	Params	Meaning
turn $d_1d_2d_3d_4d_5d_6$	d_i are dice values arranged smallest to largest.	It is your turn and here is your roll
rerolled $d_1d_2d_3d_4d_5d_6$	arranged in increasing order.	Used to communicate the result of a reroll
rolled p $d_1d_2d_3d_4d_5d_6$	p is the letter of the player	used to inform other players of Player p's final dice (Note: this message is not sent to the player who rolled those dice.)
points p v	p is the letter of the player v is a number of points	inform all players when a player is awarded points
attacks p v in/out		inform all players that Player p is attacking for v points of damage either **in**to or **out** of StLucia.
eliminated p		Inform all players that Player p is out of the game. Player p should close down at this stage.
claim p		Inform all players that Player p has claimed StLucia.
stay?		Ask the player in StLucia who has just been attacked if they wish to leave
winner p		Inform all remaining players that Player p has won
shutdown		Inform all players that it is shutting down (used if stlucia is shutting down due to a signal)

For example Player D might see:

```
rolled A 123HHH
rolled B 122AAA
claim B
points B 1
rolled C 23HAAP
attacks C 2 in
claim C
points C 1
turn 112HPP
```

Messages from player to stlucia

Message	Params	Meaning
keepall		inform stlucia that you do not wish to reroll any of your dice
reroll $d_1 \ldots d_k$	dice faces to reroll	Inform server which dice you wish to reroll. You would expect a `rerolled` in response.
stay		inform the server you wish to remain in StLucia (in response to a `stay?` message.
go		inform the server you wish to leave StLucia (in response to a `stay?` message.

Player output to stderr

When a message (eg XYZ) arrives from the hub, print the following to `stderr`
`From StLucia:XYZ`
After the message has been processed, print any player specific output described in the strategy section.
The player program will also generate some error messages which are to be sent to `stderr`.

Order of processing during a turn

After a player has decided on their dice, communications and checks should happen in the following order:

1. Inform other players what was rolled.
2. Healing (stderr reports healing amount as 0 if there were H dice but the player is already at maximum health points).
3. Attacks are processed and damage reported.
4. New player claims StLucia
5. Points for the turn are reported
6. Player eliminations reported
7. Test for game over

Hub output

The stlucia should discard any output sent to the `stderr` of player processes. As well as errors outlined below(which will also be sent to `stderr` — it's ugly but go with it), the hub should output the following to `stderr`.

- When a player's dice are finalised, print:
  `Player ? rolled ??????`
- When a player scores points, print:
  `Player ? scored ? for a total of ?`

- When a player takes damage (this amount can not be larger than their remaining health), print:
  `Player ? took ? damage, health is now ?`
- When a player heals (this amount could be zero if they are already at 10 health) print:
  `Player ? healed ?, health is now ?`
- When a player enters StLucia, print:
  `Player ? claimed StLucia`
- When a winner is declared, print:
  `Player ? wins`

Exits

Exit status for player

All messages to be printed to `stderr`.

Condition	Exit	Message
Normal exit due to game over	0	
Wrong number of arguments	1	`Usage: player number_of_players my_id`
Invalid number of players	2	`Invalid player count`
Invalid player ID	3	`Invalid player ID`
Pipe from stlucia closed unexpectedly (i.e. before a winner message was received)	4	`Unexpectedly lost contact with StLucia`
Invalid message from stlucia	5	`Bad message from StLucia`

Exit status for stlucia

All messages to be printed to `stderr`.

Condition	Exit	Message
Normal exit due to game over	0	
Wrong number of arguments	1	`Usage: stlucia rollfile winscore prog1 prog2 [prog3 [prog4]]`
Winscore is not a positive integer	2	`Invalid score`
Unable to open rolls file for reading	3	`Unable to access rollfile`
Contents of the rolls file are invalid	4	`Error reading rolls`
There was an error starting and piping to a player process	5	`Unable to start subprocess`
A player process ends unexpectedly. ie Reading from the pipe from that process fails before a gameover message has been sent to it. [only applies once all child processes have started successfully]	6	`Player quit`
One of the players has sent an invalid message	7	`Invalid message received from player`
One of the players sent a properly formed message but it was not a legal action	8	`Invalid request by player`
The stlucia received SIGINT	9	`SIGINT caught`

Note that both sides detect the loss of the other by a read failure. Write calls could also fail, but your program should ignore these failures and wait for the associated read failure. Such write failures must not cause your program to crash.

Shutting down stlucia

Whether stlucia shuts down in response to a SIGINT or of its own volition, it should follow the same procedure. It should ensure that all child processes have terminated. If they have not all terminated within 2 seconds of the hub sending the `winner` / `eliminated` / `shutdown` message, then the hub should terminate them with SIGKILL. For each player (in order), do one of the following:

1. If the process terminated normally with exit status 0, Then don't print anything.
2. If the process terminated normally with a non-zero exit status, then print (to `stderr`):

   `Player ? exited with status ?`

3. If the process terminated due to a signal, then print (to `stderr`):

```
Player ? terminated due to signal ?
```

(Fill in ? with the appropriate values)

Compilation

Your code must compile (on a clean checkout) with the command:
`make`

Submission

Submission must be made electronically by committing using subversion.