

Demo Plan:

For our demo plan, we created input files to show that the implementation of our Biquadris program is as specified. Each input file contains a sequence of commands that show an implementation of a feature in our program. The program can be run with or without the graphic display. To show the text display only, add the `-text` flag when running the program. The specifications of each input file are as follows:

Basic Functionalities:

Please use our graphic display (without `-text`).

1. Command-line Interface

We begin by introducing all our command-line options:

```
-text                : show the text display only
-seed xxx            : set the generation of blocks to xxx seed
-scriptfile[1-2] xxx : uses xxx instead of sequence[1-2].txt
                        as a source of blocks for level 0, for player
-startlevel n        : start the game in level n
```

Sample run: `./biquadris -text -startlevel 4 -seed 4`

2. Left, Right, Down, Drop movement

We have created a file of command sequences that show the block's movement. Run `./biquadris < demo/movement.in`

3. Rotate movement and Change Block command

We have created a file of command sequences that show the block's rotation and the command to change blocks. Run `./biquadris < demo/rotate.in`

4. Player Loses

When a block cannot be generated at the top left, the player has lost. Try playing the game by only dropping the blocks in its starting position (use `8drop` command to make player one lose quickly).

5. Level Change and Restart

We have created a file of command sequences that show the implementation of the level-up and level-down commands. We also show the implementation of restart and how the high score is kept on the scoreboard. Run `./biquadris < demo/level.in`

6. Block Sequence and Seed

We have created a file of command sequences that show the implementation of the script file and seed option of our program. Run `./biquadris -scriptfile2 sequence3.txt -seed 100 < demo/seq.in`. Try running also with `-scriptfile1 sequence3.txt`

7. Level 0:

We have created a file of command sequences that show the implementation of Level 0. It shows that the block sequence repeats after it ends and the scoring for level 0. P

Run `./biquadris -text < demo/level0.in`

8. Level 1:

We have created a file of command sequences that show the implementation of Level 1. It shows that the blocks are generated randomly and the scoring for Level 1.

Run `./biquadris -startlevel 1 < demo/level1.in`

9. Level 2:

We have created a file of command sequences that show the implementation of Level 2. It shows that the blocks are generated randomly and the scoring for Level 2.

Run `./biquadris -startlevel 2 < demo/level2.in`

10. Level 3:

We have created a file of command sequences that show the implementation of Level 3. It shows that the blocks are generated randomly and implements “heavy” blocks. The command sequence also shows the no-random and random command and the scoring in Level 3. Run

`./biquadris -startlevel 3 < demo/level3.in`

11. Level 4:

We have created a file of command sequences that show the implementation of Level 4. It shows that the blocks are generated randomly and implements “heavy” blocks. We also show the implementation of an additional 1x1 block if no blocks were cleared in 5 turns and the scoring in Level 4. Run `./biquadris -startlevel 4 < demo/level4.in`

12. Special Actions and Sequence:

We have created a file of command sequences that show the various implementations of the special actions. We also show the implementation of the sequence command.

Blind: Run `./biquadris < demo/blind.in`

Heavy: Run `./biquadris < demo/heavy.in`

Force: Run `./biquadris < demo/force.in`

13. Level Difference Points:

We have created a file of command sequences that show the difference in points when a row is cleared in a different level and when a block is cleared in a different level as its generation. Run `./biquadris < demo/diff.in`

14. Hint command:

Try inputting `hint` after playing some rounds. The program will give you the location for the best dropping position. To give you a better understanding, try running `./biquadris` and input

sequence `demo/fourlines.in` Now if you type `hint`, it will tell you that the rightmost column is the best place to drop the block!

15. Rename command:

We have created a file of command sequences that show the implementation of an additional rename command. Run `./biquadris < demo/rename.in`

Expected Specific Behaviours:

As you may have seen, our basic functionalities are covered both in the text display and graphic display. To save time, we hope for you to use the `-text` option for our specific behaviours.

16. Changing and rotating a block on the right edge of the board

Run `./biquadris -text` We want to show that changing and rotating a block at the right edge of the board will not change the state of the block, since we are using the bottom left position of the block as its axis. Run `./biquadris -text` Input command `10ri` then `clo` and `3ri`. Now if we input `clo`, it does not change the orientation. Now, if we input `J`, it does not change the block since `J` cannot be generated at the edge.

17. Changing blocks in a closed space

We want to show that sometimes we cannot change the block if it does not fit. Run `./biquadris -text` Then input command `seq demo/invalidchange.in` Now, try changing the block to any block. If you try to change into an `I` block, notice that it does not change since an `I` block has 4 length whereas the available area is only 3 length.

18. Changing opponent's block by `force` and opponent loses

Run `./biquadris -text` Then input command `seq demo/forcelose.in` Now, type `force` then `I`. As you may have seen, when player 2 chooses `force I` command, player one loses since `I` cannot be generated at the top left of the board.

19. Moving blocks at the bottom row

Run `./biquadris -text` Then input `13do`, down, and try moving `left` and `right`. The block has not been dropped, thus the block can still be moved. However, if you `3levelup` and restart, at the next turn (try drop 2 times)

20. Zero multiplier prefix

Run `./biquadris -text` and try using the `0` multiplier prefix for basic movement, and especially drop. When the command is `0drop` notice that the player has not change since the block has not been dropped.

21. Drop multiplier

Run `./biquadris -text` Then input `3drop` Notice that 3 blocks are dropped for the current player, and not alternating between players, as it would make the game unfair. One might say

that this would also be unfair since the current player would have more blocks than the opponent, but as they have no control with the placement, it is considered fair.

22. Changing blocks after changing level

Run `./biquadris -text` Then input `3ri` Notice that since the level is 0, the block has no weight. However, if we change the block to `I` (the same block) and move the block to the `right`, the block moves down 1 row since the new block's level is determined by the (current) level number.

23. New generated blocks after changing level

Run `./biquadris -text` Then immediately change level by inputting `3levelup`. The expected behaviour is that the current block and the next block shown below will have no weight, but the next generated block will have 1 weight. Try moving the first two blocks `right` and `left`. Then try moving the third block `right` and `left`. Notice how only the third block has weight.

24. No random from level 3 to level 4

Run `./biquadris -text -startlevel 3` Try playing the game for a few rounds and notice that the block generation is random. Now, if we set `norandom sequence3.txt` the game only generates an `I` block for the player that invokes the command. Now, if we increase the level `levelup` for the player who uses `norandom`, and try playing for some rounds, notice that `norandom` is still set since it is still relevant. If we `4leveldown`, notice that at the next round, the next block generation is now set back to the default sequence file (`sequence[1-2].txt` or the file from `-scriptfile[1-2]` option).

25. Weight of Heavy and level

Run `./biquadris -text -startlevel 3` and use `sequence demo/fourlines.in`. Now, when you drop the next block, choose special action `heavy`. Now, player two's current block should have a weight of 3 (2 from `Heavy` and 1 from level 3). Try moving left a few times and notice that with each movement, the block drops by 3 rows. After dropping the current heavy block, the next block will only have 1 weight.

26. Line cleared by a one-by-one block generated in level 4

We want to show what will happen to the one-by-one block that is generated in the middle of the board and clears a row. Run `./biquadris -text -startlevel 4`

Input command `seq demo/oneblockclear.in` (sequence 1x1 block clear).

The number of blocks dropped without clearing a line is now 4. Now input `drop`. A one-by-one block was generated (not shown in the display) but immediately clears the bottom line. The score is now 100, calculated from:

Level 4; 1 line cleared: $(4+1)^2 = 25$

Level 4; 3 blocks cleared (including one-by-one block): $3*(4+1)^2 = 75$

Total score: $25 + 75 = 100$

We decided that a one-by-one block can also increase the score to reward players. The one-by-one block can be considered as both an advantage and a disadvantage.