Overview

1.1 Description

RIOT! is a reverse tower defence game set in a prison theme. RIOT! intends to switch the roles of a traditional tower defence game, placing the user in control of what would conventionally be the enemy horde.

The game's objective is to escort the protagonist through each of correction facilities, with each one having progressive difficulty. The player must send waves of inmates (ranging from the quick-moving Speedy's to shot-absorbing Fatties) to clear the way for the protagonist.

Whether it's trying to sober up from the Drunk Tank or escaping the clutches of the fabled Guantanamo, the difficulty only goes up as more guards arrive to bring your attempts to a halt.

1.2 Technical Features

1.2.3. Graphical Interfaces

RIOT! is rendered through a text based graphics library called ncurses.

1.2.3.1. Main Menu

In the Main Menu a single ncurses windows is used to present the title and menu options to the user using a series of print statements.

1.2.3.2 Intro Screen

A function is used to retrieve intro text from corresponding map file based on selected level and displayed using the same method as the main menu

1.2.3.3 Game Screen

In the game screen the ncurses window is divided into 3 bordered windows (Header, Body, Footer).

1.2.3.3.1 Header

A function is used to read in the name, panic and rep given for a corresponding level number and is printed to the header window. These values are redrawn as panic/rep changes

1.2.3.3.2 Body

A function reads a 2D array of type char and prints each character to the screen where each character represents a separate game component.

Objects:

"." - Inmate path

"#" - Door

"%" - Mud

"$" - Starting Position of path

"&" - Ending position of path

"-". "|", "+" - Border walls

Inmates:

"h" - homeboy

"b" - bruiser

"l" - lunatic

"c" - cutie

"d" - doctor

"f" - fatty

"s" - speedy

"a" - attorney

"p" - protagonist

Guards:

"G" - guard

"D" - Dog

"L" - Lunch Lady

"P" - psychiatrist

"S" - Sharpshooter

"W" - Warden

"C" - Cyborg

Colour of inmates are used to signal their remaining hit points:

Green = 100% - 75%

Yellow = 74% - 50%

Red = 49% - 25%

Purple = 25% - 0%

A queue containing 5 slots is drawn on the right side of the game window. This is used to show Inmate inventory and is updated as units are queued and dequeued

1.2.3.3.3 Footer

An array of type char for a corresponding level is passed into a function. The array is then printed into the footer window which represents the inmates available for the current level.

1.2.3.4

Upon exit of the game the ncurses window is closed and the original terminal is restored to its previous state. "Thank you for playing" and an "Exiting" messages are then printed to the command line

2.0 Instructions

In-order to play RIOT! You will have to do following...

1. From the command line inside the root folder of the game, type and enter “make” in-order to compile the game.

2. Afterwards, type “./bin/riot” to run the game

3. Once the main menu is shown, you will be given the following options:

    i. [n]ew game

    ii. [c]ontinue

    iii. [e]xit

Choose the options with the following letter inside the bracket:

New game sets the player inside the tutorial level so that they can learn how to play.

Continue allows the player to start wherever they left off or replay a level that's been completed. They will not be able to choose a level that they haven't unlocked.

Exit exits RIOT!

4. Once a level is selected, the player will have to press any key continue to the game.

5. Within the level, the available units will be displayed in the INMATES section below. The player can pick which unit they would like to use and the selected unit will be placed in the queue according to the letter picked.

6. To select the inmate the player must have enough rep to purchase them. The higher the rep, the more inmates you can get.

7. Once the player selects the inmates they want, they can either press enter to send them or fill up the queue and that will automatically send them through.

8. With every inmate that escapes, the panic will increase, causing the towers to lose accuracy. The higher the panic, the easier it gets.

9. If you complete the level, the player will be prompted with a message that continues the story, otherwise it will prompt them to start it again.

Final statistics of the Inmates:

Name	Health	Speed	Rep	Panic	Ability/Comment
[p]rotagonist	5	4	0	0	The game's unnamed protagonist; must escape to complete the level
[h]omeboy	10	2	10	2	Basic starting unit
[b]ruiser	16	2	15	2	A beefier homeboy but at a beefier cost
[l]unatic	16	2	10	8	Causes a lot of problems
[f]atty	40	8	10	4	'Tank' unit
[s]peedy	10	1	20	2	Fastest unit
[c]utie	20	4	20	1	Mediocre but cheep unit
[a]ttorney	30	4	30	2	Knows their way around a prison
[d]octor	10	4	40	2	Heals units in front of them

Final Statics of the Guards:

Name	Damage	Range	Cool-down	Targeting	Ability/Comment
[G]uard	5	2	6	Proximity	Neutral unit
[D]ogs	2	4	6	Area of Effect	Weak but fast attacking unit
[L]unchlady	0	6	12	Area of Effect	Inmate speed halved for 12 cycles
[P]sychiatrist	0	6	16	Area of Effect	Puts an inmate to sleep for 6 cycles
[S]harpshooter	6	10	12	Closest to exit	Can reach long distances
[W]arden	100	2	60	Proximity	Instantly kills a unit
[C]yborg	12	8	8	Proximity	Available only on LunarMax

3.0 Design Changes

- There are now only 8 completable levels instead of 9, with the last level being the bonus Lunarmax level on the moon.
- Tick or cycle speed is changed to one-half of a second; originally was 1/6th of a second, but the game was deemed too fast paced.

```
- delay.tv_sec = 0;

  delay.tv_nsec = CYCLE;

  ...

  enqueue(&deployed, dequeue(queued));
```

- Because of nCurses limitations, colours to indicate the inmates' health are changed to, in order from highest hp to almost dead: green, yellow, red, purple.
- One feature that was added was the ability to press [delete] to remove a unit from the queue.
- Units with 100-75% of their initial health are green, with 74-50% are yellow, with 49-25% are orange, and lastly with 24-1% are red.

```
- init_pair(DEFAULT, COLOR_WHITE, COLOR_BLACK);
  init_pair(GREEN, GREEN, COLOR_BLACK);
  init_pair(YELLOW, YELLOW, COLOR_BLACK);
  init_pair(RED, RED, COLOR_BLACK);
  init_pair(PURPLE, PURPLE, COLOR_BLACK);
  init_pair(DAMAGED, COLOR_BLACK, DAMAGED);
  ...
  for (i=0; i< getLength(inmateList); i++){
      inmate = (struct Inmate*)nextInmate->unit;
      wattron(body,COLOR_PAIR(getColor(inmate)));
      coordinates = getCoordinate(inmate->position);
      mvwaddch(body, coordinates[0], coordinates[1], inmate->type);
      wattron(body,COLOR_PAIR(DEFAULT));
      eraseInmatePos(body, path,getPrevPos(path,inmate));
      nextInmate=nextInmate->next;
```

- riotIO is not explicitly used (refer to 4. Incomplete Features)
- The protagonist unit is now queued by pressing [P]. It was previously set to [R] as [P] is used for psychiatrist, but using the letter [P] is less confusing than [R].

```
 - switch (input) {
        case PROTAGONIST:
            if (!ifProt) {
                ifProt = TRUE;
                inmate = createInmate(input);
                enqueue(inmates, inmate);
                updateQueue(win->body, inmates, getLength(inmates));
            }
            else {
                mvwprintw(win->footer, 0, 40, "PROTAGONIST ALREADY
PRESENT");
            }
   ...
```

- Map size changed from 76x16 as the intended size did not accommodate
- There is no bribing mechanism at all. There will be so much going on the screen that it will take away from how much information the user will process at that given time.

4. Incomplete Features

- riotIO
    - riotIO was scrapped because it would have been too small to warrant its own file.
    - fileIO with writing and reading from the .riot files are handled in riotMap.
    - User input is moved to riotUI where it is join with selecting units.
- inmate profiles
    - Not implemented because of time constraints.

- Pressing f8 to quit is not implemented because nCurses does not allow inputs while the real-time portion of the game is running.
- Information screen not implemented due to time constraints.
- There were some special abilities we didn't have enough time to implement for these specific characters:
    - Dogs-> when a dog attacks you, it was supposed to wound you so the next attack you take will deal double the damage. It didn't make sense to the user playing because the wounded inmates would die after one hit from the guards.
    - Lunatic-> they were supposed to deal adjacent to inmates because of their ridiculously high statistics where their speed and panic they cause is insanely fast.
    - Speedy-> was suppose too increase adjacent inmates speed by 2, and up to a maximum of 6. But wasn't implemented due to game balancing
    - Cutie-> was supposed too disable guard attacks within a 6 tiles wasn't implemented due to time constraints
    - Also we weren't able to implement the '%' char that was placed on the path way designed to slow down users because of time restraints.