<Adventure Game>

By Marisa Rea, Tucker Dane, David Pipitone

Oregon State University

Description

<Adventure Game> was designed to be a video game designed using C++ to run on Linux. The core concept was to build a game based on features and themes from video games of the past. Themes include, Pac-Man, Pokemon, Zelda, and Super Mario.

The user plays a majority of the game from a top-down view. Two of the rooms are played from the perspective of a side-scrolling video game and feature simulated gravity.

The user can pick up, drop, equip, and use items. They can also interact with their environment by reading statues or pushing movable objects. Mastering all of these techniques will help the user beat the game.

ncurses

To help us create <Adventure Game> we used a the ncurses library, which allowed us to create different windows for inventory, world view, status, and narrative. These windows could be sized differently and related information could be stored within. For example, the status window contains the equipped item, health, and the number of cube parts.

Another feature of ncurses was color. Using ncurses allowed us to change the color of items within these windows. For example, if the user finds a yellow key on the ground and picks it up, it will display as yellow on the ground and in the inventory.

```
Room* Game::initPuzzleRoom11()
{
  Room* room = new Room("rooms/puzzle_11.room");
  room->setTeleporter(new Teleporter(12, 78, 6, 2, 12, COLOR_WHITE), 0);
  room->setDoor(new Door(14, 60, 9, 2, 23, 9121998, true, COLOR_WHITE), 0);
  room->setTeleporter(new Teleporter(15, 61, 9, 12, 12, COLOR_BLUE), 1); //

  //Trap COLUMN 1 (reading from left to right)
  room->setItem(new Trap(13, 63), 0);  //Top
  room->getItem(0)->setDamage(0);
  room->getItem(0)->setWeight(1000);

  room->setItem(new Trap(14, 63), 1);  //Middle
  room->getItem(1)->setDamage(1);
  room->getItem(1)->setWeight(1000);

  room->setItem(new Trap(15, 63), 2);  //Bottom
  room->getItem(2)->setDamage(3);
  room->getItem(2)->setWeight(1000);


  return room;
}
```

```
void Game::resolveDamage() //player walks on a trap
{
    Item **items = rooms[player.getCurrentRoom()]->getItems();
    for (int i = 0; i < rooms[player.getCurrentRoom()]->getMaxItems(); i++)
    {
        if (items[i] != NULL)
        {
            if (player.getXPos() == items[i]->getXPos() && player.getYPos() == items[i]->getYPos()) //if x and y value match
            {
                player.damageHP(items[i]->getDamage());
            }
        }
    }
}
```

The Trap Room is one of the 15 rooms that the player interacts with. To the left is a shortened version of the code used to initialize the room. We see the left most column of traps being initialized along with both teleporters (H) and the locked door.

What the player sees is the room displayed on the right. They interact and move around the world via the World Window which displays the rooms.

When the player takes damage, it resolves using the code at the bottom. The end result is lost health.

Features

- Each room was individually designed based on a theme or feature from another video game.
- The game is meant to be difficult and punishing, similar to the rogue-like genre and Dark Souls series.
- <Adventure Game> comes complete with an inventory for item management, a narrative bar to display world events and plot, a world window to interact with the room, and a status bar to display information related to the player.