Game Proposal: Outlaw’s Ascent

CPSC 427 – Video Game Programming

Team: XIV Studios (14)

Carina Bi (99145575)
Seongwon Choi (87274940)
Mike Wang (17474925)
Edric Antoine (97400501)
Clement To (66767849)

Story:

Briefly describe the overall game structure with a possible background story or motivation.
Focus on the gameplay elements of the game over the background story.

18-19th century America (the Wild West). After being outcast from society, you have resolved to become the greatest outlaw that has ever lived. However, the path to glory is arduous, as you must contend with the wilderness and threats from enemy gangs.

To achieve your goal, you must traverse a procedurally-generated world divided into many individual ‘cells’, each with a specific event (e.g. enemy encounters, loot, towns, NPCs). These ‘cells’ are linked together in a grid-like pattern, and you can move between adjacent cells. As an added challenge, if a cell contains an enemy encounter, you cannot leave that cell without killing all enemies first.

Our ambitious protagonist starts as a lousy bandit in the town of Dusty Ridge, the town where he was first sentenced and banished and labeled as a criminal. With your lousy pistol, our protagonist begins his journey to find the legendary outlaw "Silas Blackthorn", the leader of "The Crimson Vipers", a group that everyone fears. But to locate the man, our protagonist needs to find 3 lackeys that hold pieces to a map to the location of the gang. The four hideouts are named Dead Man's Rest, Whiskey Hollow, and Raven's Bluff.

The protagonist starts with the easiest, Dead Man's Rest but needs to cross the Cactus Flats first. The Cactus Flats is a harsh desert that contains a lot of dangerous wild animals and basic bandits. The boss of the Cactus Flats is a large wild animal that the protagonist must defeat. Reaching the first and weakest outpost Dead Man's Rest, the protagonist fights through bandits and outlaws to reach the first lackey of Silas and defeats him to get the first piece of the map.

The rest of his journeys are similar to that of the first outpost, the protagonist must cross environments such as Maple Mountains, a heavily forested mountain that contains forest bandits and forest animals, and Black Ridge Mines, an abandoned ore mine that contains experienced bandits and dangerous wild animals.

The remaining 2 outposts that the protagonist must reach are Whiskey Hollow (forest outpost), and Raven's Bluff (cave outpost) each with strong pseudo-legendary outlaws that work under Blackthorn.

Once the protagonist goes through all the outposts and finishes the map, the protagonist reaches The Crimson Serpent, the main headquarters for The Crimson Viper, all the enemies are extremely skilled outlaws who have proven themselves to work under Blackthorn's command. Fighting through the well-equipped and experienced bandits, the protagonist finally finds and defeats Blackthorn crowning himself as the greatest outlaw to exist.

Multiple levels correspond to his journey:
Level 1: Cactus Flats (desert)
Level 2: Dead Man's Rest (outpost 1)
Level 3: Maple Mountains (Forested Mountains)
Level 4: Whiskey Hollow (outpost 2)
Level 5: Black Ridge Mines (A mine)
Level 6: Raven's Bluff (outpost 3)
Level 7: The Crimson Vipers

Our minimum plan for levels is to implement at least two levels and one outpost. Time permitting, we will consider expanding the number of levels beyond the first outpost in accordance with the story.

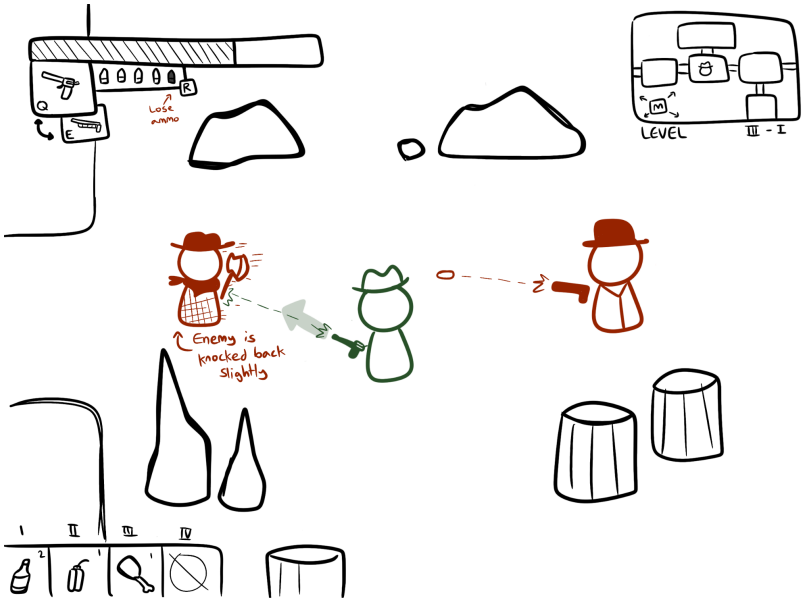
Similar to Binding of Isaac, where we have multiple levels, and each level has randomly generated cell placements, with different monsters, and different items to pick up and each level has a “mini” boss that the player needs to find. Once the boss is defeated they are rewarded with a bunch of loot and the option to proceed to the next level. Once the player reaches the final level and finds the boss “cell” they are met with the final boss - the legendary outlaw and defeating him makes you the greatest outlaw. The enemies and bosses will progressively be more difficult as you enter higher levels, i.e. Dusty Ridge Town will have very basic enemies whilst The Crimson Vipers will have advanced enemies that have complicated attack patterns and have much more HP. The pool of potential enemies and bosses will also be different depending on the levels the player is on.

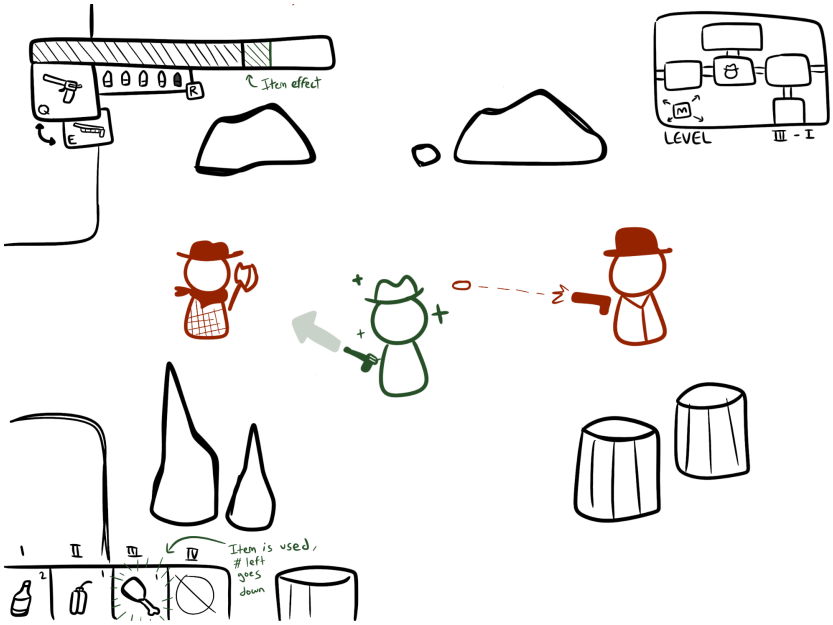
Scenes:

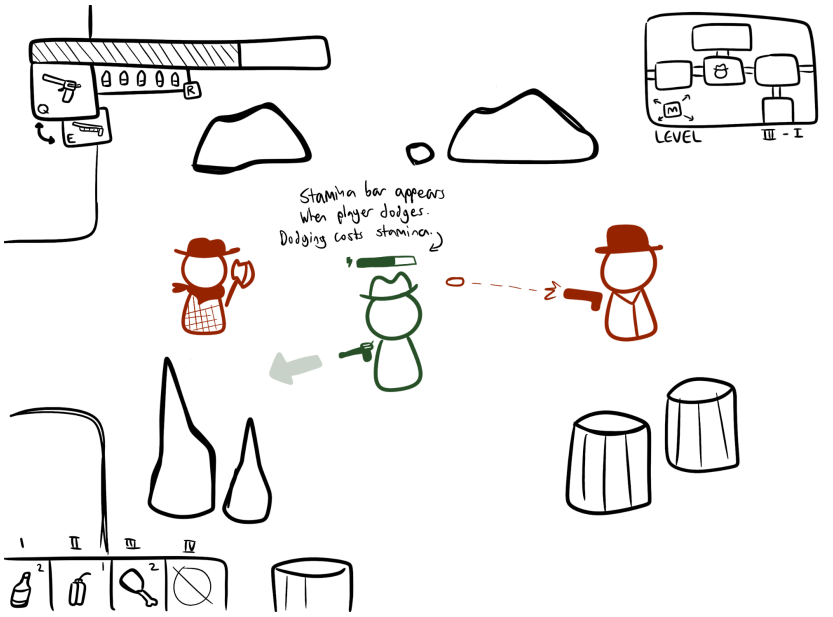
Produce basic, yet descriptive, sketches of the major game states (screens or scenes). These should be consistent with the game design elements, and help you assess the amount of work to be done. These should clearly show how players will interact with the game and what the outcomes of their interactions will be. For example, jumping onto platforms, shooting projectiles, enemy pathfinding or ‘seeing’ the player. This section is meant to demonstrate how the game will play and feeds into the technical and advanced technical element sections below.

If taking inspiration from other games, you can include annotated screenshots that capture the

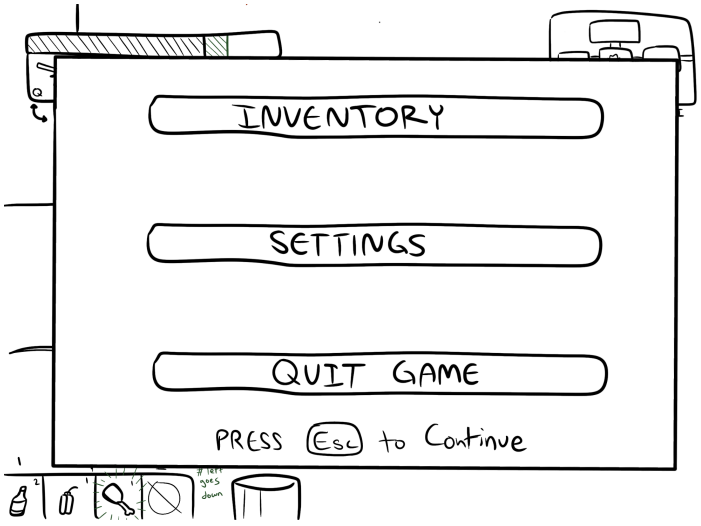
game play elements you are planning to copy.


Q
R
E
Lose ammo
M
LEVEL
III - I
Enemy is knocked back slightly
I
II
III
IV

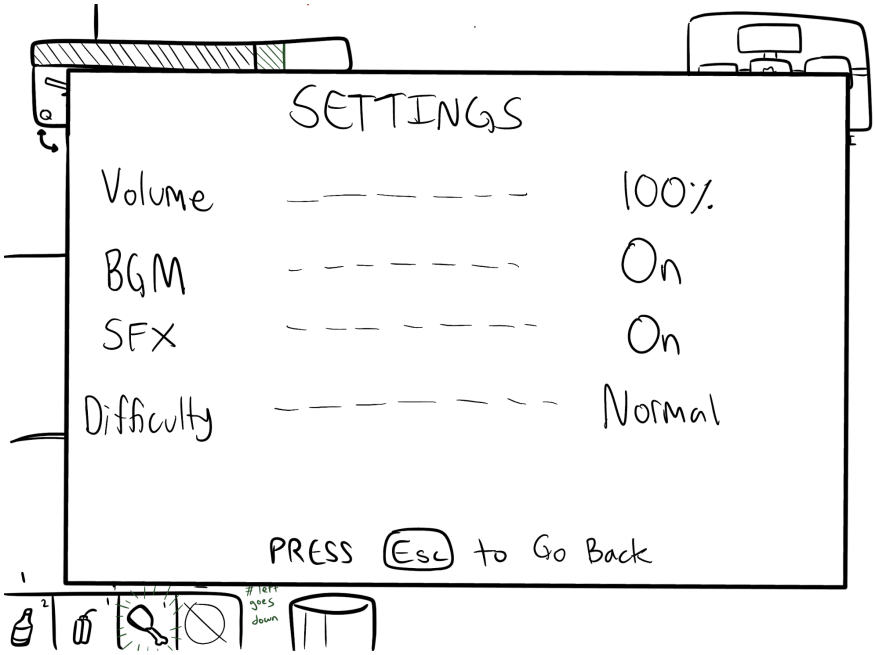

Q
R
E
Item effect
M
LEVEL
III - I
I
II
III
IV
Item is used, # left goes down

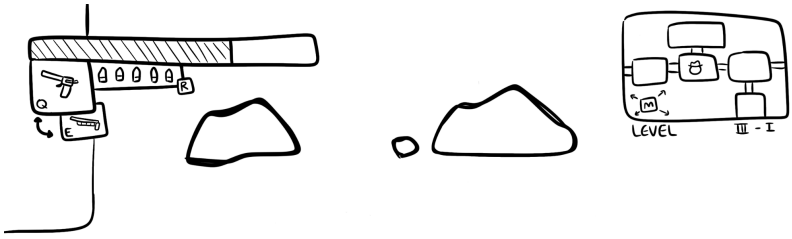

Q
R
E
M
LEVEL
III - I
Stamina bar appears when player dodges. Dodging costs stamina.
I
II
III
IV
2
1
2

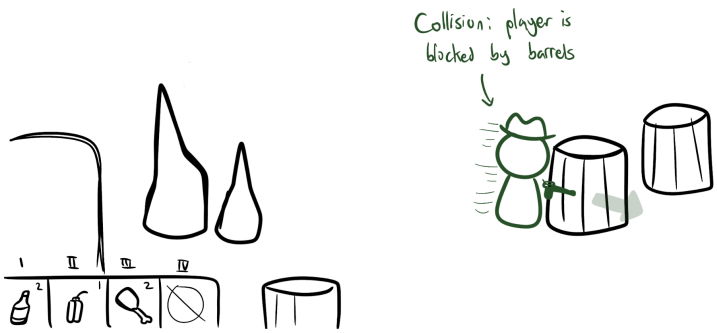

Q
R
E
M
LEVEL
III - I
Enemy pathfinding seeks & chases player
I
II
III
IV
2
1
2

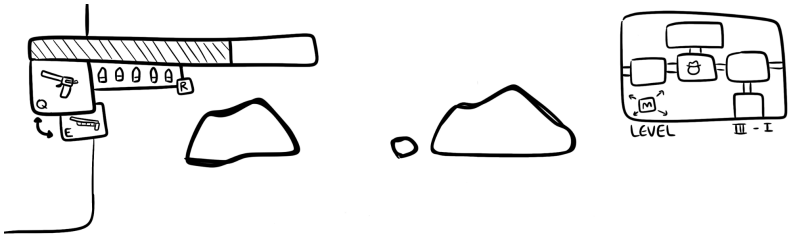

INVENTORY
SETTINGS
QUIT GAME
PRESS Esc to Continue


STATS
HP 390/500
STM 350
ATK 250
DEF 200
SPD 150
LUCK 50
WEAPON
DMG 300
RANGE 20
FIRE RATE 3
Revolver
AMMO 6
RELOAD 1 sec
ITEMS
LUCKY RABBIT'S FOOT
+10 LUCK
PRESS Esc TO GO BACK

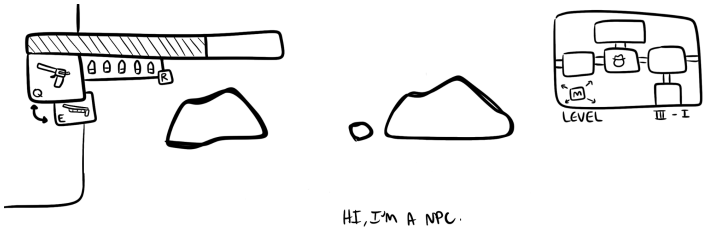

SETTINGS
Volume 100%
BGM On
SFX On
Difficulty Normal
PRESS Esc to Go Back

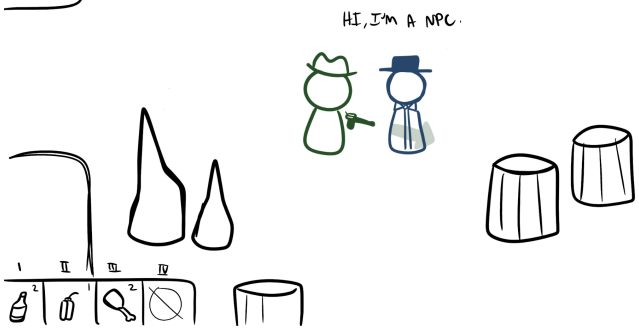

Q
R
E
M
LEVEL
III - I

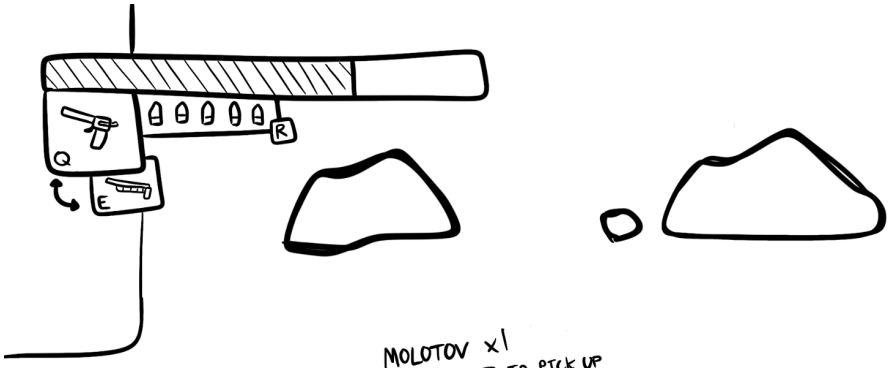

Collision: player is blocked by barrels
I
II
III
IV
2
1
2

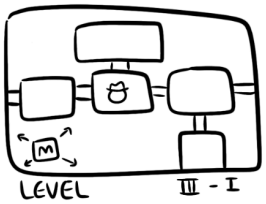

Q
R
E
M
LEVEL
III - I


SPACE
TALK
I
II
III
IV
2
1
2

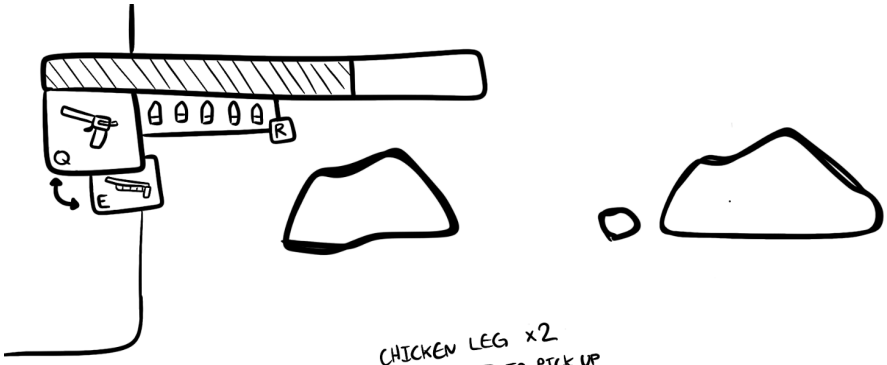

Q
R
E
M
LEVEL
III - I

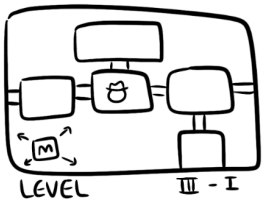

HI, I'm A NPC.
I
II
III
IV


Q
R
E


M
LEVEL
III - I


determines which hotbar key the item will go into
MOLOTOV x1
1 2 3 4 TO PICK UP
player standing over/in close proximity to item
I
II
III
IV

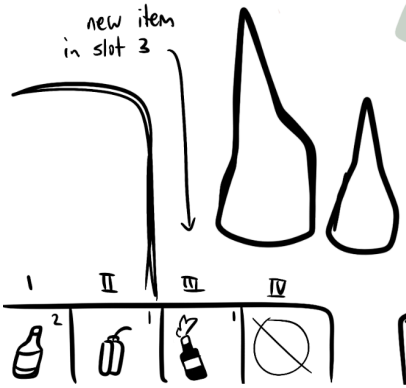


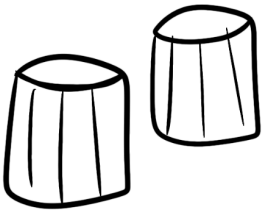


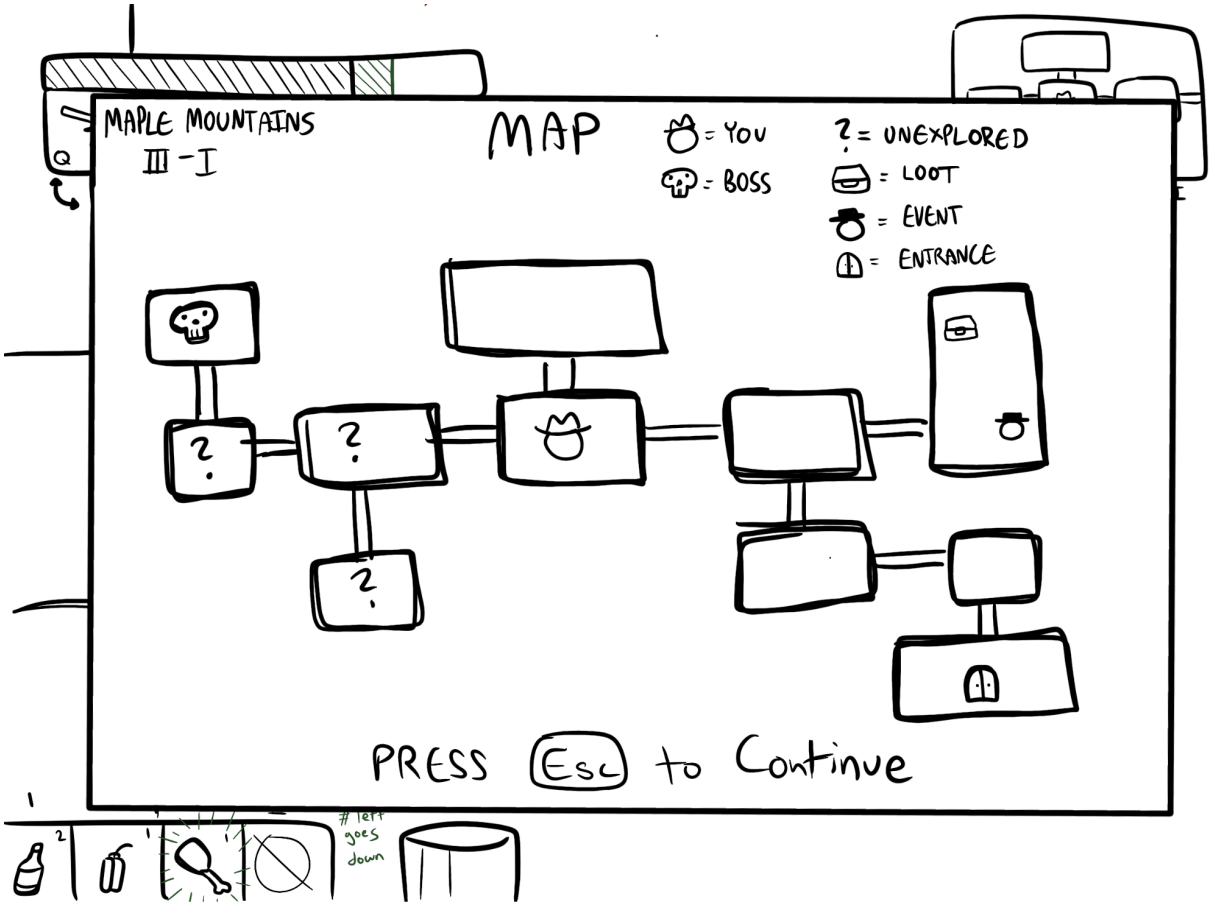
CHICKEN LEG x2
1 2 3 4 TO PICK UP

player pressed "3",
what was in slot 3
before is now dropped

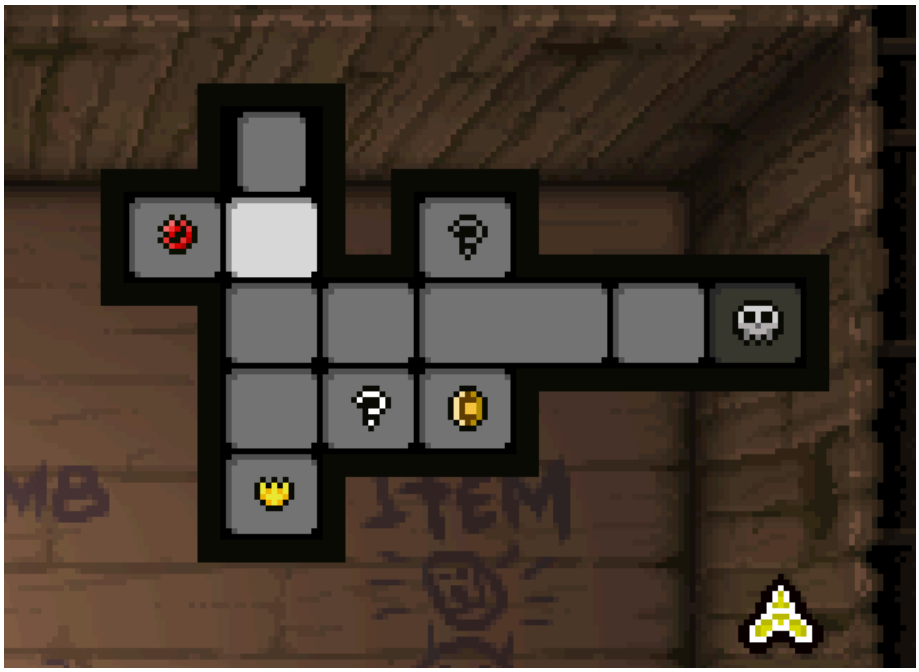
new item
in slot 3



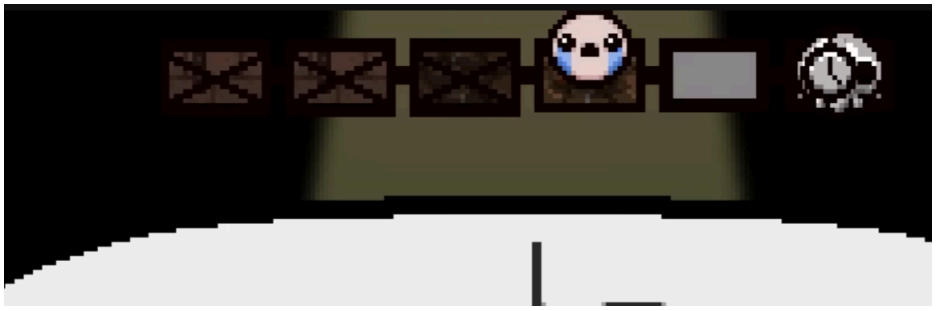




Inspiration: The Binding of Isaac



(Procedurally generated levels with a boss that must be defeated to proceed)
(The Binding of Isaac: Rebirth)



(different levels)
(The Binding of Isaac: Rebirth)

Technical Elements:

Identify how the game satisfies the core technical requirements: rendering; geometric/sprite/other assets; 2D geometry manipulation (transformation, collisions, etc.); gameplay logic/AI, physics.

Rendering: The game will display 2D sprites at a top-down angle to make the world look a little more 3D without actually being 3D. Constant visual effects such as shadows will be used to add depth to the visuals in general, and specific moments such as very powerful attacks can be highlighted through the use of effects like blur and zoom.

Assets: Sprites (which are generally used for moving objects and characters) and tilesets will make up most of the visual assets of the game. The game will also have audio such as background music and various sound effects.

2D Geometry Manipulation: Transformation is used for the movement of most entities on screen, from characters to bullets to background elements. Most entities will also be able to collide with walls, obstacles, or hazards which will prevent movement, and of course, collision with bullets or attack hitboxes will deal damage to the colliding entity.

Gameplay Logic: Player movement, attacks, and the action of equipping items are bound to keys (ie. WASD) or mouse movement. During exploration in the open world, the player may interact with NPCs who will slowly uncover the secrets behind the story.

AI: Melee-based enemies will use pathfinding to move towards and attack the player while avoiding obstacles and hazards. Ranged enemies will also use pathfinding, but try to stay a certain distance away from the player and in a position where they can shoot directly at the player. Bosses or stronger enemies will use differing attacks depending on their proximity to the player (e.g., melee attacks when the player is nearby and ranged / AOE attacks when they are far away). Passive entities (NPCs, passive animals, etc.) will have more ‘random’ movement with no particular goal but might run away if you shoot at or near them.

Physics: Entities will stop movement when they interact with a wall or similar obstacle. Being hit by an attack will also knock them back slightly in the direction they were hit. Using guns and other weapons, as well as certain environmental interactions, will also create particle effects such as smoke and sparks, which are affected by physics.

Sound: Mobs such as wild animals, town sheriffs, and other outlaws will have their own sound effects when they are within a close enough distance to the player. Different sorts of music will play depending on the current state of the game (e.g. battle and overworld music that is different based on the area the player is in, peaceful music for certain events, etc.)

Advanced Technical Elements:

List the more advanced and additional technical elements you intend to include in the game prioritized on likelihood of inclusion. Describe the impact on the gameplay in the event of skipping each of the features and propose an alternative.

- Procedural generation: We plan to have levels be procedurally generated. If we do not include this feature, we will instead need to hard-code the layouts of each level. The game would still be playable, but the replayability of levels would be reduced, and given that the game is similar to a roguelike, this would have a negative impact on the gameplay.

Devices:

We plan on supporting keyboard controls first. If we have time, we will also consider supporting gamepads as well.

Keyboard control mapping:

- W/S/A/D = move up/down/left/right (holding multiple keys allows for diagonal movement), navigate within a menu
- Mouse move = aim equipped weapon
- Mouse left-click = attack with equipped weapon
- Mouse right-click OR R key = reload equipped weapon
- 1/2/3/4 keys = use consumable item
- Mouse scroll wheel, Q/E keys = change equipped weapon
- Space bar = Interact with object or NPC, ‘confirm’ in menus, advance dialog
- Shift = dodge
- M key = open map
- Tab key = open inventory
- Esc = open / close game menu, ‘cancel’ in menus
- F key = toggle fps

Tools:

We will stick with the base libraries and tools (C++/OpenGL) for now.

(Add additional libraries used here…)

Team management:

Identify how you will assign and track tasks and describe the internal deadlines and policies you will use to meet the goals of each milestone.

We will use an agile software development method for this project. To track task status, we will use a Kanban board on GitHub, where we are able to divide tasks into categories including “Backlog”, “Ready”, “In Progress”, “In Review”, and “Done”. To assign tasks, we will give each task a number of story points corresponding to the expected work required, and attempt to distribute story points evenly among group members. We will also try to allow group members to work on tasks of the most interest to them if possible.

In order to meet the goals of each milestone, we will set deadlines for each week during our weekly meeting (Mondays). We will try to have deadlines on Saturday, which will give us time to discuss the previous week’s work during the next weekly meeting or on Sunday if needed. We will also have short mid-week check-ins on Wednesdays to update each other on the progress of the tasks: any blockers, what’s more difficult than usual, etc. If a group member is not able to meet a deadline, we will use our extra time on Sundays or meeting times to discuss and catch up with content that may have not been finished within time.

Development Plan:

Provide a list of tasks that your team will work on for each of the weekly deadlines. Account for some testing time and potential delays, as well as describing alternative options (plan B). Include all the major features you plan on implementing (no code).

List of advanced features: swarm behavior (20), enemy group behavior (20), reloadability (10), camera controls (10), basic physics (10), simple pathfinding? (10), Advanced decision-making (20)

Milestone 1: Skeletal Game

Week 1

- Implement the ECS
- List of entities: player, one enemy, one type of obstacle (cactus)
- Components: health, speed, position
- Movement system: Have a basic player that can move around (WASD)
    - Viewport should follow player movement
- Rendering system: Have a basic cell with various features (obstacles? water?)
- Create the testing plan and bug list (Google Sheets)

Week 2

- Have an enemy entity that does something (random or hard-coded ex. Moves to the right)
- Basic collision logic between players and walls (gamespace boundaries)
- Have a basic hard-coded level with various cells, to test the map concept: includes enemies, items…
- If we are happy with the concept, we will consider implementing an algorithm to procedurally generate levels. This will also depend on how much extra time we have.
- Think up more concrete designs of what we want our minibosses/final boss to look like and do
- Implement attack functionality of the player in one direction (no ability to aim yet).
- incorporate linear interpolation for the bullets from player attacks

Milestone 2: Minimal Playability

Week 1

- Implement equip, reload functionality of the player
- Implement static sprites for entities
- Implement mouse interactions for: map, F key, inventory
- Implement the ability to aim with the mouse
- Implemented more complicated AI logic for enemies: (simple pathfinding)
    - Vision system for enemies: they start chasing you when you are within a certain range from them, signifying start of battle

Week 2

- Have a procedurally generated level with some randomness in cell layout and the contents of each cell (time permitting)
- Animate sprites (waving cactus, player movement)
- Render sprites for dropped weapons and items
- Implement a UI for inventory management
- Create some examples of weapons (revolver, shotgun) consumables (dynamite, healing item), and passive items
- Implement Esc = open / close game menu, ‘cancel’ in menus
- Implement basic tutorial level

Milestone 3: Playability

Week 1

- Implement persistency - game should autosave whenever you complete a cell
- Have multiple (procedurally generated?) levels available for the player to progress through
- Extend enemy AI to show enemy group behavior or advanced decision making

Week 2

- Implement at least one boss
- Review memory management & performance
- Implement a start screen

Milestone 4: Final Game

Week 1

- Implement all levels in game, player should be able to play through story from start to finish
- Implement particle effects for muzzle flash, explosions, blood
- Implement SFX for player, enemy interactions
- Fix most critical (P1-2) bugs
- Playtest and balance the game

Week 2

- Implement music (time permitting)
- Fix bugs
- More playtesting and game balance