Core

- [x] Draw Map
- [x] Input, Player Moves in Map
- [x] Player Collides with Walls
- [x] Entry & Exit
- [x] Traverse to the next level
- [x] Map Rooms
- [x] Data logs
- [x] Player and Enemy Attributes
- [x] Static Enemies, participate in combat
- [x] Enemies move when in range
- [x] Treasure Chests
- [x] Traps
- [x] 10 different maps

Evaluation

- [ ] Clean, organized code
- [ ] Naming Conventions
- [ ] Good partitioning of classes and methods. Make sure your code resides where it should.
- [ ] Use of suitable data structures

Advanced Features

Easy

- [x] Prerequisites: Add requirements for leaving a level (such as you must kill all the enemies on the playing field, find a key, etc.)

- [x] Console Colors: Use colors within the console to denote different entities
- [ ] SFX: Add at least 2 sound effects
- [ ] Progression: Add a progression system so that enemies will become stronger from level to level
- [x] Healing Potions: add healing potions that can be used inside \ outside of battle.
- [x] Smooth Refresh: make the frame refresh smooth
- [x] Valid Random Enemy Locations: make enemies spawn in random (and valid!) locations on the map

Medium

- [ ] Colored HUD: create a HUD to present game data, such as the player's HP, gold count, EXP, etc. Hud will replace the easy Data Feature. You must use console colors for the different attributes
- [ ] Options Menu: create an options menu that can set various gameplay elements:
    - [ ] Choose the player's avatar
    - [ ] Choose the enemy's avatar
    - [ ] Difficulty level (only if chosen Progression)
    - [ ] 2 more valid options of your choice
- [x] Inventory: Create an inventory system: the player can collect at least two items and use them in the game world in some way (keys, weapons, armor, etc.)
- [x] Doors: insert doors into your game. The player can open doors with keys \ levers \ simple interaction. Doors provide a way to enter rooms within a level.
- [ ] File: read the maps from a file\files. Cannot be done with Procedural Levels.

Hard

- [ ] Big Enemies: Create enemies bigger than one tile
- [ ] Save and Load system: each map progression is autosaved. Make sure you save the player progression, items, etc.
- [ ] Asymmetric Maps: Make the maps asymmetrical
- [x] Combat System: Make an elaborate combat system that uses:
    - [x] Damage Reduction

- [x] Evasion
- [x] Critical hits
- [x] Hit Chance
- [x] You may add more parameters to your liking
- [x] The player must have more than 4 options when entering combat

- [ ] Shop: Create shops that sell items, boosts, and potions either within the level or between levels. Add currency that can be dropped from enemies or found in treasure chests.
- [x] Real-Time: Make the game real-time (the game refreshes without waiting for player input)
- [ ] Procedural Levels: Create the levels procedurally. This includes the map layouts, enemies, treasure chests, traps, etc. The game is now a roguelike, where the player can play an indefinite amount of maps but when he dies the game is over. Must choose Progression to implement this.
- [ ] Menu System: there's a Main Menu that can be used to pick a new game, see credits, change options, and Load (if chosen). Must pick the Options feature to choose this. The menu must be traversed with realtime input.

Deliverables

- [x] This checklist
- [x] Video of gameplay
- [x] .zip \ .rar of the entire project structure
- [x] Tutorial, either a README file of part of the gmae

Custom Features

Features not listed in the brief you decided to add cause it's f'ing cool.

- [x] Some feature A Text to speech
- [x] Some feature B MIDI Player
- [ ] …