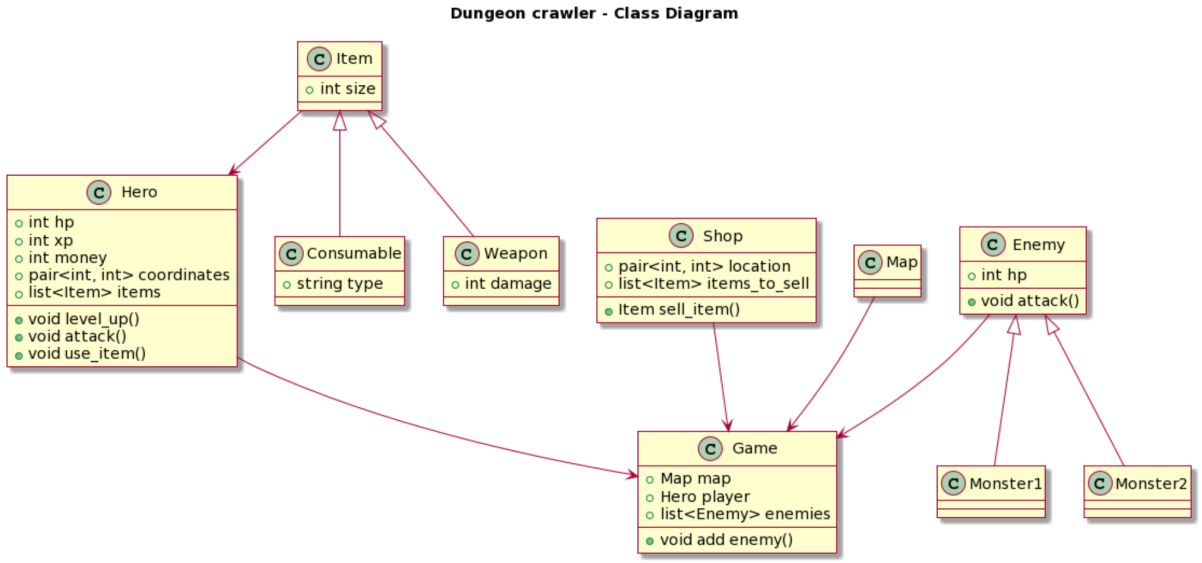
Scope of the work: what features and functionalities will be implemented, how is the program used, and how does it work:

- Simple 2D graphics
    - Simple 2D graphics using SDL.
- Moving through corridors and rooms.
    - Yes.
- Combat between the player and different types of monsters
    - Real-time combat.
- Collectibles which can be used later on (weapons, potions, etc.)
    - Different types of items.
- Some sort of progression (winning & losing conditions)
    - Lose by dying.
    - Win by killing the final boss.
- Randomly generated dungeons
    - Pre-defined dungeons with small differences depending on few randomized variables.
- Randomly generated monsters and items
    - Pre-defined monsters with small differences depending on few randomized variables.
    - Predefined items that will drop randomly depending on a few variables.
- Character leveling (skills & abilities)
    - Killing monsters grants you experience points. Upon leveling up skills and abilities are increased (player stats).
- Ability to craft / modify items
    - On certain coordinates, you can for example upgrade some items.
- Quests or other events
    - Finishing quests grants some kind of currency in game that can be used to buy items.
- Other kinds of non-player characters (shopkeepers, allied combat, etc.)
    - You can use the in game currency to buy items from the shopkeeper. You can also sell items to the shopkeeper.
- Additional UI elements (dungeon map, interactive inventory, etc.)
    - Interactive inventory, where you can drop or equip/unequip items.
- Sound effects
    - BOOMBOOM

The high-level structure of the software: main modules, main classes (according to current understanding):



The planned use of external libraries:

SDL (Simple DirectMedia Layer):
Cross-Platform Audio and Input. SDL can be used for handling audio, keyboard, and mouse input, ensuring cross-platform compatibility and delivering an immersive gaming experience.

JSON Libraries (e.g., JSON for Modern C++):
Data Serialization. To save and load game data, including character attributes, item statistics, and game progress, JSON libraries facilitate data serialization and deserialization.

Random Number Libraries (e.g., Random123):
Randomization: For procedural generation of dungeons, loot, and events, high-quality random number libraries ensure the unpredictability and uniqueness of each gameplay session.

Maybe if needed:

Image and Texture Libraries (e.g., STB Image or FreeImage):
Image Loading. For handling image assets and textures, these libraries simplify the loading and manipulation of image files for in-game visuals.

Boost C++ Libraries:

Threading and Networking. Boost offers a comprehensive set of libraries for various functionalities, including multithreading and networking. This can be essential for implementing multiplayer support and optimizing game performance.

Division of work and responsibilities between the group members:

We will share work evenly. Each member is a chairman on their own week. Chairman has the final responsibility of things getting done. Chairman cycles each week. In our avant-garde approach to project management, we will uphold the sacred principle of "equal opportunity chaos."

Planned schedule and milestones before the final deadline of the project:

The optimal milestones are once a week meetings where we see the current state of the project and think what to do next. The meetings will be held in a room adorned with motivational posters featuring majestic eagles soaring above mountains and phrases like "Unleash Your Inner Ninja" and "Achieve the Impossible!". We have around six weeks to finish the project and so we have six milestones planned. The content of each milestone depends on the previous progress. With this impeccable plan, we are guaranteed to achieve unprecedented success and bask in the glory of our groundbreaking project.