**Software Design/Briefing**

As my final project, I’ve decided to construct a game/puzzle in which the user will take on the role of Batman. After careful thought I settled on Batman since he’s my favorite super hero in the super hero universe and as you can imagine I’m a big comic book fan.

The user’s ultimate goal is to find the Joker. Below is a synopsis of the game “plot”:

***The Joker has just escaped Batman’s custody. It’s now it’s Batman’s job to find him before it’s too late. If Batman doesn’t find the Joker, the Joker will unleash his toxic nerve gas on the people of Gotham City. Along the way, Batman will need to collect certain gadget’s and may encounter other puzzles/enemies.***

***Batman must find him by his 8th move, or else the Joker will succeed in his plan to release the toxin. Once found, however, the Joker will not go quietly.***

As the author, I’ve decided to create a **linear** **linked list** to simulate Batman’s gadget/utility belt. This structure contains the following data members/functions:



Below is a diagram on how the structure is constructed:



I made a conscious decision to not include a “remove” function since technically speaking, there is no reason why Batman would want to remove a Gadget once it’s in his possession. These gadget’s will be required if he’s to move around the environment which is a linked node structure.

Additionally I created an inheritance hierarchy for the DCCharacters (Batman and Joker) with inherit set and get methods.

Below is a diagram of the inheritance tree:



Finally, I created a diagram of the linked structure which details the levels and how the user will traverse the structure:



**Reflections**

Throughout the course of incrementally developing the program I found I had several memory leaks. In particularly when developing the spaces. I found using pointer’s with dynamic assignments in the space class and derived classes was causing memory leaks. I was unsure why so instead I declared a static variable for the various objects I was using, in particular I was having trouble with the space classes that instantiated a string in the constructor. Not entirely sure why this was happening.

In addition, I found several logic errors which I later corrected in my main program during the menu options.

I also borrowed the logic concepts from our assignment 3 of fantasy combat for when Batman faces the Joker and there’s a battle. Much of the same logic in the sense of attack, defense, health, and so forth.

Each character will have a max offense (Batman top offense is 25) and Joker’s top offense is 15 while having a slightly larger health.

**Test Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case** | **Input Values** | **Driver Functions** | **Expected Outcome** | **Observed Outcomes** |
| Ensure there were no memory leaks in the creation of various dynamic space classes and dc character classes | Various dynamic allocations with pointers of objects including gadget container and utilities functions | Standard instanitation in main | No memory leaks | Memory leaks were observed when using valgrind.  This was corrected in the .hpp and .cpp files by using static objects when applicable |
| Ensure the structure is properly constructured and accessible to the user including restriction of visiting or advancing to other pointers within the structure without completing the necessary levels | Ensure the user cannot access the Batcave and the Arkham Roof without completing the other levels first | While loops and Boolean variables | For the program to warn the user he is not allowed to visit this part of the structure without completing the necessary levels | Program warned the user he/she must complete the necessary levels |
| Proper pointer assignments within the structure and no pointers were lost or assigned as null | N/A | While loops and Boolean if statements | For the program to polymorphically tell me what levels I’m currently in | Program reacted correct and only used DCCharacter and Space pointers throughout the program and each time it printed the correct “Space” name and “Character” name |
| Ensure the program is only accepting numbers when applicable and not character | Integers | While statements and my utilities class | For the program to warn the user of incorrect input | Program did warn the user of incorrect input |