Analyzing the design #2
CS 202: Programming Systems

Alves Silva, Otavio Augusto
PSU ID: 902840168

1. Design effectiveness

The design was able to give me a big idea about the hierarchic of my classes as well as responsibilities and jobs of each one in the purpose of the program. Therefore, it made the job helping not starting programing for the zero and having none idea about how I could start the code.

3. OOP Results

The program was able to achieve OOP design and abstract base class requirements. The classes are a group of specialized other classes using the concept of inheritance as the method of “is a” object and “has a” object. For example, the class level one is a specialized class of the abstract base class level and inherits the base class data members and member functions and the class level two is a specialized class of the abstract base class level. Therefore, the usage of abstract base class was achieved.

Another fact about the OOP requirements, the program used a node class object that was totally able to implement the data structure used(BST) in the program as well as the graph node and vertex classes that were used in the data structure(Graph).

The abstract base class level was successfully used by the level one(BST) and level two(Graph) achieving the polymorphisms that the abstract base class and dynamic binding can provided.

4. Changes in the design

The design have big changes, but the data structure part of the design wasn’t chose and constructed. Therefore, the class code is basically a binary tree, which has a root node, having the functionality to implement the maze particularities.

The amount of function members in the graph class was bigger than the expected because it had more than more than just a job to insert a new path. Therefore, more functions were necessary to create this approach, the class graph has to execute the random algorithm code to insert the job on each node in the graph.

The class node wasn’t used in this program #2 instead of that, the job and the responsibility of the code class on the last program passed to the data structure in each level. Therefore, level one and level two have their own code with different algorithms.

6. Code efficiency

The random algorithm code in each level was able to create a new random maze, which has different length and number of paths, each time a new game is started. The player was totally able to walk through the maze without any problem, but he needs lives for that. The paths were capable to interact with the player giving points, taking lives and showing the results. However, the traps, prizes and maze end will always be in the same point. The traps will be always an odd path value, the prize will always be an even path value and the maze end will be always the biggest path value in the maze.

Another fact in the second level of the maze is that the graph isn’t a directed mapped graph. There is two links between each edge. This generates a great probability of that the algorithm generates a little amount of dead points in the graph. However, even with this problem the main purpose of the program was achieved with success.