Assignment 1: Design

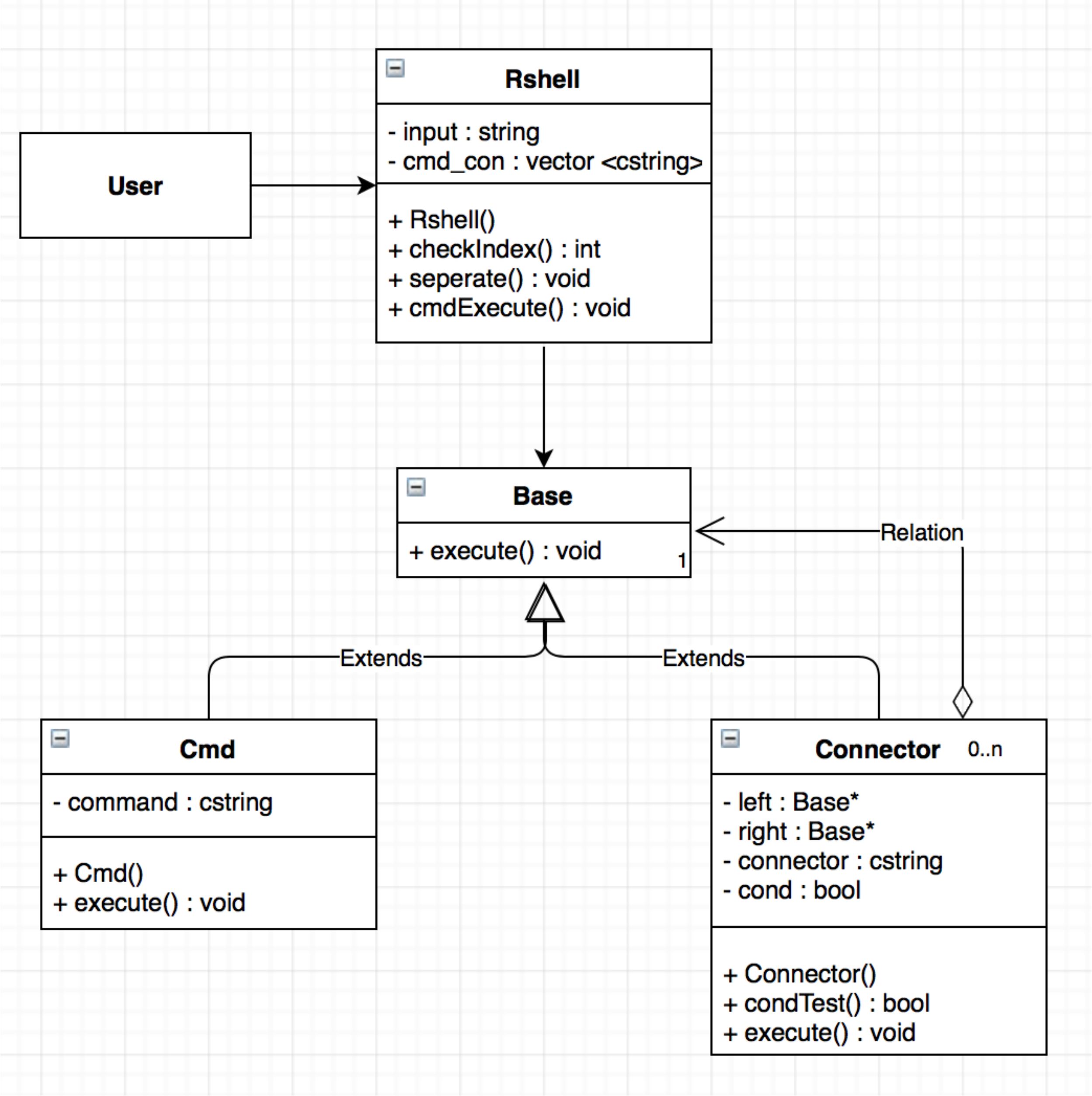
January 29, 2018
Winter 2018

Sili Guo 861220955
Zelai Wang 862017188

Introduction

Assignment 2 is a program that execute a command shell program in C++. The program can be separated to three parts. It first output a “$” sign in terminal; then the program read one-line command that user input; and finally, it will not end until user enter the sensitive keyword. In specific, the first part requires a function in main to output the “$” sign; the second part requires an interface class, and two separate classes to generate the execution function for command and connector; the last part requires a folk function to prohibit the program exiting after each command line.

UML



Classes Groups

Rshell: Rshell class is a interface class that will generate the command that user input and separate to command and connector to use the correct execute function.

- This class contains a string variable that stores command that user input, and a cstring vector that contains each command and connector.
- This class contains a constructor, a function to check where each connector located, a function that separate the commands with connectors and store them into a vector, and a function to decide which execute function should be used for command and connector.

Base: Base class is an abstract class that only contains a virtual function execute() for commands and connectors.

Cmd: Cmd is a leaf of Base class, it inherits from Base and override execute() function for command.

- This class contains a variable in cstring that store the command that has been separated by separate() function in Rshell.
- This class contains a constructor and a void function execute() that execute command.

Connector: Connector class is a composite class that inherits and composite from Base class. It use the tree structure to control the order of the execution of commands from left to right, and the different connectors will decide whether the command on the right side will execute or not.

- This class contains two Base pointers that use the structure of binary tree to order the command and connector from left to root to right, and a cstring variable that indicates which connectors ("&&", "||", or ";") are currently been stored; it also contains a bool variable cond that indicates if the command on the left child execute successful or not which can be used to determine the execution of command on the right child.
- This class contains a constructor, a cstring function that will test if the previous command execute successful or not and return a bool value; the execute() function will based on which connector it is to decide if the command on right child will execute or not.

Coding Strategy

Basically, coding will be separated to two parts, so that Sili and Zelai can work on their own after class and will put together in the end.

Zelai Wang will be charge in the interface part, which contains the Rshell class. Major tasks are:

- Prompt from user input, then separate and convert the string to command and connectors in cstring, and store them in vectors.
- Write the cmdExecute() function which will call the correct execute() function for commands and connectors.

Sili Guo will be charge in the Base, Cmd, and Connectors classes. Major tasks are:

- The tree structure that will place command and connectors in order.
- Execute the command line

- In execute() function in Connector class, based on different connectors and if the previous command execute successfully, call the execute() function for left child or right child from Cmd class.

Group meeting will held on each Monday, Wednesday and Friday to discuss the current progress and may change the plan based on the work have done. Current plan is to finish each person's tasks before Monday, February 2, 2018, so that there will be enough time to put everything together and do the test cases.

Roadblocks

- **Deadline Problem**

  We both have our own habits about when to start and finish the assignment, but different schedule may waste both of our time; so we will separate assignments to several steps and make a schedule acceptable for us. We will change the schedule on group meeting if it is needed

- **Coding Problem**

  We have never learned how to call the commands in C++, that might be a problem for us to write code successfully; and the solution is to search some examples from internet and be familiar with it.

- **Teamwork Problem**

  Since this is the first time we work together, our thought may be different and the code we write may not work well with the other one's code; to solve this problem, we plan to hold three group meeting each week to communicate with what we have done, and what is need to be changed later.

  Also, it may be hard for us to understand each other's code, and it may take long time when we try to put things together; so we will first discuss and agree on the comment we write during coding to save time in the final step