# Assignment 1: Design

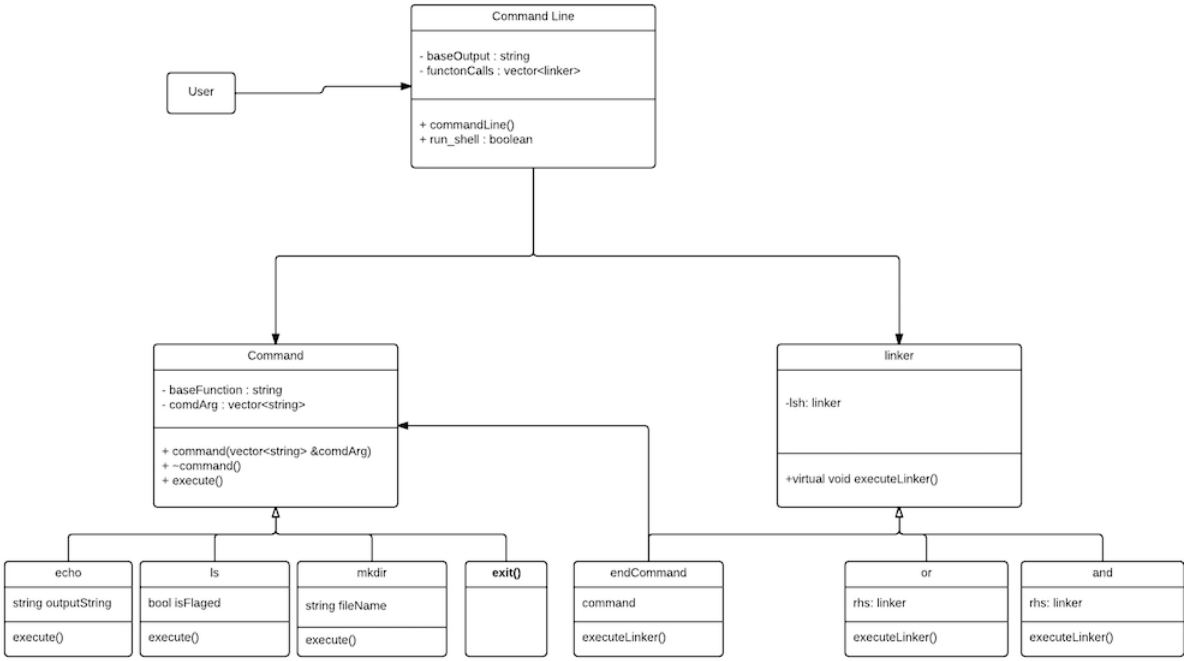
James Luo, Kelly Luu

October 20 2017 Fall 2017

## 1 Introduction

The purpose of a bash shell is for any user regardless of skill level to be able to navigate their computer's hard drive and perform administrative tasks such as "mkdir","echo", "ls". This program will allow the user to perform tasks much faster compared to just using the graphical interface built by the operating system.
The design that we are going with is like a tree of classes in which certain classes, like the output line in which the program will take in data and output data, will contain objects of different classes such as a function class and a directory class.

## 2 Diagram




## 3 Classes

1. Command Line class
   command line class responsible for creating the command line and taking input from the user

   - vector of functions $->$ responsible for keeping track of linkers in the current user command

   - runShell function $->$ a bool function that prints the output message and takes input from user.
     should return true if the user does not enter exit when user enters exit then exit the program.

2. function class
   this super class and its inherited classes is responsible for building the functions and then having a function that executes it along with returning a pair that contains a flag to determine if it was completed without error and the string output.

   - executefunction() $->$ responsible for determining what functions to run and taking in a vector of strings that will either store a message code and a output string
   - string variable to store the command being called
   - a vector used to store arguments being used

3. linker Class
   This super class contains class that have other linkers and its job is to determine which functions should be executed based on the arguments that get passed in.

   executeLinker $->$ its job is to call the linkers functions inside it. This can range from just calling the function class or calling another linker class

# 4 coding strategy

We will divide the code in which we will first create a bare bone model of functions. Then we will split the tasks in half base on what type of functions they are. In terms of building the linker class, we will work together to come up with the super class and then split off into working on the sub classes

# 5 road blocks

When the code is being stuck at a stalemate. We should find ways to rebuild the function. This can range from adding another class or new data member and see how we can change our design. Another way to do it is to increase the layer of abstraction in which more of the functions get called in from the lower base classes so that we do not need to worry that much about the higher level classes.