Design Document: dog

Alan Li

1 Goals

The goal of this program is to perform a function similar to that of the basic “cat” command. The basic cat command is able to take a lot of flags, but our program will not need to support this. Files will be read in the order they are given and copied, then changed to standard output.

We will also need to account for when “-” is given as an input. We will have to use standard input for the file. When no files are given, we need to copy standard input to standard output until there are no more inputs. The program should throw standard errors and handle file errors as well.

2 Design

The program first reads for arguments and then attempts to read and process it. It will then go through each file one by one and write to the standard output.

2.1 Handling arguments

Just like cat, dog will have to handle arguments such as files. When given just a “ “, the program will do the same as “-” in which it echoes the user’s input in the command prompt until they press ctrl+d. Files will be read in the standard input and with a buffer, we will then output to standard out.

Input: argument count, arguments

Algorithm 1: checks for user's arguments and count to determine what to do

```
If (arg count = 1: user's first input is " "){
    similar to a "-" case; echo user input to stdout
    Increase incrementer to go to next argument
}
Create Incrementer
While(incrementer less than argument cont) {
    (go through arguments){
        if("-" is entered) {
            Special case like " "; echo input to stdout
        }
        if(open = 1 & read = -1; this is a directory) {
            throw error
            Increase incrementer to go to next argument
        }else{
            Warn if invalid file (when read = -1)
            Else If valid file, read and put to standard output
            Increase incrementer to go to next argument
        }
}
```

2.2 Printing

We will need to handle what to do when given input. This means we first have to check arguments and determine what to do after. We have conditionals to check what the user has inputted or would like to do. From that, we are able to use functions to print file to stdout, or if - is given, use standard input for that(like cat).

Algorithm 2: Printing from file
Input: user input, file arguments   Output: stdout

```
If (read(file) != -1 ) {
    Read file into buffer; then write from buffer to stdout
}
```

Algorithm 3: Printing from stdin/ user types in " ", "-"
Input: User input   Output: echo to stdout

```
Determine if user entered "-" or " "
While (user has not pressed ctrl+d ; read!=0){
    Write to stdout
    Read next input
}
```