Network Programming Assignment, Design Document: Q1

Student Name	**Student ID**
Aaditya Kumar	2017A3PS0332P
Anirudh Buvanesh	2016B4A70614P
Ashish Kumar	2016B4A70636P

A Linux like shell had to be created with support for '||' and '|||' piping functions apart from running standard linux commands with input and output redirections and a user defined shortcut-cut (SC) mode.

1. **Data Structures**

Sl No.	Name	Purpose	Definition
a.	commandGroup	This is the primary structure that stores the parsed input and is passed along to the executing pipelines. It is a Linked List implementation.	typedef struct commandGroup { char *command[3]; char *argv[3][MAX_ARGS]; int pipeType; bool outputRedirect[3]; bool outputAppend[3]; char *inputFilename[3]; char *outputFilename[3]; bool isBackground; struct commandGroup* next; } commandGroup;
2.	pipeline	This structure is used to execute a commandGroup along various pipelines it may contain.	typedef struct pipeline{ commandGroup *firstCommand; int numberOfCommands; }pipeline;

   a. **commandGroup:** This data structure is designed to handle upto a three way piping of a command. On encountering any sort of piping, the current node of the commandGroup linked list will point to the parsed form of the input on the right of the pipe till another pipe is encountered and this process will keep repeating until

there are no more pipes at the end. The pointer to the next node is stored in the **next** field. If there are *n - 1* pipes, there will be *n* nodes in this linked list.
The **pipeType** field in each node will give us an indication of what the current node pipes its command output towards. (0-> no pipes or STDOUT , 1->single pipe, 2-> double pipe, 3-> triple pipe).
For handling double and triple pipes the output from the previous command is buffered and replicated to temporary pipes which the present command reads from.

The **command** and **argv** fields are string arrays containing the parsed commands and their respective arguments. Likewise, the **inputRedirect**, **outputRedirect** and **outputAppend** fields indicate the IP/OP natures of upto 3 piped commands. The **inputFilename** and **outputFilename** store the file names for I/O. The **isBackground** field will indicate whether a process runs in the foreground or background.

b. pipeline: This structure encapsulates the job in a linked list structure with fields as mentioned above.

2. Input Parsing

a. Key Assumptions

i. There are no nested pipes.
ii. The '&' sign to indicate a background process will always be at the end of an input line.
iii. In any input stream, the first continuous set of characters, separated by whistles space from other sets of characters, is considered to be the base command.
iv. The arguments to a command are any other continuous streams of characters which begin with a '-' or do not begin with '>', '<' or '&'.
v. Extensive error testing has not been done since it is a demo model and the inputs are expected to be syntactically correct to a reasonable extent.

b. All parsing utilities can be found in the *parseInput.c* file.