Void cd(command){

If(argc>2):

Perror;

If(argc == 1):

Print getcwd();

If path exists:

chdir(path);

else:

perror;

}

Void clr(){

Print “\033[H\033[2J”

}

Void dir(cmd){

If(argc>2):

Perror;

If(argc==1):  
 strcat(path, getcwd());

else:

strcat(path, arg[1]);

If path exists:

dir = opendir(path);

if dir == NULL:

perror;

make a while loop with readdir(dir):

print content;

closedir(dir);

}

Void environ(){

make a while loop with environ[index] != NULL:

print(environ[index]);

index += 1;

}

Void echo(cmd){

If(argc == 1):

Print a line of spaces;

else:

make a for loop with argc && arg[index] != NULL:

printf(arg[index]);

print(“\n”);

}

Void pause(){

while(getchar() != ‘\n’){ //until user hits enter

;

}

}

Void quit(){  
 exit(parameter)

}

Void help(){  
 if(getcwd() == NULL):

Perror;

open a file to read with getcwd();

char c = fgetc(file);

make a while loop with (c != EOF):

print©

c = fgetc(file);

fclose(file)

}

Void input\_redirect(cmd, line){

Token = strtok\_r(line, “<” savedfile)

If(token != NULL):

call paresespace();

if(savedptr == ‘<’):

inputmode = 2;

savedptr += 1;

else:

inputmode = 1;

if(savedptr != NULL):

make a while loop with (savedptr == ‘ ‘):

savedptr += 1;

}

Void output\_redirect(cmd, line){

Token = strtok\_r(line, “>” savedfile)

If(token != NULL):

call paresespace();

if(savedptr == ‘>’):

inputmode = 2;

savedptr += 1;

else:

outputmod = 1;

if(savedptr != NULL):

make a while loop with (savedptr == ‘ ‘):

savedptr += 1;

}

Void input\_output\_redirect(cmd, line){

Token1 = strtok\_r(line, “>” savedfile)

If(token != NULL):

call paresespace();

if(savedptr == ‘>’):

outputmode = 2;

savedptr += 1;

else:

outputmod = 1;

if(savedptr != NULL):

make a while loop with (savedptr == ‘ ‘):

savedptr += 1;

Token2 = strtok\_r(line, “<” savedfile)

If(token2 != NULL):

call paresespace();

if(savedptr == ‘>’):

inputmode = 2

savedptr += 1;

else:

inpumode = 1;

if(savedptr != NULL):

make a while loop with (savedptr == ‘ ‘):

savedptr += 1;

}

Void exec(){

If pip:

Create pipe;

Create fork;

Child is to wrte in pipe;

parent is to output the result;

re

Create fork;

Child is to do command;

If commondline.inpumode:

Open a file named “readonly1”;

Redirect input from file;

If commondline.outputmode:

Open a file name “writeonly1”;

Redirect output to file;

Create fork;

Call execv();

}

Int main{

If batchfile exists:

Open a file named “batch.txt” to read;

Make a for loop in file:

Call parseline();

Call exec();

Else{

Read commondlne:

Case to call cd();

Case to call cir();

Case to call dir();

Case to call environ();

Case to call echo();

Case to call pause();

Case to call quit();

Case to call help();

Else:

Call exec();

}