IMPLEMENTATION OF SHELL USING C

FINAL REVIEW REPORT

Submitted by

Ankit Dutta (15BCE2084)
Mayank D. Shikhar (15BCE0020)

Prepared For

OPERATING SYSTEMS (CSE2005) – PROJECT COMPONENT

Submitted To

Prof. Prabakaran N.

Assistant Professor (Sr)

School of Computer Science and Engineering

Table of Contents

1. INTRODUCTION

Simply put, a shell program (sometimes called a shell script) is a text file that contains standard UNIX and shell commands. Each line in a shell program contains a single UNIX command exactly as if you had typed them in yourself. The difference is that you can execute all the commands in a shell program simply by running the shell program (rather than typing all the commands within).Shell programs are interpreted and not compiled programs. This means when you run a shell program a child shell is started. This child shell reads each line in the shell program and carries out the command on that line. When it has read all the commands the child shell finishes.

2. PROJECT SCOPE

Shell script usage has not reduced instead increased over the years. Most of the middleware tools such as ETL, MB etc are installed in UNIX servers. To maintain those servers, pre-processing file , log collection and performance improvement shell scripts are used. In this project we are using C coding to implement some of the features present in a typical shell such as arithmetic operation, viewing history, etc.

3. PROJECT RESOURCE REQUIREMENTS

3.1 Software Resource Requirements

Any UNIX based systems such as Linux, Ubuntu,

etc. Dev C++/Geany/Any C compiler

4. CODE

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<ctype.h>
#include<unistd.h>
#include<math.h>
#include<dirent.h>
#include<time.h>

int i,j,ec,fg,ec2;
int last = 0;
char fn[20],e,c;
FILE *fp1,*fp2,*fp;

void Create();
void Append();
```

```
void Delete();
void Display();
void TextEditor();
void cd();
void listHistory();
void newCmd(char*
cmd); void ls();
void disp_time();

typedef struct{
        char history[30];
}list;

list history_list[100];

void basic_loop(void){
        char input[50] = "";
        char file_name[30];
        char process_name[30];

        while(strcmp("exit",input)!=0){
                printf("\n>>");
                scanf("%s",input);

                newCmd(input);

                if(strcmp("help",input) == 0){
                        printf("The following are the features of this shell: \n");
                        printf("1) Basic operations like addition,subtraction,multiplication \n\t,power &
division can be performed for 2 integers with no spacing.\n");
                        printf("2) editor - Allows the user to use a basic text editor.\n");
                        printf("3) cd   - Prints the current directory.\n");
                        printf("4) history - This allows us to view the previous commands entered!\n");
                        printf("5) ls   - Shows all files in current directory!\n");
                        printf("6) time  - Displays current date & time!\n");
                        printf("7) gcc   - Runs a compiler to compile C/C++ files in the current directory!\n");
                        printf("8) start   - Shows the output of the compiled file!\n");
                        printf("9) run  - Runs any application!\n");
                        printf("10) exit  - Exits the shell!");
                }

                else if(strcmp("exit",input)==0)
                        {
                printf("Goodbye !");
                exit(0);
                }

                else if(strcmp("ls",input)==0)
                        {
                ls();
                }

                else if(strcmp("gcc",input) == 0){
                        char system_call[100];
                        scanf("%s",file_name);
                        sprintf(system_call, "%s %s", input, file_name);
```

```
int error = system(system_call);
if(error == 0){
		printf("Compilation has been successfull!\n");
	}
}

else if(strcmp("start",input) == 0){
	char system_call[100];
	scanf("%s",file_name);
	sprintf(system_call, "%s %s", input,
	file_name); int error = system(system_call);
	if(error == 0){
		printf("Compilation has been successfull!\n");
	}
}

else if(strcmp("run",input) == 0){
	char system_call[100];
	scanf("%s",process_name);
	sprintf(system_call,"%s",process_name);
	printf("The application will be started in a few
	moments.\n"); system(system_call);
	printf("The application has been closed.\n");
}

else if(strcmp("time",input) ==
	0) {
	disp_time();
}

else if(isdigit(input[0])){
	float first, second;

	if(strchr(input,'+')){
		sscanf(input,"%f+%f",&first,&second);
		printf("%f",first+second);
	}

	else if(strchr(input,'-')){ sscanf(input,"%f-
		%f",&first,&second);
		printf("%f",first-second);
	}

	else if(strchr(input,'*')){
		sscanf(input,"%f*%f",&first,&second);
		printf("%f",first*second);
	}

	else if(strchr(input,'/')){
		sscanf(input,"%f/%f",&first,&second);
		printf("%f",first/second);
	}

	else if(strchr(input,'^')){
		sscanf(input,"%f^%f",&first,&second);
```

```
                                printf("%f",pow(first,second));
                        }

                }

                else if(strcmp("editor",input) == 0){
                        printf("\nWelcome to the basic text
                        editor!"); TextEditor();
                }

                else if(strcmp("cd",input)==0)
                        {
                cd();
                }

                else if(strcmp("history",input)==0)
                        {
                listHistory();
                }

                else{
                        printf("Unknown command! Please enter 'help' to get the list of commands");
                }
        }

}

void TextEditor()
{
   do
   {
      printf("\n\nSelect one of the follwing functions:");
      printf("\n1.Create a file.\n2.Display the contents.\n3.Append in a file.\n4.Delete a file.\n5.Exit the
      editor.\n");
      printf("Enter your choice: ");
      scanf("%d",&ec);
      switch(ec)
      {
      case 1:
         Create();
         break;
      case 2:
         Display();
         break;
      case 3:
        Append();
        break;
      case 4:
         Delete();
         break;
      case 5:
         basic_loop();
         break;
      }
   } while(1);
}
```

```
void Create()
{
   fp1=fopen("temp.txt","w");
   printf("\nEnter the text and press '.' to
   save!\n\n");
   while(1)
   {
      c=getchar();
      fputc(c,fp1);
      if(c == '.')
      {
         fclose(fp1);
         printf("\nEnter then new filename:
         "); scanf("%s",fn);
         fp1=fopen("temp.txt","r");
         fp2=fopen(fn,"w");
         while(!feof(fp1))
         {
            c=getc(fp1);
            putc(c,fp2);
         }
         fclose(fp2);
         break;
      }
   }
}

void Display()
{
   printf("\nEnter the file name:
   "); scanf("%s",fn);
   fp1=fopen(fn,"r");
   if(fp1==NULL)
   {
      printf("\n\tFile not
      found!"); goto end1;
   }
   while(!feof(fp1))
   {
      c=getc(fp1);
      printf("%c",c);
   }
   end1:
   fclose(fp1);
   printf("\n\nPress any key to continue!");
}

void Delete()
{
   printf("\nEnter the file name:
   "); scanf("%s",fn);
   fp1=fopen(fn,"r");
   if(fp1==NULL)
   {
      printf("\nFile not found!");
```

```
      goto end2;
   }
   fclose(fp1);
   if(remove(fn)==0)
   {
      printf("\n\nFile has been deleted
      successfully!"); goto end2;
   }
   else
      printf("\nError!\n");
   end2:
   printf("\n\nPress any key to continue!");
}

void Append()
{
   printf("\nEnter the file name:
   "); scanf("%s",fn);
   fp1=fopen(fn,"r");
   if(fp1==NULL)
   {
      printf("\nFile not
      found!"); goto end3;
   }
   while(!feof(fp1))
   {
      c=getc(fp1);
      printf("%c",c);
   }
   fclose(fp1);
   printf("\nType the text and press 'Ctrl+S' to
   append.\n"); fp1=fopen(fn,"a");
   while(1)
   {
      if(c==19)
         goto end3;
      if(c==13)
      {
         c='\n';
         printf("\n\t");
         fputc(c,fp1);
      }
      else
      {
         printf("%c",c);
         fputc(c,fp1);
      }
   }
   end3:
   fclose(fp1);
}

void cd()
{
```

```
   char *buf;
        buf=(char *)malloc(100*sizeof(char));
        getcwd(buf,100);
        printf("The current directory is : %s",buf);
}

void ls(){
   DIR *d;
   struct dirent *dir;
   d = opendir(".");
   printf("\nThe files in the directory are:
   \n"); if (d){
      while ((dir = readdir(d)) != NULL){
         printf("%s\n", dir->d_name);
      }
      closedir(d);
   }
}

void newCmd(char* cmd) {
   strcpy(history_list[last].history,cmd);
        last++;
 }

void listHistory(){
        printf("\nThe commands last entered
        are:\n");
        for (int z = 0;z < last;z++){
                printf("%s\n",history_list[z].history);
        }
}

void  disp_time(){
        time_t   t;
        time(&t);
        printf("Today's date and time : %s",ctime(&t));
}

int main(){
        printf("Welcome user to this basic UNIX shell! This has been implemented using C
        code.\n");
        basic_loop();
}
```

5. CODE BREAKDOWN

The following functions have been implemented in this project:

1) Basic arithmetic operations like addition, subtraction, multiplication, division and taking exponential power.
2) A basic text editor to create and edit text files.
3) Printing the current directory and the files in the directory.
4) It allows us to see a maximum of 10 items as the history of the previous commands entered.
5) See the current date and time.
6) Call the inbuilt compiler to compile files (C/C++ files) and display their output.
7) Start any other process using the compiler.

6. WORK BREAK DOWN

Team Member Registration Number	Name	Work Assigned
15BCE0020	Mayank D. Shikhar	Ppt And Report
15BCE2084	Ankit Dutta	Code