A Lab Mini Project Report

On

**OUR OWN SHELL**

Semester: IV

Section: CSE-A

**COURSE:** Operating Systems Lab

**BY**

**ADITYA. N**

**(1602-18-733-004)**

**T. HEMANTH REDDY**

**(1602-18-733-022)**



UNDER THE GUIDANCE OF

**Smt. Sunitha Reddy**

**ASSISTANT PROFESSOR**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**VASAVI COLLEGE OF ENGINEERING(AUTONOMOUS)**

**AFFLIATED TO OSMANIA UNIVERSITY**

**IBRAHIMBAGH**

**HYDERABAD-500031**

**ACKNOWLEDGEMENT**

*We would like to express our heartfelt gratitude to Smt. Sunitha Reddy, our project guide, for her valuable guidance and constant support, along with her capable instructions and persistent encouragement.*

*We are grateful to our Head of Department, Dr. T. Adilakshmi, for her steady support and the provision of every resource required for the completion of this project.*

*We would like to take this opportunity to thank our Principal, Dr. S. V. Ramana, as well as the management of the institute, for having designed an excellent learning atmosphere.*

*We are thankful to and fortunate enough to get constant encouragement, support and guidance which helped us in successfully completing our project work.*

**Operating Systems Lab**

**Mini Project**

**on**

**OUR OWN SHELL**

**ABSTRACT**

**Shell** Made Easy to all the users. If you learn how to write shell scripts like a pro, you can solve real-world problems, or automate repetitive and complex tasks.

Shell is a window where many of the developers used to solve complex problems with less effort and less commands. It is the platform where developer can control total OS at one place.

We came up with a similar kind of platform where one can create directories, files, docx etc. using mkdir , one can move from one location to another location using cd **,**one can perform mathematical operations on various data types etc..

On further developing our project it might become the future OS which has good user friendly outlook, and even a beginner to programming world can understand and can code for various problems.

**TABLE OF CONTENTS**

1. Introduction 1

2. System Design 2

3. System Analysis 3

3.1. Functionalities present 3

3.2. Operation Commands 4

4. Code 5

3.1. arithemeticoperations.c 5

3.2. Basic\_Operations.c 7

3.3. String\_operations.c 14

3.4. arrayoperations.c 18

3.5. mainshell.c 23

4. Output 32

5. Conclusion 38

5. References 39

**LIST OF FIGURES**

Fig 2.1: main tree 2

Fig 2.2: sub trees 2

Fig 5.1: Help Command 32

Fig 5.2: Manipulating Commands 33

Fig 5.3: Arithematic Operations 34

Fig 5.4: Basic Operations 35

Fig 5.5: Array Operations 36

Fig 5.6: String Operations 37

1. **INTRODUCTION**

Our Own Shell a platform similar to all kinds of shells like  bourne (sh), KornShell (ksh), C shell (csh), Z shell (zsh), with few functions and commands for beginners . This shell is built in C language. So here in this shell we have a wide operations that can be used by the user. So most of the basic shells will have operations on Arithemetic and some basic operations on string. But whereas coming to our shell one can have many operations that he can perform on Arrays and Strings aswell. And we have an added feature of some directory manipulating commands where a user can switch from one directory/file/folder to any specified location he want and he could also manipulate the files i.e one can add as many number of files he wants and he could also list the contents that are present in the file. The Color (additional feature) where a user could use which brings brightness to the shell.

1. **STYSTEM DESIGN**

**FLOW CHART**

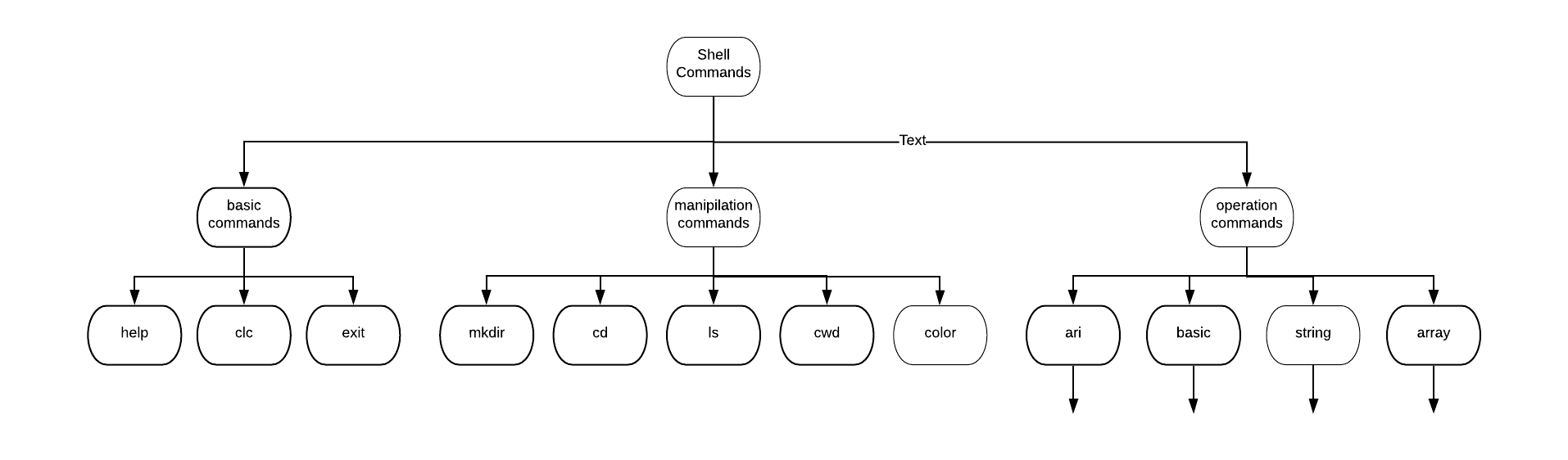


Fig 2.1. main tree

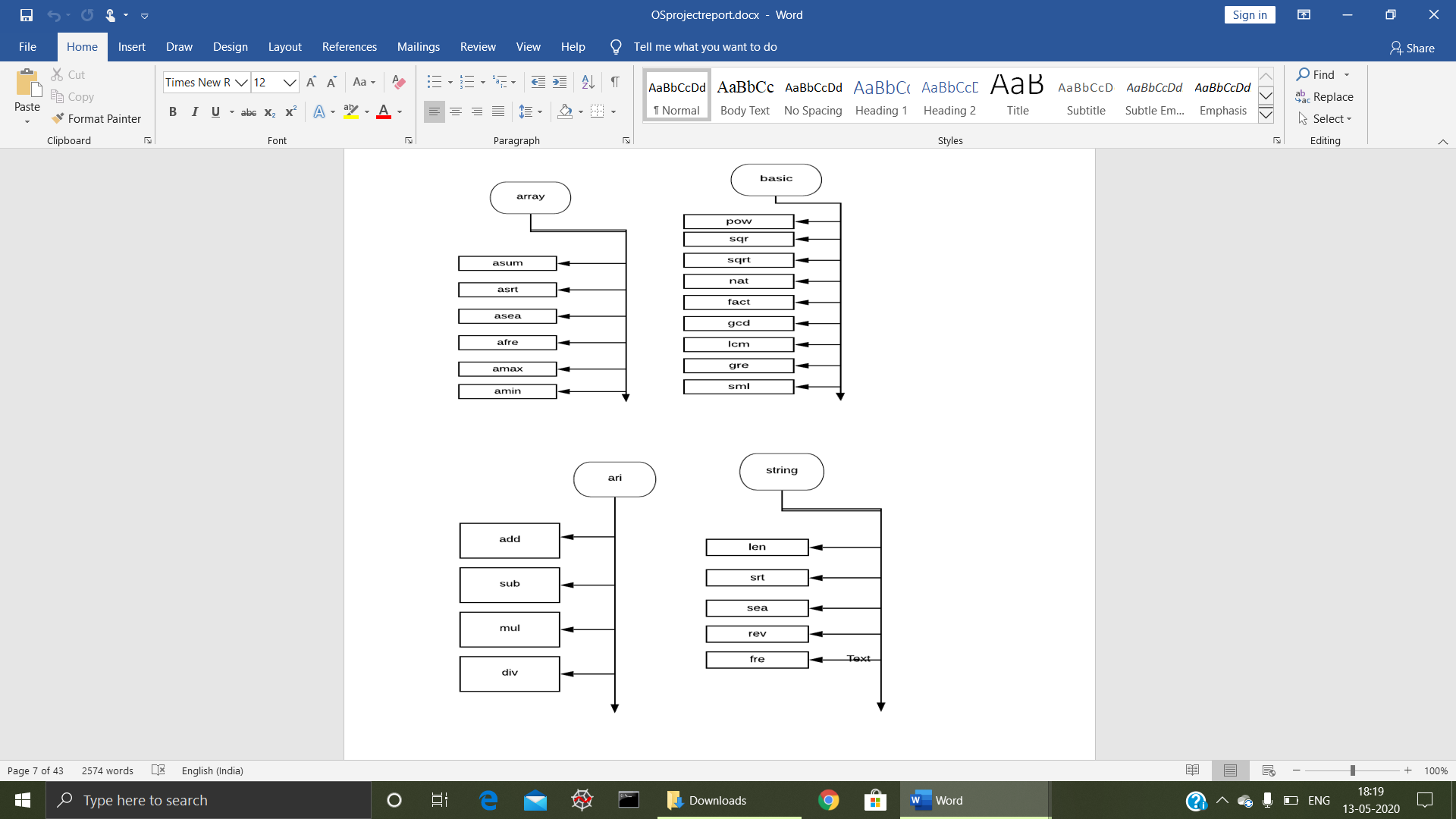


Fig 2.2. Sub trees

1. **SYSTEM ANALYSIS**

**Functionalities present :**

**Basic Commands**

* help : helps in showing what all commands present.
* clc : clears the screen.
* Exit : exit from the shell.

**Manipulating Commands**

* mkdir : creates a directory/file/docx etc..
* cd : change directory. ( used to move from one location to another location).
* ls : lists all the contents. ( like directories, files, docx etc..)
* cwd : shows the current working directory.
* color : used for formatting the text.(includes red, cyan, blue, yellow, green etc..)

**Operation Commands**

* ari : library contains Arithemetic Operations.
* string : library contains String Operations.
* basic : library contains Basic Operations.
* array : library contains Array Operations.

**Operation Commands :**

* **ari**
* add : Addition between two numbers.
* sub : Subtraction between two numbers.
* mul : Multiplication between two numbers.
* div : Division between two numbers.
* **basic**
* fact : Factorial of a number
* pow : Power of a number
* sqr : Square of a number
* sqrt : Square Root of a number
* gcd : GCD of any given numbers
* lcm : LCM of any given numbers
* gre : Greatest of any given numbers
* sml : Smallest of any given numbers
* nat : Nature of the number (even or odd)
* **string**
* len : Length of the String
* srt : Sort the String
* sea : Search the given Character in String
* rev : Reverse the String
* fre : Frequency of a Character
* **array**
* asum : Sum of the given Numbers
* asrt : Sort the given Numbers
* asea : Search the given Number
* afre : Frequency of a Number
* amax : Maximum of the given Numbers
* amin : Minimum of the given Numbers
* Additional to all these commands we have one more command
* back : Used when required function is not present in that library

1. **CODE**

**arithemeticoperations.c**

#include<stdio.h>

#include<math.h>

//for Arithemetic operations like Addition,Substraction,Division,Multiplication

void division(double a,double b)

{

printf("Quotient is %lf \n",a/b);

}

void multiplication(double a,double b)

{

printf("Product is %lf\n",a\*b);

}

void addition(double a,double b)

{

printf("Sum is %lf\n",a+b);

}

void substraction(double a,double b)

{

printf("Difference is %lf\n",a-b);

}

void Arithemeticfunctions(char str[])

{

double a;

double b;

printf("Enter the First Number\n");

scanf("%lf",&a);

printf("Enter the Second Number\n");

scanf("%lf",&b);

if((strcmp(str,"add"))==0)

{

addition(a,b);

}

else if((strcmp(str,"sub"))==0)

{

substraction(a,b);

}

else if((strcmp(str,"mul"))==0)

{

multiplication(a,b);

}

else if((strcmp(str,"div"))==0)

{

division(a,b);

}

}

**Basic\_Operations.c**

#include<stdio.h>

#include<math.h>

// Basic operations to perform factorial,power,square,square root,gcd,lcm,even or odd,greater,smaller

void factorial()

{

int n,i,p=1;

printf("\n enter number");

scanf("%d",&n);

for(i=1;i<=n;i++)

{

p=p\*i;

}

printf("\n factorial of %d is %d",n,p);

}

void power()

{

int b,e,i,r=1;

printf("\n enter base : ");

scanf("%d",&b);

printf("\n enter power : ");

scanf("%d",&e);

for(i=0;i<e;i++)

{

r=r\*b;

}

printf("%d^%d = %d",b,e,r);

}

void square()

{

int n,i;

printf("\n enter element : ");

scanf("%d",&n);

i=n\*n;

printf("square of %d is %d",n,i);

}

void squareroot()

{

double n;

double i;

printf("\n enter element : ");

scanf("%ld",&n);

i=n;

printf("square of %ld is %lf",n,i);

}

void gcd()

{

int n,i;

int j;

int minimum;

int GCD;

int temp;

printf("\n enter no.of elements : ");

scanf("%d",&n);

int array[n];

for(i=0;i<n;i++)

{

printf("enter element %d",(i+1));

scanf("%d",&array[i]);

if(i==0)

{

minimum=array[0];

}

else

{

if(minimum>array[i])

{

minimum=array[i];

}

}

}

for (i=1; i < minimum/2; i++)

{

for (j = 0; j < n;j++)

{

GCD = 2\*i;

temp = ((array[j])/(GCD));

int check = temp%1;

if (check == 0)

break;

}

}

printf("\nThe Greates Common Devisor is: %d", GCD);

}

void lcm()

{

int n,i;

int j;

int maximum;

int LCM;

int temp;

printf("\n enter no.of elements : ");

scanf("%d",&n);

int array[n];

for(i=0;i<n;i++)

{

printf("enter element %d",(i+1));

scanf("%d",&array[i]);

if(i==0)

{

maximum=array[0];

}

else

{

if(maximum<array[i])

{

maximum=array[i];

}

}

}

LCM=maximum;

while(1)

{

for (j = 0; j < n;j++)

{

if(LCM%array[i]!=0)

{

break;

}

}

if(j==n)

{

break;

}

LCM=LCM+1;

}

printf("\nThe Least Common Multiple is: %d", LCM);

}

void evenodd()

{

int n;

printf("\n enter element : ");

scanf("%d",&n);

if(n%2==0)

{

printf("\n%d is even",n);

}

else

{

printf("\n%d is odd",n);

}

}

void greater()

{

int first,second;

printf("\n enter first element : ");

scanf("%d",&first);

printf("\n enter second element : ");

scanf("%d",&second);

if(first>second)

{

printf("\n Yes %d is greater than %d",first,second);

}

else

{

printf("\n No %d is not greater than %d",first,second);

}

}

void smaller()

{

int first,second;

printf("\n enter first element : ");

scanf("%d",&first);

printf("\n enter second element : ");

scanf("%d",&second);

if(first<second)

{

printf("\n Yes %d is smaller than %d",first,second);

}

else

{

printf("\n No %d is not smaller than %d",first,second);

}

}

void Basic\_Operations(char \*str)

{

if(strcmp(str,"fact")==0)

{

factorial();

}

else if((strcmp(str,"pow"))==0)

{

power();

}

else if((strcmp(str,"sqr"))==0)

{

square();

}

else if((strcmp(str,"sqrt"))==0)

{

squareroot();

}

else if((strcmp(str,"gcd"))==0)

{

gcd();

}

else if((strcmp(str,"lcm"))==0)

{

lcm();

}

else if((strcmp(str,"nat"))==0)

{

evenodd();

}

else if((strcmp(str,"gre"))==0)

{

greater();

}

else if((strcmp(str,"sml"))==0)

{

smaller();

}}

**String\_operations.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

// String operations such as length,reverse,sort,search,frequency

void search(char string[])

{

char ch;

int i;

printf("\nenter element to be searched");

scanf("%c",&ch);

for(i=0;string[i]!='\0';i++)

{

if(ch==string[i])

{

printf("\npresent");

break;

}

}

if(string[i]=='\0')

{

printf("\nnot present");

}

}

void sort(char string[])

{

char temp;

int i,j;

int n = strlen(string);

printf("String before sorting - %s \n", string);

for (i = 0; i < n-1; i++) {

for (j = i+1; j < n; j++) {

if (string[i] > string[j]) {

temp = string[i];

string[i] = string[j];

string[j] = temp;

}

}

}

printf("String after sorting - %s \n", string);

}

void reverse(char string[])

{

int i,j;

for(i=0; string[i]!='\0'; i++)

{

}

char array[i];

j=i-1;

for(i=0; j>=0; i++,j--)

{

array[i]=string[j];

}

array[i]='\0';

for(i=0; array[i]!='\0'; i++)

{

printf("%c",array[i]);

}

}

void frequency(char string[])

{

int i,c=0;

char ch;

printf("\nenter element ");

scanf("%c",&ch);

for(i=0;string[i]!='\0';i++)

{

if(ch==string[i])

{

c=c+1;

}

}

printf("\nfrequency of %c is %d",ch,c);

}

void String\_Operations(char \*ch)

{

char string[1000];

printf("enter a String");

gets(string);

if(strcmp(ch,"sea")==0)

{

search(string);

}

else if(strcmp(ch,"srt")==0)

{

sort(string);

}

else if(strcmp(ch,"len")==0)

{

printf("string length %d",strlen(string));

}

else if(strcmp(ch,"rev")==0)

{

reverse(string);

}

else if(strcmp(ch,"fre")==0)

{

frequency(string);

}

}

**arrayoperations.c**

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<string.h>

// for operations on array such as minimum,maximum,sort,find..

void sumofarrayelements(int \*arr,int n)

{

int i;

int sum=0;

for(i=0;i<n;i++)

{

sum=sum+arr[i];

}

printf("The sum of array elements is %d\n",sum);

}

void maxofarrayelements(int \*arr,int n)

{

int i;

int max=arr[0];

for(i=1;i<n;i++)

{

if(max < arr[i])

{

max=arr[i];

}

}

printf("The maximum of array elements is %d\n",max);

}

void minofarrayelements(int \*arr,int n)

{

int i;

int min=arr[0];

for(i=1;i<n;i++)

{

if(min > arr[i])

{

min=arr[i];

}

}

printf("The minimum of array elements is %d\n",min);

}

void search\_array\_element\_linearsearch(int \*arr,int n,int search\_element)

{

int i;

for(i=0;i<n;i++)

{

if(arr[i]==search\_element)

{

printf("Element Found");

break;

}

}

if(i==n)

{

printf("Element Not Found");

}

}

int comp(const void \*a,const void \*b)

{

int x=\*(int \*)a;

int y=\*(int \*)b;

return x > y;

}

void sort\_of\_elements(int \*arr,int n)

{

qsort(arr,n,sizeof(int),comp);

int i;

printf("Printing the elements in the sorted Order\n");

for(i=0;i<n;i++)

{

printf("%d ",arr[i]);

}

printf("\n");

}

void frequency\_of\_an\_element(int \*arr,int n,int ele\_req)

{

int i;

int count=0;

for(i=0;i<n;i++)

{

if(arr[i]==ele\_req)

{

count++;

}

}

printf("The frequency of %d is %d",ele\_req,count);

}

void Array\_Functions(char str[])

{

printf("Enter the number of elements of the Array");

int n,i;

scanf("%d",&n);

int \*arr=(int \*)malloc(n \* sizeof(int));

printf("Enter the %d elements",n);

for(i=0;i<n;i++)

{

scanf("%d",&arr[i]);

}

if((strcmp(str,"asum"))==0)

{

sumofarrayelements(arr,n);

}

else if((strcmp(str,"asea"))==0)

{

printf("Enter the element that you wanted to search\n");

int ele;

scanf("%d",&ele);

search\_array\_element\_linearsearch(arr,n,ele);

}

else if((strcmp(str,"amin"))==0)

{

minofarrayelements(arr,n);

}

else if((strcmp(str,"amax"))==0)

{

maxofarrayelements(arr,n);

}

else if((strcmp(str,"asrt"))==0)

{

sort\_of\_elements(arr,n);

}

else if((strcmp(str,"afre"))==0)

{

printf("Enter the element that you wanted to search\n");

int ele;

scanf("%d",&ele);

frequency\_of\_an\_element(arr,n,ele);

}

}

**mainshell.c**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<readline/readline.h>

#include<limits.h>

#include <unistd.h>

#include <dirent.h>

#define clear() printf("\033[H\033[J")

#include "String\_operations.c"

#include "arithemeticoperations.c"

#include "Basic\_Operations.c"

#include "arrayoperations.c"

void init\_shell()

{

printf("\n\n\n\n\t\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\n\n\n\t\t\t\t\t\t \*\*\*\*Aditya's Hemanth's SHELL\*\*\*\*");

char\* username = getenv("USER");

printf("\n\n\nUSER is: @%s", username);

printf("\n");

sleep(05);

}

void HelpFunction()

{

printf("\n clc : Clears the command prompt");

printf("\n ls : List all the files present in the Directory");

printf("\n cd : To change the directory");

printf("\n cwd : Current Working Directory");

printf("\n mkdir : To create a Directory");

printf("\n ari : Arithematic Operations");

printf("\n string : String Operations");

printf("\n basic : Basic Operations");

printf("\n array : Array Operations");

printf("\n color : Formatting the Text to different colors");

printf("\n exit : Exit from the command prompt\n");

}

void checkfun(char \*str)

{

if(strcmp(str,"help")==0)

{

HelpFunction();

}

else if(strcmp(str,"clc")==0)

{

clear();

}

else if(strcmp(str,"string")== 0)

{

char iistr[20];

printf("\n len : Length of the String");

printf("\n srt : Sort the String");

printf("\n sea : Search the given Character in String");

printf("\n rev : Reverse the String");

printf("\n fre : Frequency of a Charecter\n");

printf(" back : No required function\n");

printf(">>>");

scanf("%s",iistr);

if((strcmp(iistr,"len")==0) || (strcmp(iistr,"srt")==0) || (strcmp(iistr,"sea")==0) || (strcmp(iistr,"rev")==0) || (strcmp(iistr,"fre")==0))

{

String\_Operations(iistr);

}

else if(strcmp(iistr,"back")== 0){}

else

{

checkfun(iistr);

}

printf("\n");

}

else if(strcmp(str,"basic")== 0)

{

char iistr[20];

printf("\n fact : Factorial of a number");

printf("\n pow : Power of a number");

printf("\n sqr : Square of a number");

printf("\n sqrt : Square Root of a number");

printf("\n gcd : GCD of any given numbers");

printf("\n lcm : LCM of any given numbers");

printf("\n gre : Greatest of any given numbers");

printf("\n sml : Smallest of any given numbers");

printf("\n nat : Nature of the number (even or odd)\n");

printf(" back : No required function\n");

printf(">>>");

scanf("%s",iistr);

if((strcmp(iistr,"fact")==0) || (strcmp(iistr,"pow")==0) || (strcmp(iistr,"sqr")==0) || (strcmp(iistr,"sqrt")==0) || (strcmp(iistr,"gcd")==0) || (strcmp(iistr,"lcm")==0) || (strcmp(iistr,"gre")==0) || (strcmp(iistr,"nat")==0) || (strcmp(iistr,"sml")==0))

{

Basic\_Operations(iistr);

}

else if(strcmp(iistr,"back")== 0){}

else

{

checkfun(iistr);

}

printf("\n");

}

else if(strcmp(str,"ari")== 0)

{

char iistr[20];

printf("\n add : Add two numbers");

printf("\n sub : Subtract two numbers");

printf("\n mul : Multiply two numbers");

printf("\n div : Divide two numbers\n");

printf(" back : No required function\n");

printf(">>>");

scanf("%s",iistr);

if((strcmp(iistr,"add")==0) || (strcmp(iistr,"sub")==0) || (strcmp(iistr,"mul")==0) || (strcmp(iistr,"div")==0))

{

Arithemeticfunctions(iistr);

}

else if(strcmp(iistr,"back")== 0){}

else

{

checkfun(iistr);

}

}

else if(strcmp(str,"array")==0)

{

char iistr[20];

printf("\n asum : Sum of the given Numbers");

printf("\n asrt : Sort the given Numbers");

printf("\n asea : Search the given Number");

printf("\n afre : Frequency of a Number");

printf("\n amax : Maximum of the given Numbers");

printf("\n amin : Minimum of the given Numbers\n");

printf(" back : No required function\n");

printf(">>>");

scanf("%s",iistr);

printf("\n");

if((strcmp(iistr,"asum")==0) || (strcmp(iistr,"asrt")==0) || (strcmp(iistr,"asea")==0) || (strcmp(iistr,"afre")==0) || (strcmp(iistr,"amax")==0) || (strcmp(iistr,"amin")==0))

{

Array\_Functions(iistr);

}

else if(strcmp(iistr,"back")== 0){}

else

{

checkfun(iistr);

}

}

else if(strcmp(str,"exit")==0)

{

exit(0);

}

else if(strcmp(str,"cwd")==0)

{

char cwd1[PATH\_MAX];

if (getcwd(cwd1, sizeof(cwd1)) != NULL)

{

printf("Current working dir: %s\n", cwd1);

}

else

{

perror("getcwd() error");

}

}

else if(strcmp(str,"ls")==0)

{

DIR \*d;

struct dirent \*dir;

d = opendir(".");

if (d)

{

while ((dir = readdir(d)) != NULL)

{

printf("%s\n", dir->d\_name);

}

closedir(d);

}

}

else if(strcmp(str,"cd")==0)

{

char iistr[50];

char cwd1[PATH\_MAX];

if (getcwd(cwd1, sizeof(cwd1)) != NULL){}

else

{

perror("getcwd() error");

}

printf("Enter the destination folder");

scanf("%s",iistr);

if(strcmp(iistr,"..")==0)

{

chdir("..");

}

else

{

strcat(cwd1,"//");

chdir(strcat(cwd1,iistr));

}

}

else if(strcmp(str,"mkdir")==0)

{

int check;

char dirname[50];

printf("Enter the Directory name:-\n");

scanf("%s",dirname);

check = mkdir(dirname);

if (!check)

printf("Directory created\n");

else

{

printf("Unable to create directory\n");

}

}

else if(strcmp(str,"color")==0)

{

printf("\n red : Red");

printf("\n gre : Green");

printf("\n blu : Blue");

printf("\n yel : Yellow");

printf("\n whi : White");

printf("\n brd : BoldRed");

printf("\n mag : Magneta");

printf("\n cya : Cyan\n");

char iistr[20];

printf(" back : No desired color\n");

printf(">>>");

scanf("%s",iistr);

if(strcmp(iistr,"red")==0)

{

printf("\033[0;31m");

}

else if(strcmp(iistr,"blu")==0)

{

printf("\033[0;34m");

}

else if(strcmp(iistr,"yel")==0)

{

printf("\033[0;33m");

}

else if(strcmp(iistr,"cya")==0)

{

printf("\033[0;36m");

}

else if(strcmp(iistr,"brd")==0)

{

printf("\033[1;31m");

}else if(strcmp(iistr,"mag")==0)

{

printf("\033[0;35m");

}else if(strcmp(iistr,"gre")==0)

{

printf("\033[0;32m");

}else if(strcmp(iistr,"whi")==0)

{

printf("\033[0m");

}

else if(strcmp(iistr,"back")==0){}

else

{

checkfun(iistr);

}

}

else

{

printf("\n Please Enter a Valid Command\n");

}

}

int input(char \*str)

{

char\* buf;

buf = readline(">>>");

if (strlen(buf) != 0) {

strcpy(str, buf);

checkfun(str);

return 0;

} else {

return 1;

}

}

int main()

{

clear();

init\_shell();

char inpstr[1000];

while(1)

{

input(inpstr);

}

return 0;

}

1. **OUTPUT**

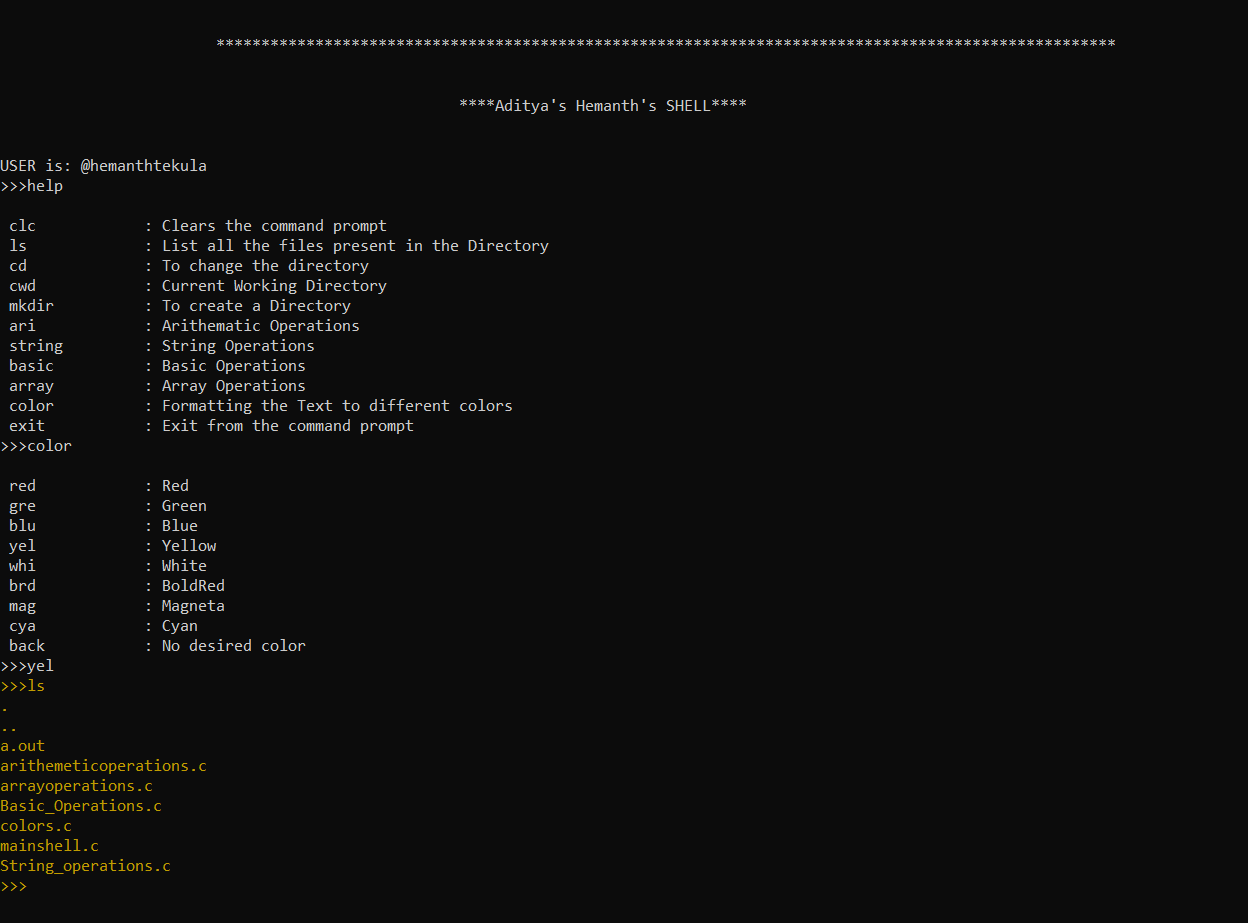
****

Fig 5.1. Help Command

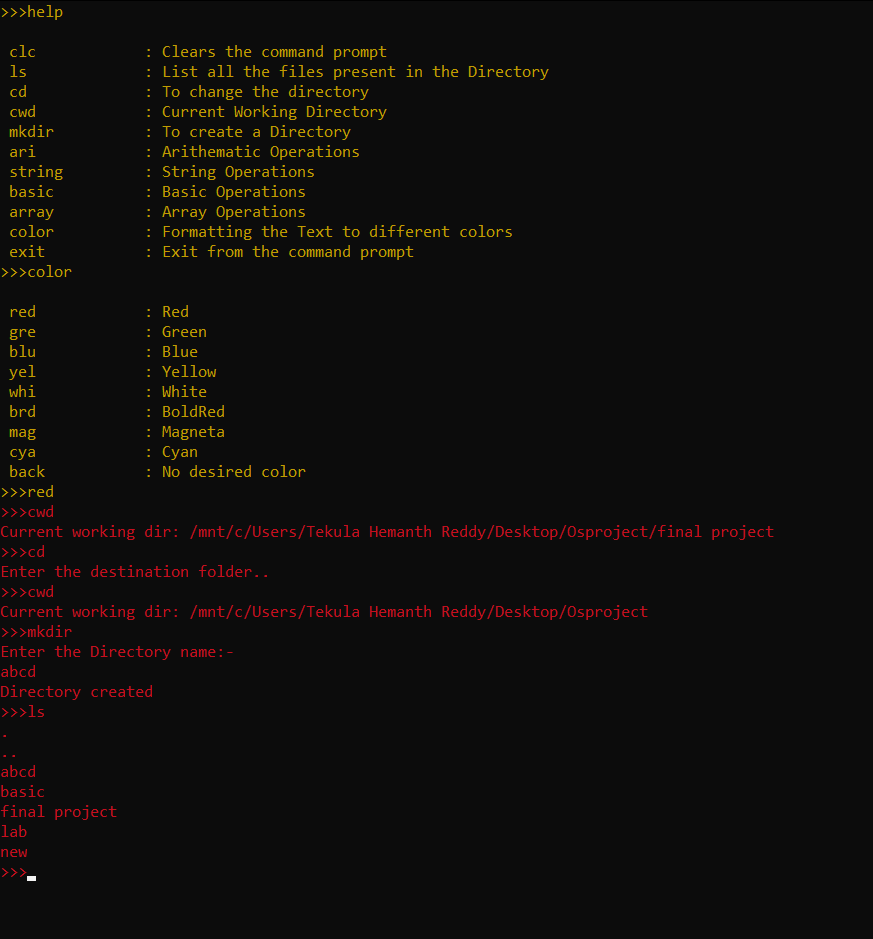
****

Fig 5.2. Manipulating Commands

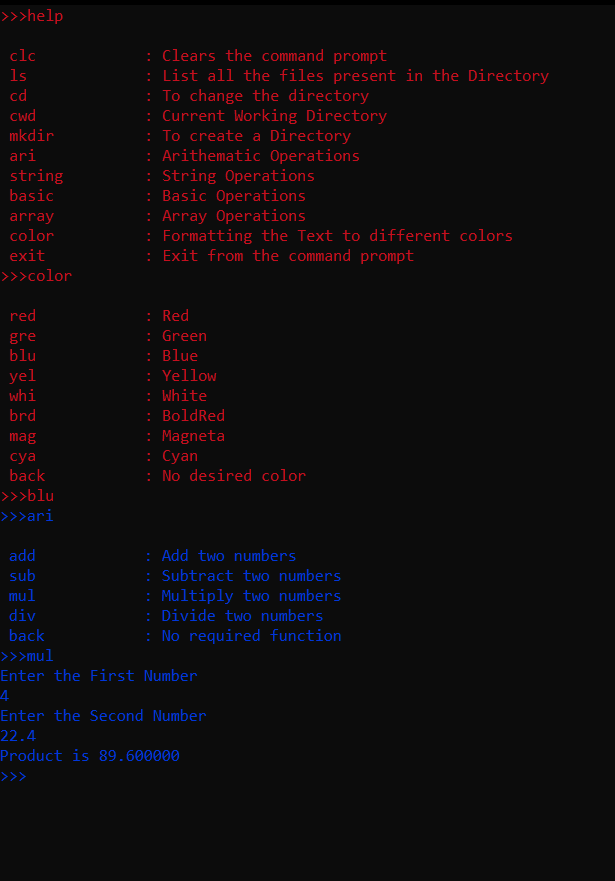
****

Fig 5.3. Arithematic Operations

****

Fig 5.4. Basic Operations

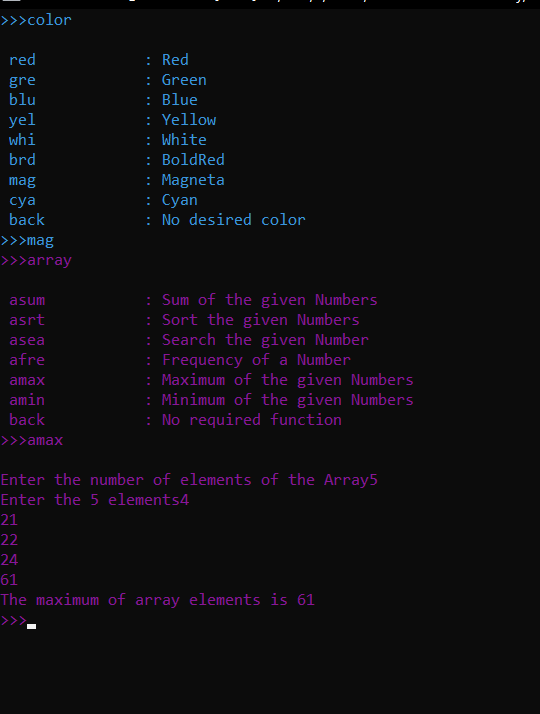
****

Fig 5.5. Array Operations

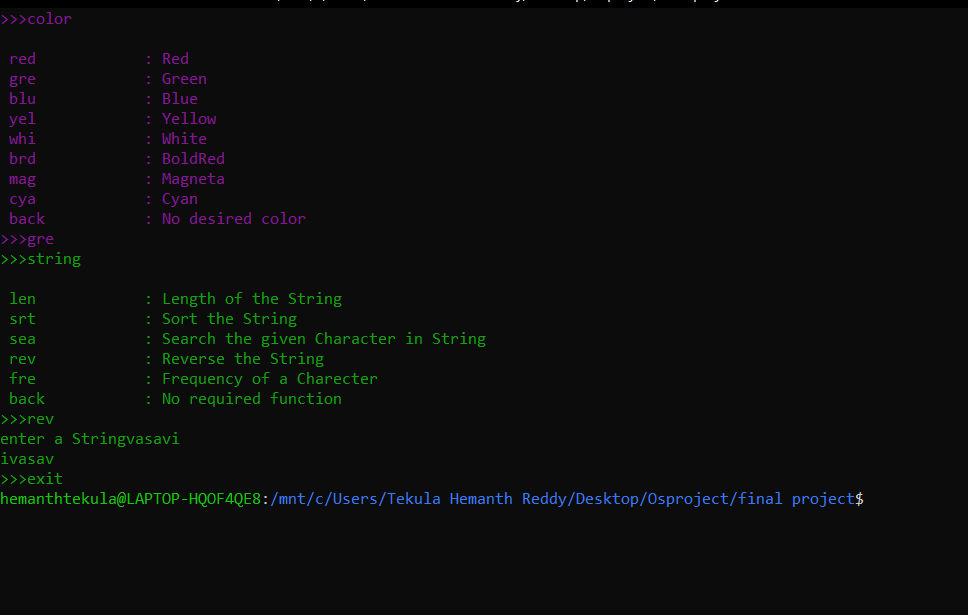
****

Fig 5.6. String Operations

1. **CONCLUSION**

We conclude that this project could be a good kick start for a developer who has a goal of developing his own Operating System. For any operating system there need to be a Command Prompt like that kind of a shell. As a developer if he could work on the most basic and the most important part of any Operating System i.e the Shell/Command Prompt the OS can have a great start off. As we have seen in the project there were commands like cd, mkdir , ls where a user is not directly going through the UI but he is able to workout with them in the background.

**Shell Made Easy to all the users. If you learn how to write shell scripts like a pro, you can solve real-world problems, or automate repetitive and complex tasks.**

1. **REFERENCES**

* OPERATING SYSTEM CONCEPTS - BY PETER B GALVIN ,ABRAHAM SILBERSCHARTZ,GERG GAGNE
* <https://www.youtube.com/watch?v=K6AdwYBxI2I> 🡪 Youtube video on building shell
* <https://www.geeksforgeeks.org/making-linux-shell-c/>
* <https://stackoverflow.com/questions/7430248/creating-a-new-directory-in-c>