Operating Systems Lab

Semester-5, B.tech(ICT)

Roll No:-1401010, Name:-Deep C. Patel

Assignment-2

1).

Write down a program which works like a scientific calculator and performs following operations:-

1) Floating point calculations

2) Trigonometric operations

3) Logarithms

4) Exponent etc.

(Hint: Use awk and bc)

Code:-

```
#!/bin/bash

#Created by Deep C. Patel - 1401010

clear

while true
do

    echo -e "\nEnter from following:-"
    echo "1).Basic Operations Calculator-Floating Point"
    echo "2).Trigonometric Calculations"
    echo "3).Log Calculations"
    echo "4).Exponential Calculations"
    echo "5).Exit"
    echo -e "Enter:\c"
    read choice

    case $choice in
         [1])
```

```
                echo -e "\nEnter from following:-"
                echo "1).Addition"
                echo "2).Subtraction"
                echo "3).Multiplication"
                echo "4).Division"
                echo "5).Modulus"
                echo "6).Exit"
                echo -e "Enter:\c"
                read choice1

                case $choice1 in
                     [1])
                        echo -e "\nEnter X:\c"
                        read x;

                        echo -e "Enter Y:\c"
                        read y;

                        echo -e "\nSum=\c"
                        echo "scale=3;$x+$y" | bc

                        echo -e "\nPress any Key to Continue"
                        read ch
                        continue;;

                    [2])
                        echo -e "\nEnter X:\c"
                        read x;

                        echo -e "Enter Y:\c"
                        read y;

                        echo -e "\nDifference=\c"
                        echo "scale=3;$x-$y" | bc

                        echo -e "\nPress any Key to Continue"
                        read ch
                        continue;;

                    [3])
                        echo -e "\nEnter X:\c"
                        read x;

                        echo -e "Enter Y:\c"
                        read y;

                        echo -e "\nProduct=\c"
                        echo "scale=3;$x*$y" | bc
```

```
                    echo -e "\nPress any Key to Continue"
                    read ch
                    continue;;

                [4])
                    echo -e "\nEnter X:\c"
                    read x;

                    echo -e "Enter Y:\c"
                    read y;

                    echo -e "\nQuotient=\c"
                    echo "scale=3;$x/$y" | bc


                    echo -e "\nPress any Key to Continue"
                    read ch
                    continue;;

                [5])
                    echo -e "\nEnter X:\c"
                    read x;

                    echo -e "Enter Y:\c"
                    read y;

                    echo -e "\nModulus=\c"
                    echo "$x%$y" | bc

                    echo -e "\nPress any Key to Continue"
                    read ch
                    continue;;

                [6]) continue;;
            esac
            continue;;

        [2])
           echo -e "\nEnter Angle(degree): \c"
           read angle

           pi=$( echo "4*a(1)" | bc -l )
           rad=$( echo "$angle*($pi/180)" | bc -l )

           sin=$( echo "scale=5;s($rad)" | bc -l )
           cos=$( echo "scale=5;c($rad)" | bc -l )
```

```
            echo -e "\nSIN($deg):$sin"
            echo "COS($deg):$cos"
            echo -e "TAN($deg):\c"
            echo "scale=5;$sin/$cos" | bc -l
            echo -e "COSEC($deg):\c"
            echo "scale=5;1/$sin" | bc -l
            echo -e "SEC($deg):\c"
            echo "scale=5;1/$cos" | bc -l
            echo -e "COT($deg):\c"
            echo "scale=5;$cos/$sin" | bc -l

            echo -e "\nPress any Key to Continue"
            read ch
            continue;;

         [3])
            echo -e "\nEnter Number:\c"
            read number
            echo -e "\nAnswer:\c"
            echo "l($number)/l(10)" | bc -l

            echo -e "\nPress any Key to Continue"
            read ch
            continue;;
        [4])
            echo -e "\nEnter a Base:\c"
            read base
            echo -e "Enter Power:\c"
            read power

            echo -e "\nExponent:\c"
            echo "$base $power" | awk '{ print (($1)^$2) ); }'

            echo -e "\nPress any Key to Continue"
            read ch
            continue;;

        [5]) exit;;

        *)
            echo -e "\nImproper Choice, Press any Key to
Continue"
            read ch
            continue;;

    esac
done
```

Screenshot:-


ramkabir@Ramkabir: /media/ramkabir/OS/Users/cmp/Deep/Semester 5/OS/Lab/Lab 2
SIN():.70710
COS():.70710
TAN():1.00000
COSEC():1.41422
SEC():1.41422
COT():1.00000
Press any Key to Continue
Enter from following:-
1).Basic Operations Calculator-Floating Point
2).Trigonometric Calculations
3).Log Calculations
4).Exponential Calculations
5).Exit
Enter:3
Enter Number:10000
Answer:4.00000000000000000001
Press any Key to Continue
ramkabir@Ramkabir: /media/ramkabir/OS/Users/cmp/Deep/Semester 5/OS/Lab/Lab 2
Modulus=0
Press any Key to Continue
Enter from following:-
1).Basic Operations Calculator-Floating Point
2).Trigonometric Calculations
3).Log Calculations
4).Exponential Calculations
5).Exit
Enter:2
Enter Angle(degree): 45
SIN():.70710
COS():.70710
TAN():1.00000
COSEC():1.41422
SEC():1.41422
COT():1.00000
Press any Key to Continue

```
ramkabir@Ramkabir: /media/ramkabir/OS/Users/cmp/Deep/Semester 5/OS/Lab/Lab 2
Enter from following:-
1).Basic Operations Calculator-Floating Point
2).Trigonometric Calculations
3).Log Calculations
4).Exponential Calculations
5).Exit
Enter:1

Enter from following:-
1).Addition
2).Subtraction
3).Multiplication
4).Division
5).Modulus
6).Exit
Enter:5

Enter X:4
Enter Y:1

Modulus=0

Press any Key to Continue
```

```
ramkabir@Ramkabir: /media/ramkabir/OS/Users/cmp/Deep/Semester 5/OS/Lab/Lab 2
Enter from following:-
1).Basic Operations Calculator-Floating Point
2).Trigonometric Calculations
3).Log Calculations
4).Exponential Calculations
5).Exit
Enter:1

Enter from following:-
1).Addition
2).Subtraction
3).Multiplication
4).Division
5).Modulus
6).Exit
Enter:1

Enter X:4
Enter Y:5

Sum=9

Press any Key to Continue
```

---

2).

Find out following files on your machine.

a. Biggest file, if more than one exists than list them

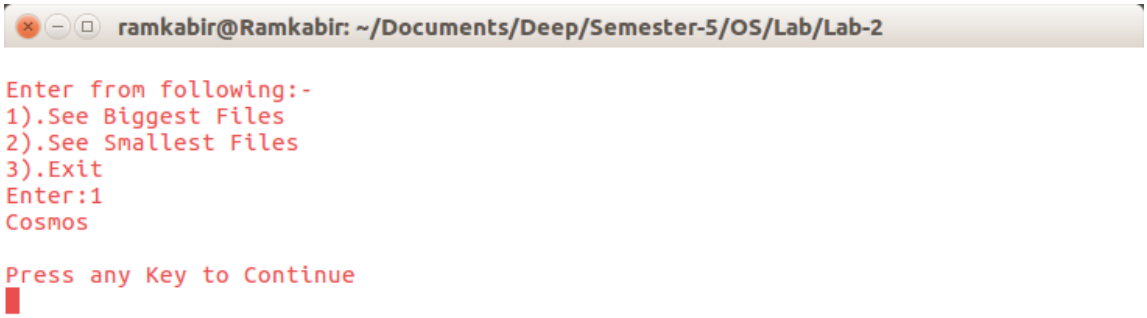
b. Smallest file, if more than one exists than list them

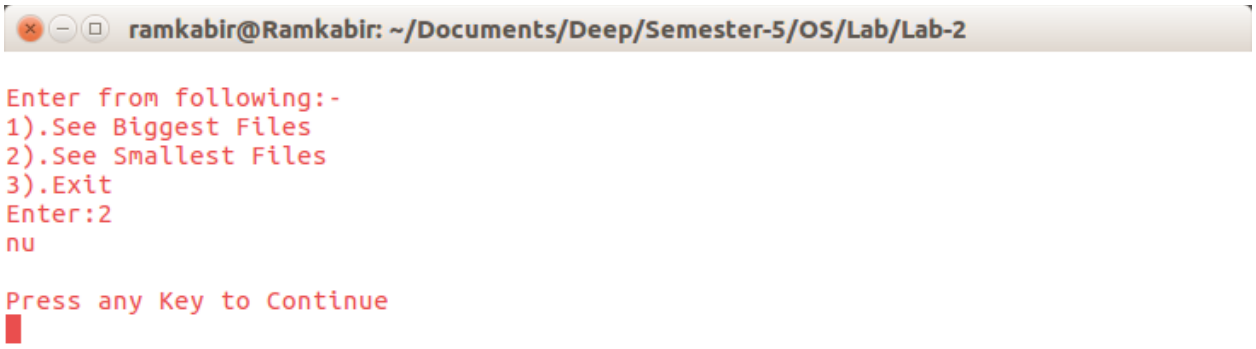
Code:-

```
#!/bin/bash

#Created by Deep C. Patel - 1401010

clear

echo -e "\nEnter Path:"
read path

cd $path
clear

while true
do

    echo -e "\nEnter from following:-"
    echo "1).See Biggest Files"
    echo "2).See Smallest Files"
    echo "3).Exit"
    echo -e "Enter:\c"
    read choice

    case $choice in
         [1])
                set `ls -lR | awk '{ print $1 "\t" $5 "\t" $9
}'| awk '/^[-]/' | awk '{ print $2 "\t" $3 }' | sort -nr`

                echo $2

                stop=$1

                shift 2

                for true
                do
                    if [ "$1" = "$stop" ]
                    then

                        echo $2
                        shift 2

                    else

                        break;

                    fi
```

```
                done

                echo -e "\nPress any Key to Continue"
                read ch
                continue;;

        [2])
                set `ls -lR | awk '{ print $1 "\t" $5 "\t" $9
}'| awk '/^[-]/' | awk '{ print $2 "\t" $3 }' | sort -n `

                echo $2

                stop=$1

                shift 2

                for true
                do
                    if [ "$1" = "$stop" ]
                    then

                        echo $2
                        shift 2

                    else

                        break;

                    fi

                done

                echo -e "\nPress any Key to Continue"
                read ch
                continue;;

        [3]) exit;;

        *)
            echo -e "\nImproper Choice, Press any Key to
Continue"
            read ch
            continue;;

    esac
done
```

Screenshot:-


ramkabir@Ramkabir: ~/Documents/Deep/Semester-5/OS/Lab/Lab-2
Enter from following:-
1).See Biggest Files
2).See Smallest Files
3).Exit
Enter:1
Cosmos
Press any Key to Continue


ramkabir@Ramkabir: ~/Documents/Deep/Semester-5/OS/Lab/Lab-2
Enter from following:-
1).See Biggest Files
2).See Smallest Files
3).Exit
Enter:2
nu
Press any Key to Continue

3).

Run all the programs related to process creation/termination shown in the lectures (Also given at ftp).

Code 1:- f1.c

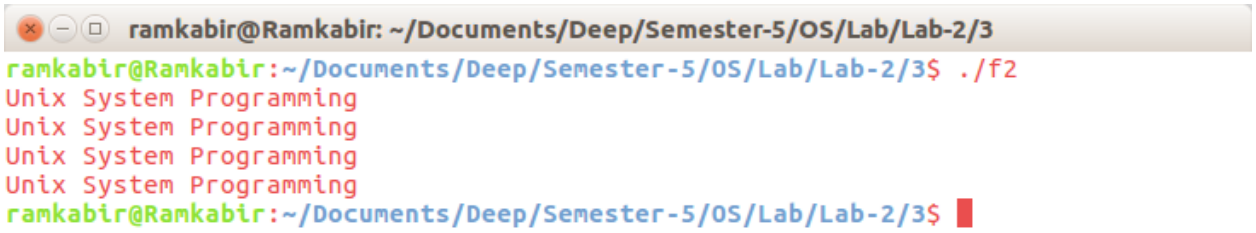
```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int main(void)
{
    int index;
    for (index = 1; index < 4; index++)
        fork();
    printf("Unix System Programming\n");
    exit(0);
}
```

Screenshot:-

Code 2:- f2.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int main(void)
{
    fork();
    fork();
    printf("Unix System Programming\n");
    exit(0);
}
```

Screenshot:-

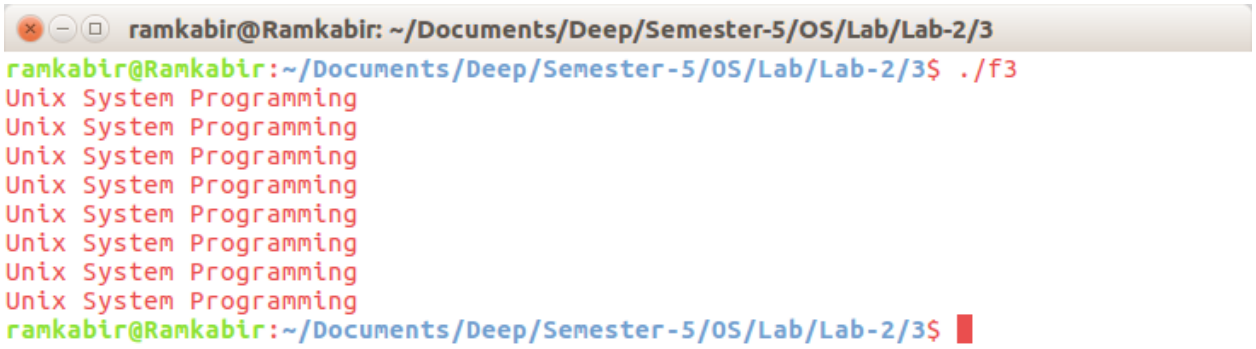


Code 3:- f3.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int main(void)
{
    fork();
```

```
    fork();
    fork();
    printf("Unix System Programming\n");
    exit(0);
}
```

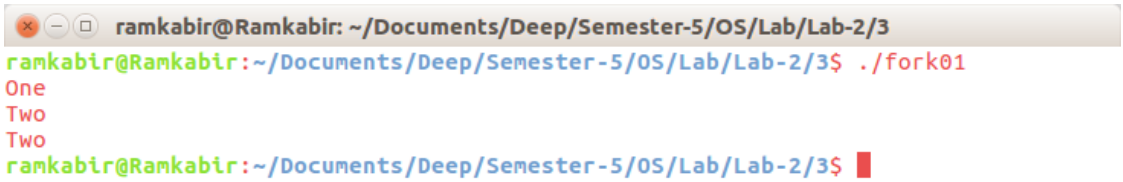
Screenshot:-



Code 4:- fork01.c

```
#include <sys/types.h>
#include <unistd.h> /* for fork() */

main()
{
pid_t pid;  /*holds process-id in parent*/

printf("One\n");
pid=fork();
printf("Two\n");
}
```

Screenshot:-



Code 5:- fork02.c

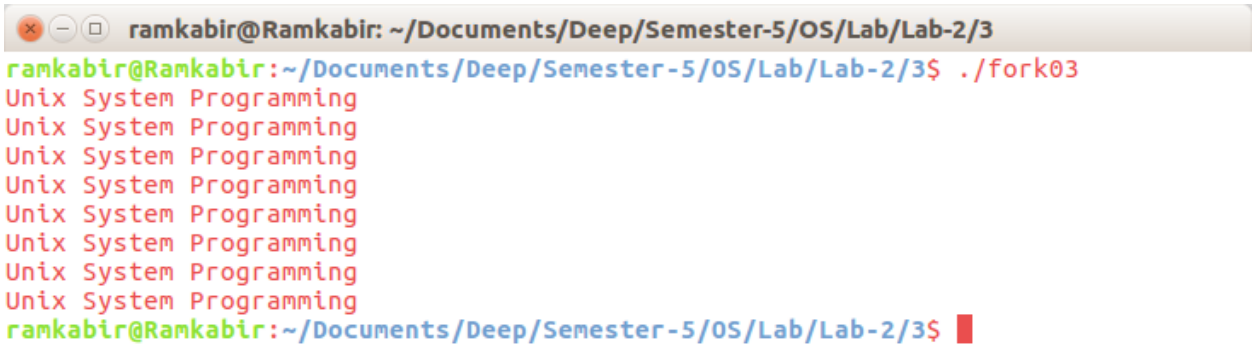
```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

void main(void)
{
pid_t pid;  /*holds process-id in parent*/
char answer[1];

printf("Just One Process so far\n");
pid=fork(); /* create new process */

printf("\nPress Enter");
gets(answer);

if(pid == 0)
    printf("I am the child");
else if(pid > 0)
    printf(" I am the parent, child has pid %d\n", pid);
else
    printf("Fork returned error code, no child\n");
}
```

Screenshot:-

Code 6:- fork03.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

void doit(void)
{
pid_t pid;
fork();
fork();
printf("Unix System Programming\n");
return;
}

int main(void)
{
    doit();
    printf("Unix System Programming\n");
    exit(0);
}
```

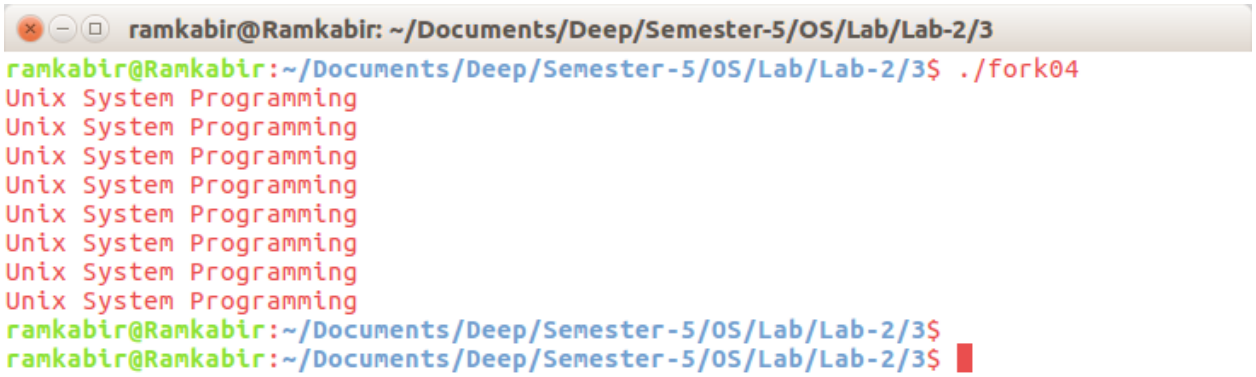
Screenshot:-



Code 7:- fork04.c

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int main(void)
{
    int index;
    for (index = 1; index < 4; index++)
        fork();
    printf("Unix System Programming\n");
    exit(0);
}
```

Screenshot:-



---

4).

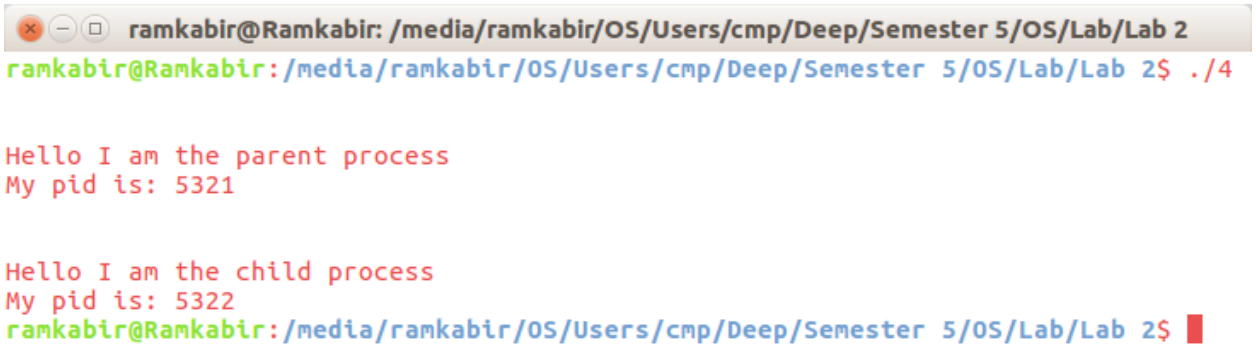
Create one child process and make sure that child process runs first and then parent executes and vice versa (if it is possible).

Code:-Parent Process before Child Process

```c
#include<stdio.h>
#include<sys/types.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>

int main()
{
    pid_t pid;
    int pi,status;
    char buf[100];
```

```
    bzero(buf,strlen(buf));

    pid=fork();

    //pi=wait(&status);


    if(pid<0)
    {
        printf("\n Error ");
        exit(1);
    }
    else if(pid==0)
    {
        sprintf(buf, "\n\nHello I am the child process ");
        write(1, buf, strlen(buf));

        pi=getpid();

        bzero(buf,strlen(buf));

        sprintf(buf, "\nMy pid is: %d\n", pi);
        write(1, buf, strlen(buf));

        exit(0);
    }
    else
    {

        sprintf(buf, "\n\nHello I am the parent process ");
        write(1, buf, strlen(buf));

        pi=getpid();

        bzero(buf,strlen(buf));

        sprintf(buf, "\nMy pid is: %d\n", pi);
        write(1, buf, strlen(buf));

        exit(0);
    }
    //getpid() to retrieve the process ID assigned to this
process;
    return 0;
}
```

Screenshot:-



Code:-Child Process before Parent Process

```
#include<stdio.h>
#include<sys/types.h>
#include<stdlib.h>
#include<string.h>
#include<unistd.h>

int main()
{
    pid_t pid;
    int pi,status;
    char buf[100];

    bzero(buf,strlen(buf));

    pid=fork();

    pi=wait(&status);


    if(pid<0)
```

```
    {
        printf("\n Error ");
        exit(1);
    }
    else if(pid==0)
    {
        sprintf(buf, "\n\nHello I am the child process ");
        write(1, buf, strlen(buf));

        pi=getpid();

        bzero(buf,strlen(buf));

        sprintf(buf, "\nMy pid is: %d\n", pi);
        write(1, buf, strlen(buf));

        exit(0);
    }
    else
    {

        sprintf(buf, "\n\nHello I am the parent process ");
        write(1, buf, strlen(buf));

        pi=getpid();

        bzero(buf,strlen(buf));

        sprintf(buf, "\nMy pid is: %d\n", pi);
        write(1, buf, strlen(buf));

        exit(0);
    }
    //getpid() to retrieve the process ID assigned to this
process;
    return 0;
}
```

Screenshot:-

---

5).

Create one child process and make it as zombie. (Use sleep(time);).

Code:-

```
//ps -e -o pid,ppid,stat,cmd

#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int main()
{

  pid_t pid;

  char buf[50];
```

```c
  bzero(buf,strlen(buf));

  pid = fork ();

  if (pid != 0)
  {

    pid=getpid();

    sprintf(buf, "\nParent pid is: %d\n", pid);
    write(1, buf, strlen(buf));

    pause();    //Pausing the parent
  }
  else
  {
    sprintf(buf, "\nChild pid is: %d\n", pid);
    write(1, buf, strlen(buf));

    sleep(2);   //Letting child live for 2 seconds before
becoming Zombie

    exit (0);
  }

  return 0;
}
```

Screenshot:-