Experiment No. 9

Name : Rinoy Kuriyakose
Roll No: 56

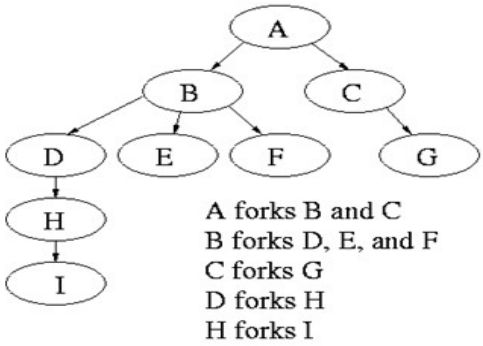
Aim:

Familiarization of various system calls in UNIX operating system

(a) Program to accept the limiting value ‘n’as input and generate the Fibonacci sequence of n numbers using the child process while the parent process generate the first n prime number

(b) Generate an N level hierarchy of processes and also display the parent id of process.

(c)



Program (a)

```
#include<stdio.h>
#include<unistd.h>
#include<sys/wait.h>
void fibonacci(int n){
   int a =0 ,b = 1,c,i;
   printf(" %d  %d ",a,b);
   c=a+b;
   while(a+b<=n) {
      c = a + b;
      printf(" %d ",c);
      a = b;
      b = c;
  }
}

void prime(int n){
   int i,j,flag;
   for (i=2; i<=n; i++){
      flag=1;
      for (j=2; j<=i/2; j++){
         if(i%j == 0){
            flag=0;
            break;
         }
```

```
        }
        if(flag==1){
            printf(" %d ",i);
        }
    }
}

int main(){
    int n;
    printf("\n\nEnter the value of n :");
                scanf("%d",&n);
                if(fork() ==0 ){
                    printf("\n\nChild Process : Fibonacci Series \n");
                    fibonacci(n);
                }else{
                    wait(NULL);
                    printf("\n\nParent Process : Prime Numbers \n");
                    prime(n);
                }
                printf("\n");

}
```

Output:

```
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$ ./program#a


Enter the value of n :20


Child Process : Fibonacci Series
 0  1  1  2  3  5  8  13


Parent Process : Prime Numbers
 2  3  5  7  11  13  17  19
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$ ./program#a


Enter the value of n :50


Child Process : Fibonacci Series
 0  1  1  2  3  5  8  13  21  34


Parent Process : Prime Numbers
 2  3  5  7  11  13  17  19  23  29  31  37  41  43  47
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$
```

Program (b)

```
#include<stdio.h>
#include<unistd.h>
#include<sys/wait.h>


void main(){
   int n,i;
   printf("Enter the value of N : ");
   scanf("%d",&n);
   for(i =0; i<n ; i++){
      if(fork() == 0){
         printf("Level %d:  Child pid: %d   Parent pid: %d \n",i,getpid(),getppid());
      }else{
         wait(NULL);
         break;
      }
   }
}
```

Output:

```
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$ ./program#b
Enter the value of N : 8
Level 0:  Child pid: 2084, Parent pid: 2083
Level 1:  Child pid: 2085, Parent pid: 2084
Level 2:  Child pid: 2086, Parent pid: 2085
Level 3:  Child pid: 2087, Parent pid: 2086
Level 4:  Child pid: 2088, Parent pid: 2087
Level 5:  Child pid: 2089, Parent pid: 2088
Level 6:  Child pid: 2090, Parent pid: 2089
Level 7:  Child pid: 2091, Parent pid: 2090
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$ ./program#b
Enter the value of N : 14
Level 0:  Child pid: 2093, Parent pid: 2092
Level 1:  Child pid: 2094, Parent pid: 2093
Level 2:  Child pid: 2095, Parent pid: 2094
Level 3:  Child pid: 2096, Parent pid: 2095
Level 4:  Child pid: 2097, Parent pid: 2096
Level 5:  Child pid: 2098, Parent pid: 2097
Level 6:  Child pid: 2099, Parent pid: 2098
Level 7:  Child pid: 2100, Parent pid: 2099
Level 8:  Child pid: 2101, Parent pid: 2100
Level 9:  Child pid: 2102, Parent pid: 2101
Level 10:  Child pid: 2103, Parent pid: 2102
Level 11:  Child pid: 2104, Parent pid: 2103
Level 12:  Child pid: 2105, Parent pid: 2104
Level 13:  Child pid: 2106, Parent pid: 2105
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$
```

Program(c)

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
void main() {
  printf("Parent A: %d\n", getpid());
  if (fork() == 0) {
     printf(" Child B: %d forked by Parent A: %d\n", getpid(), getppid());
     if (fork() == 0) {
        printf(" Child D: %d forked by Parent B: %d\n", getpid(), getppid());
        if (fork() == 0) {
           printf(" Child H: %d forked by Parent D: %d\n", getpid(), getppid());
           if (fork() == 0) {
              printf(" Child I: %d forked by Parent H: %d\n", getpid(), getppid());
            }else{
             wait(NULL);
            }
        }else{
           wait(NULL);
        }
     }else if (fork() == 0) {
        printf(" Child E: %d forked by Parent B: %d\n", getpid(), getppid());
     }else if (fork() == 0) {
        printf(" Child F: %d forked by Parent B: %d\n", getpid(), getppid());
     }else{
        wait(NULL);
     }
  }else if (fork() == 0) {
     printf(" Child C: %d forked by Parent A: %d\n", getpid(), getppid());
     if (fork() == 0) {
       printf(" Child G: %d forked by Parent C: %d\n", getpid(), getppid());
     }else{
       wait(NULL);
     }
  }else{
     wait(NULL);
  }
}
```

Output:

```
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$ ./program#c
Parent A: 2113
 Child B: 2114 forked by Parent A: 2113
 Child C: 2115 forked by Parent A: 2113
 Child D: 2116 forked by Parent B: 2114
 Child E: 2117 forked by Parent B: 2114
 Child H: 2120 forked by Parent D: 2116
 Child G: 2119 forked by Parent C: 2115
 Child F: 2118 forked by Parent B: 2114
 Child I: 2121 forked by Parent I: 2120
rinoy2002@rinoy-Hyper-V:~/Desktop/OS Lab/Day 9 System Calls$
```