PROGRAM CODE

```
#include<stdio.h>
#include<unistd.h>

void fibonocci(int n)
{
    int a=0,b=1;
    int c;
    printf("0   1");
    for(int i=2;i<n;i++)
    {
        c=a+b;
        printf("\t%d",c);
        a=b;
        b=c;
    }
    printf("\n");
}
void prime(int n)
{
    for(int i=2;i<n;i++)
    {
        int flag=0;
        for(int j=2;j<i/2;j++)
        {
            if(i%j==0)
            {
                flag=1;
            }
        }
        if(flag==0)
        {
            printf("%d\t",i);
        }
    }
}

void main()
{
    int n;
    printf("\nenter the number for prime and fibonnocci\n");
    scanf("%d",&n);
    if(fork()==0)
    {
        printf("\nfibonnocci sequence for child:-\n");
        fibonocci(n);
```

```
    }
    else
    {
        printf("\nprime sequence for parent:-\n");
        prime(n);
    }
}
```

OUTPUT

```
sahal@kali:~/bash_script$ ./a.out

enter the number for prime and fibonnocci
10

prime sequence for parent:-
2       3       4       5       7
fibonnocci sequence for child:-
0   1   1       2       3       5       8       13      21
34
```

PROGRAM CODE

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

void main()
{
    int n;
    printf("\nenter the level\n");
    scanf("%d",&n);
    printf("MAIN PARENT PROCESS pid=%d at level 0\n",getpid());
    for(int i=1;i<n;i++)
    {
        if(fork()==0)
        {
            printf("child_pid=%d,parent_pid=%d at
level=%d\n",getpid(),getppid(),i);
        }
        else if(fork()==0)
        {
            printf("child_pid=%d,parent_pid=%d at
level=%d\n",getpid(),getppid(),i);
        }
        else
```

```
        {
            wait(NULL);
            break;
        }

    }
}
```

OUTPUT

```
sahal@kali:~/bash_script$ ./a.out

enter the level
4
MAIN PARENT PROCESS pid=2389 at level 0
child_pid=2393,parent_pid=2389 at level=1
child_pid=2394,parent_pid=2389 at level=1
child_pid=2397,parent_pid=2393 at level=2
child_pid=2395,parent_pid=2393 at level=2
child_pid=2398,parent_pid=2394 at level=2
child_pid=2396,parent_pid=2394 at level=2
child_pid=2401,parent_pid=2396 at level=3
child_pid=2402,parent_pid=2397 at level=3
child_pid=2399,parent_pid=2397 at level=3
child_pid=2405,parent_pid=2396 at level=3
child_pid=2403,parent_pid=2398 at level=3
child_pid=2406,parent_pid=2395 at level=3
child_pid=2404,parent_pid=2395 at level=3
child_pid=2400,parent_pid=2398 at level=3
```

PROGRAM CODE

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

void main()
{
    printf("\nA:main parent pid=%d\n",getpid());
    if(fork()==0)
    {
        printf("B:%d forked by %d\n",getpid(),getppid());
        if(fork()==0)
        {
            printf("D:%d forked by %d\n",getpid(),getppid());
            if(fork()==0)
```

```
            {
                printf("H:%d forked by %d\n",getpid(),getppid());
                if(fork()==0)
                {
                    printf("I:%d forked by
%d\n",getpid(),getppid());
                }
                else
                {
                    wait(NULL);
                }
            }
            else
            {
                wait(NULL);
            }
        }
        else if(fork()==0)
        {
            printf("E:%d forked by %d\n",getpid(),getppid());
        }
        else if(fork()==0)
        {
            printf("F:%d forked by %d\n",getpid(),getppid());
        }
        else
        {
            wait(NULL);
        }
    }
    else if(fork()==0)
    {
        printf("C:%d forked by %d\n",getpid(),getppid());
        if(fork()==0)
        {
            printf("G:%d forked by %d\n",getpid(),getppid());
        }
        else
        {
            wait(NULL);
        }
    }
    else
    {
        wait(NULL);
    }
}
```

OUTPUT

```
sahal@kali:~/bash_script$ ./a.out

A:main parent pid=2341
C:2343 forked by 2341
B:2342 forked by 2341
G:2345 forked by 2343
D:2344 forked by 2342
F:2347 forked by 2342
E:2346 forked by 2342
H:2348 forked by 2344
I:2349 forked by 2348
```