**OS lab assignment 7**

**Agnim Gupta**

**2028082**

**A-23 CSSE**

**Question 1(a)**

#include  <stdio.h>

#include  <unistd.h>

#include  <sys/types.h>

int main()

{

    pid\_t  pid1,  pid2,  pid3,  pid4;

    printf  ("Parent[P1]  of  all:  %d\n",  getpid  ());

    pid1 = fork();

    if (pid1 == 0)

    {

        printf  ("Child[P2]  with  id:  %d  and  it's  Parent  id:  %d\n",  getpid  (), getppid  ());

        pid2 = fork();

        if (pid2 == 0)

        {

            printf  ("Child[P4]  with  id:  %d  and  it's  Parent  id:  %d\n", getpid  (),  getppid  ());

        }

    }

    if (pid1 > 0)

    {

        pid3 = fork ();

        if (pid3 == 0)

        {

            printf  ("Child[P3]  with  id:  %d  and  it's  Parent  id:  %d\n", getpid  (),  getppid  ());

            pid4 == fork (); if (pid4 == 0)

            {

                printf  ("Child[P5]  with  id:  %d  and  it's  Parent  id:  %d\n", getpid (), getppid ());

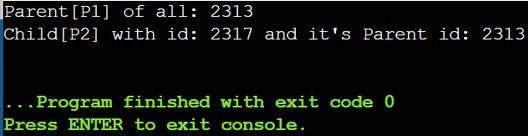
            }

        }

    }

}

**Output**

****

**Question 1(b)**

#include  <stdio.h>

#include  <unistd.h>

#include  <sys/types.h>

int main()

{

    pid\_t  pid1,  pid2,  pid3,  pid4;

    printf  ("Parent[P1]  of  all:  %d\n",  getpid  ());

    pid1 = fork();

    if (pid1 == 0)

    {

        printf  ("Child[P2]  with  id:  %d  and  it's  Parent  id:  %d\n",  getpid  (), getppid  ());

        pid2 = fork ();

        if  (pid2  ==  0)

        {

            printf  ("Child[P4]  with  id:  %d  and  it's  Parent  id:  %d\n",  getpid  (), getppid  ());

        }

    }

    if (pid1 > 0)

    {

        pid3 = fork (); if  (pid3  ==  0)

        {

            printf  ("Child[P3]  with  id:  %d  and  it's  Parent  id:  %d\n",  getpid  (), getppid  ());

            pid4 == fork (); if  (pid4  ==  0)

            {

                printf  ("Child[P5]  with  id:  %d  and  it's  Parent  id:  %d\n", getpid (), getppid ());

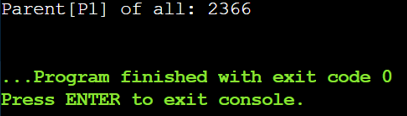
            }

        }

    }

}

**Output**

****

**Question 1(c)**

#include  <stdio.h>

#include  <unistd.h>

#include  <sys/types.h>

int main ()

{

    for(int  i=0;i<2;i++)

    {

        int  temp=fork(); if(temp==0)

        {

            printf("%d  \n",getpid());

            return 0;

        }

    }

    int  temp=fork(); if(temp==0)

    {

        printf("%d\n",  getpid());

        int  temp=fork();

        if(temp==1)

        {

            printf("%d\n",getpid());

            return  0;

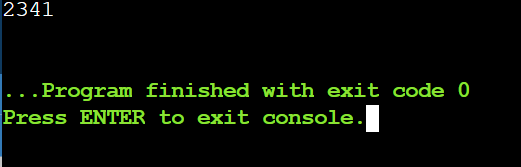
        }

        return  0;

    }

}

**Output**



**Question 3**

#include  <stdio.h>

#include  <unistd.h>

int main ()

{

    int  num;

    printf  ("Enter  the  Upper  limit  :  ");

    scanf  ("%d",  &num);

    int  sum  =  0,  count  =  0;

    int  n;

    n = fork ();

    if (n > 0)

    {

        for  (int  i  =  0;  i  <  num;  i++)

        {

            count++;

            sum  =  sum  +  count;

        }

        printf  ("This  is  Parent  Process  \n:  ");

        printf  ("Sum  of  numbers  is  %d  \n\n",  sum);

    }

    else

    {

        printf  ("This  is  Child  Process  :  \n");

        printf  ("Prime  numbers  between  0  to  %d  were  :  ",  num);

        int  i,  a  =  1,  count; while (a <= num)

        {

            count  =  0;

            i  =  2;

            while (i <= a / 2)

            {

                if (a % i == 0)

                {

                count++; break;

                }

                i++;

            }

            if  (count  ==  0  &&  a  !=  1)

            {

                printf  ("  %d  ",  a);

            }

            a++;

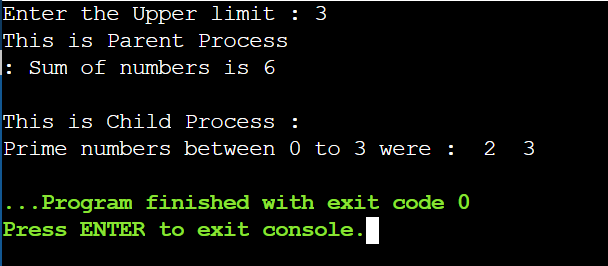
        }

    }

    return  0;

}

**Output**



**Question 4**

#include  <stdio.h>

#include  <stdlib.h>

#include  <unistd.h>

// Driver code

int main()

{

    int  pid,  pid1,  pid2;

    pid  =  fork();

    if (pid == 0)

    {

        sleep(3);

        printf("child[1]  -->  pid  =  %d  and  ppid  =  %d\n", getpid(), getppid());

    }

    else {

        pid1 = fork();

        if (pid1  ==  0)

        {

            sleep(2);

            printf("child[2]  -->  pid  =  %d  and  ppid  =  %d\n", getpid(),  getppid());

        }

        else {

        pid2  =  fork(); if (pid2 == 0) {

        printf("child[3]  -->  pid  =  %d  and  ppid  =  %d\n", getpid(),  getppid());

        }

        else

        {

            sleep(3);

            printf("parent  -->  pid  =  %d\n",  getpid());

        }

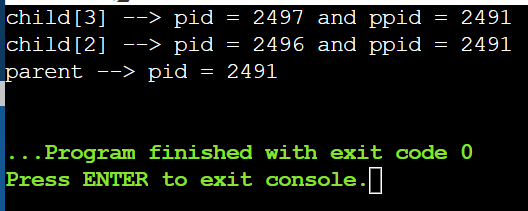
        }

    }

    return  0;

}

**Output**



**Question 5**

#include<unistd.h>

#include<sys/types.h>

#include<stdio.h>

#include<sys/wait.h>

int main()

{

    pid\_t  p;

    printf("before  fork\n");

    p=fork();

    if(p==0)

    {

        printf("Child  having  id  %d\n",getpid());

        printf("Parent  id  is  %d\n",getppid());

    }

    else

    {

        wait(NULL);

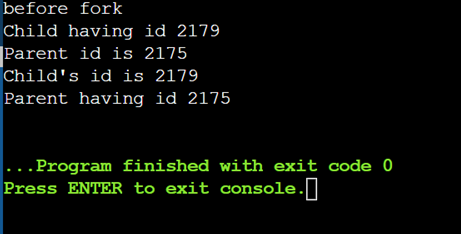
        printf("Child's  id  is  %d\n",p);

        printf("Parent  having  id  %d\n",getpid());

    }

}

**Output**

****