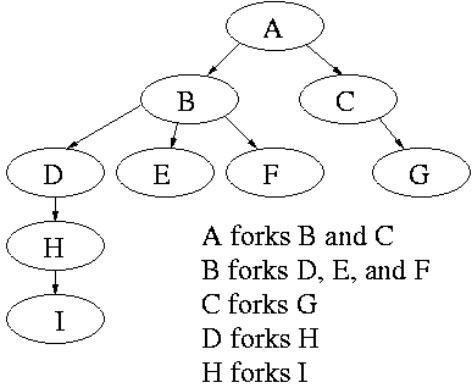
**Cycle - 2**

# Experiment No : 2.3

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



Program:

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

void forkCreation(char process, int level) {

pid\_t pid = fork();

if (pid < 0) {

printf("Fork failed\n");

return;

} else if (pid == 0) {

printf("Process: %c (PID: %d, Parent PID: %d, Level: %d)\n",

process, getpid(), getppid(), level);

fflush(stdout);

if (level < 4) {

switch (process) {

case 'A':

forkCreation('B', level + 1);

forkCreation('C', level + 1);

break;

case 'B':

forkCreation('D', level + 1);

forkCreation('E', level + 1);

forkCreation('F', level + 1);

break;

case 'C':

forkCreation('G', level + 1);

break;

case 'D':

forkCreation('H', level + 1);

break;

case 'H':

forkCreation('I', level + 1);

break;

default:

break;

}

}

exit(0);

}

}

int main() {

forkCreation('A', 0);

return 0;

}

Output:

Process: A (PID: 17033, Parent PID: 1, Level: 0)

Process: B (PID: 17034, Parent PID: 17033, Level: 1)

Process: F (PID: 17038, Parent PID: 1, Level: 2)

Process: E (PID: 17037, Parent PID: 1, Level: 2)

Process: D (PID: 17036, Parent PID: 1, Level: 2)

Process: C (PID: 17035, Parent PID: 1, Level: 1)

Process: H (PID: 17039, Parent PID: 1, Level: 3)

Process: G (PID: 17040, Parent PID: 1, Level: 2)

Process: I (PID: 17041, Parent PID: 1, Level: 4)

# Experiment No : 2.4

Write a program to create four processes (1 parent and 3 children) where they terminate in a sequence as follows:

* 1. Parent process terminates at last
  2. First child terminates before parent and after second child.
  3. Second child terminates after last and before first child.
  4. Third child terminates first.

Program:

#include<stdio.h>

#include<unistd.h>

#include<sys/wait.h>

#include<stdlib.h>

int main(){

int Child1, Child2, Child3 ;

Child1 = fork() ;

if (Child1 == 0)

{

sleep(3) ;

printf("PID of Child 1 is %d\n", getpid()) ;

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

}

else

{

Child2 = fork() ;

if (Child2 == 0)

{

sleep(2) ;

printf("PID of Child 2 is %d\n", getpid()) ;

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

}

else

{

Child3 = fork() ;

if (Child3 == 0)

{

printf("PID of Child 3 is %d\n", getpid()) ;

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

}

else

{

sleep(4) ;

printf("PID of main is %d\n", getpid()) ;

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

}

}

}

}

Output:

PID of Child 3 is 2981

Parent PID is 2978

PID of Child 2 is 2980

Parent PID is 2978

PID of Child 1 is 2979

Parent PID is 2978

PID of main is 2978

Parent PID is 2971