**1. Echo arguments sent through Command-Line**

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/types.h>

int main(int argc, char\* argv){

if(argc < 2){

printf("Invalid number of arguments entered...\n”);

exit(1);

}

fork();

for(int i = 1; i <= argc; i++)

printf("%s \n", argv[i]);

}

**2. Create a child process and make it execute myecho.exe**

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/types.h>

int main(int argc, char\* argv){

if(argc != 2){

printf("Invalid number of arguments entered...\n);

exit(1);

}

int this\_pid, child\_pid;

char\* file = 'myecho.exe';

this\_pid = getpid();

printf("pid: %d Parent process : About to fork this process...\n", this\_pid);

child\_pid = fork();

if(child\_pid != 0)

printf("pid: %d Parent process : Created a child process with pid : %d\n", this\_pid, (int)child\_pid);

else{

this\_pid = getpid();

printf("pid: %d Child Process, executing myecho.exe\n", this\_pid);

execl(file, argv[1]);

}

printf("Exiting...\n");

exit(0);

}

**3. Create a child process to create a text file. The parent process creates another child process which displays the text file...**

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

int main(int argc, char\* argv){

if(argc != 2){

printf("Invalid number of arguments...\n");

exit(1);

}

int this\_pid, child\_pid;

int f;

int first\_child\_status, second\_child\_status;

this\_pid = getpid();

char buffer;

printf("pid: %d Parent process : About to fork this process...\n", this\_pid);

child\_pid = fork();

if(child\_pid != 0){

printf("pid: %d Parent process : Created the first child process with pid : %d\n", (int)this\_pid, (int)child\_pid);

printf("Waiting for the first child process to finish creating the file...\n");

wait(&first\_child\_status);

printf("Firstchild process has finished it's execution with status: %d", first\_child\_status);

child\_pid = fork();

if(child\_pid != 0){

printf("pid: %d Parent process : Created the second child process with pid : %d\n", this\_pid, child\_pid);

printf("Waiting for the second child process to finish displaying the file...\n");

wait(&second\_child\_status);

printf("Secondchild process has finished it's execution with status: %d", second\_child\_status);

}

else {

while(read(f, buffer, 1) > 0)

printf(buffer);

}

printf(“Exiting….\n );

exit(0);

}

else {

this\_pid = getpid();

printf("pid: %d Child Process, creatng a new file\n", this\_pid);

creat("test.txt", S\_IWRITE | S\_IREAD);

f = open("test.txt", O\_RDWR, 0777);

if(f == -1){

printf("There was a problem in opening the file...\n");

exit(1);

}

write(f, argv[1], strlen(argv[1]));

close(f);

}

printf("Exiting...\n);

exit(0);

}