This homework focuses on learning how to create processes in Linux/Unix.

1. Using Fork() and Exec() or Clone(), create four child processes. Load the Hello" program in each process after creation (same program). Each child program should print its own PID or some other parameters distinguishing its execution.

2. Each process should run and return to the parent process where the parent is waiting to terminate. The parent should only terminate after all children returned from execution.

Run your program, capture its output and create single PDF file with source code and its output and submit in Blackboard. You can run it on CYGWIN, or any Linux system. The code should be written in C.

Source Code (Linux)

Notes:
This will not run correctly if path provided to execlp is incorrect.

```
/* Hello World program Homework #3- Linux GCC*/

#include<stdio.h>
#include<stdlib.h>

int main()
{


	pid_t pid_parent, pid;
	int i;

	pid_parent = getpid();
	printf("\nParent process PID: %d\n", pid_parent);

	for (i = 0; i < 4; i++){
		//fork child process
		pid = fork();

		if (pid < 0){ // error
			printf(stderr, "Fork failed");
			exit(1);
		} else if (pid == 0){ //child
			printf("\nHello World!  I'm the CHILD!  My PID is: %d\n", getpid());
			execlp("/home/user/git/Operating-
Systems/Homeworks/Homework2/Homework2Linux/Debug/Homework2Linux", "Homework2Linux", NULL);
		} else {
			wait(NULL);
		}

	}

	printf("\n\nHello World!  I'm the PARENT!  My PID is: %d\n", getpid());

	return(0);
```

}

The code above calls the “Hello” program from Homework 2 (below).

Notes:
“input.txt” and “output.txt” will be created/ located in the same directory of the parent calling program – Homework 3.

```
/* Hello World program -Homework 2- Linux GCC*/

#include<stdio.h>
#include<stdlib.h>

int main()
{

	char *outputFilename = "output.txt";
	char ch;

	FILE *ifp, *ofp;

	printf("Hello World - Homework 2");

	ifp = fopen("input.txt", "a");

	if (ifp == NULL){
		fprintf(stderr, "Can't open input file input.txt!\n");
		exit(1);
	}

	ofp = fopen(outputFilename, "a");

	if (ofp == NULL) {
		fprintf(stderr, "Can't open output file %s!\n",
	  outputFilename);
		exit(1);
	}

	while (1) {
	ch = fgetc(ifp);

	if (ch == EOF)
	   break;
	else
	   putc(ch, ofp);
	}

	fprintf(ifp, "..appending text to INPUT file.");
	fprintf(ofp, "..appending text to OUTPUT file.");

	fclose(ifp);
	fclose(ofp);

	return 0;
}
```

Output

* The output was run 3 times in order to ensure that Parent process would always terminate last.

Test#1

```
Parent process PID: 6771

Hello World!  I'm the CHILD!  My PID is: 6775
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6778
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6779
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6780
Hello World - Homework 2

Hello World!  I'm the PARENT!  My PID is: 6771
```

Test#2

```
Parent process PID: 6832

Hello World!  I'm the CHILD!  My PID is: 6833
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6837
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6838
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6840
Hello World - Homework 2

Hello World!  I'm the PARENT!  My PID is: 6832
```

Test#3

```
Parent process PID: 6854

Hello World!  I'm the CHILD!  My PID is: 6856
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6858
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6859
Hello World - Homework 2
Hello World!  I'm the CHILD!  My PID is: 6860
Hello World - Homework 2

Hello World!  I'm the PARENT!  My PID is: 6854
```