

Operating Systems 1

Tanmay Garg CS20BTECH11063

Programming Assignment 1

Program Design:

- We first take input from the user from command line which should be a positive integer
- Then we call the fork() system call
- If it is not able to fork() then it gives pid the value less than 0 and throws an error
- If it is able to fork() then it makes pid as 0 and now the child process is working
 - When the child process is started it runs a while loop to begin with the Collatz Conjecture
 - If n is even, $n \leftarrow n/2$
 - If n is odd, then n becomes $n \leftarrow 3n + 1$
 - The process continues until $n = 1$
- In all other situations it calls the wait() system call to wait for the child process to get over
- The child process outputs the list of numbers generated by the Collatz Conjecture

```
C: Assign1Src_CS20BTECH11063.c > main(int, char *[])
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <math.h>
4  #include <sys/types.h>
5  #include <unistd.h>
6  #include <sys/wait.h>
7
8  int main(int argc, char *argv[])
9  {
10     //If number of arguments is not 2 then throw an error and terminates the program
11     if(argc != 2)
12     {
13         printf("Incorrect number of inputs\n");
14         printf("Input: final_exec n (where n is the number)\n");
15         return EXIT_FAILURE;
16     }
17     //Our number n on which Collatz conjecture will be tested
18     int n = atoi(argv[1]);
19     //If n is less than zero then ask for input again
20     while(n <= 0)
21     {
22         printf("Please enter number greater than 0: ");
23         scanf("%d", &n);
24     }
25
26     //forking a child process
27     pid_t pid = fork();
28     if (pid < 0)
29     {
30         //If error occurs
31         fprintf(stderr, "Fork Failed\n");
32         return EXIT_FAILURE;
33     }
34     else if (pid == 0)
35     {
36         //Child Process
37         printf("Child Process is working: ");
38         printf("%d ", n);
39         while (n != 1)
40         {
41             //If n is even it is divided by 2
42             if (n % 2 == 0)
43             {
44                 n = n / 2;
45             }
46             //else n is 3n+1
47             else
48             {
49                 n = 3 * n + 1;
50             }
51             printf("%d ", n);
52         }
53         printf("\nChild Process is completed\n");
54     }
55     else
56     {
57         //Parent Process
58         //Parent waits for child process to be completed
59         printf("Parent Process is waiting for Child Process\n");
60         wait(NULL);
61         printf("Parent Process is completed\n");
62     }
63     return EXIT_SUCCESS;
64 }
```

Program Analysis:

- When the program performs `fork()` it assigns a Process ID to `pid`
- If the Process ID is 0 then it proceeds to the Child Process
- The Loop performs Collatz Conjecture on the number till it reaches the value 1
- The Program has been tested for different values such as 65, 40, 101 and all of them converge into the series 4, 2, 1
- If the Process ID is greater than 0 then it waits for the child process to be completed
- If the Process ID is less than 0 then it shows an error as the process could not be created and hence returns `EXIT_FAILURE`
- If the entire program works and gives a proper output, then it simply returns `EXIT_SUCCESS`
- In the situation when the number of input arguments in command line is not 2 then it throws an error and exits the program such as shown in the demo below
- In the situation when the input number is less than 0 then it throws an error and asks user to input positive number

```
bash: devel/setup.bash: No such file or directory
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ gcc -o final_exec Assgn1Src_CS20BTECH11063.c
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec
Incorrect number of inputs
Input: final_exec n (where n is the number)
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec 10 23
Incorrect number of inputs
Input: final_exec n (where n is the number)
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec -9
Please enter number greater than 0: 10
Parent Process is waiting for Child Process
Child Process is working: 10 5 16 8 4 2 1
Child Process is completed
Parent Process is completed
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec 10
Parent Process is waiting for Child Process
Child Process is working: 10 5 16 8 4 2 1
Child Process is completed
Parent Process is completed
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec 65
Parent Process is waiting for Child Process
Child Process is working: 65 196 98 49 148 74 37 112 56 28 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
Child Process is completed
Parent Process is completed
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec 40
Parent Process is waiting for Child Process
Child Process is working: 40 20 10 5 16 8 4 2 1
Child Process is completed
Parent Process is completed
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$ ./final_exec 101
Parent Process is waiting for Child Process
Child Process is working: 101 304 152 76 38 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
Child Process is completed
Parent Process is completed
LinuxItanmay@ubuntu:~/Desktop/OS1/Assignment 2$
```