[CSS553]

Name: Rahul Ranjan
Roll No: 20CS8016

Assignment 1

1. Write a code such that it takes input n and then the parent process (p) creates n number of child processes (c_1, c_2, c_3, ..., c_n) all of them as its direct descendent i.e. p is the parent of c_1, c_2, c_3 .... and also c_n. Each process including the parent should display the pid & ppid once in the terminal and also store the values in a common file (log.txt). The parent can create the file log.txt and all its child processes can get access it.

2. Write a code such that it takes input n and then the parent process p creates a child c_1, then c_1 creates c_2, c_2 creates c_3 ... and so on till c_n. Each process including the parent should display the pid & ppid once in the terminal and also store the values in a common file (log.txt). The parent can create the file log.txt and all its child processes can get access it.

3. Write a suitable code to speedup finding all prime numbers in a given range [1, N], creating M child processes by the parent process. First try to run it for two processes then generalize it for M child processes

4. Learn the use of ps, kill, cat, chmod ls, pwd, etc. commands.

5. Execute the balance update code discussed in the class and have a feel of race condition in OS.

Answer 1 – Single parent process has n child processes.

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void main(void){

  FILE *log;
  log = fopen("log1.txt", "w");
  pid_t x;
  int n = 5;
  int i;
  fprintf(log, "PID: %d\n\n", getpid());
  fflush(log);
  for (i = 0; i < n; i++){
    x = fork();
    if(x == 0){ // It's a child::
      fprintf(log, ">>  PID %d. PPID %d; Child no: %d\n",
      getpid(), getppid(), i);
      fflush(log);
      exit(0);
    }
  }
}
```

>> Output

Q1 – log.txt

```
PID: 16632
>>  PID 16633. PPID 16632; Child no: 0
>>  PID 16634. PPID 16632; Child no: 1
>>  PID 16635. PPID 16632; Child no: 2
>>  PID 16636. PPID 16632; Child no: 3
>>  PID 16637. PPID 16632; Child no: 4
```

Answer 2 – Chain of processes.

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

void create_process(FILE * log, int depth){
  pid_t x;
  if (depth == 0)
    exit(0);
  else {
    x = fork();
    if (x == 0){
      fprintf(log, ">> PID: %d PPID: %d @depth:%d\n",
      getpid(), getppid(), depth);
      fflush(log);
      create_process(log, depth-1);
    }
  }
}

void main(void){

  FILE * log;
  log = fopen("log2.txt", "w");
  pid_t x;
  int n = 6;
  create_process(log, n);
}
```

>> Output

Q2 – log.txt

```
>> PID: 17115 PPID: 17114 @depth:6
>> PID: 17116 PPID: 17115 @depth:5
>> PID: 17117 PPID: 17116 @depth:4
>> PID: 17118 PPID: 17117 @depth:3
>> PID: 17119 PPID: 17118 @depth:2
>> PID: 17120 PPID: 17119 @depth:1
```

Answer 3 – Checks if a given range number are prime or not by dividing the range of N numbers into M intervals and brute force to check if a number is prime or not.

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <stdbool.h>

bool isPrime(int n){
  if (n <= 1)
    return false;
  for (int i = 2; i < n; i++)
    if (n % i == 0) return false;
  return true;
}

void show_primes(const bool* is_prime, int start, int end){
  for (int i = start; i <= end; i++){
    if (is_prime[i])
      printf("%d - Prime\n", i);
    else printf("%d - Not Prime\n", i);
  }
  printf("\n");
}

void prime(bool* arr, int n, int i, int siblings){
  int start = (i-1)*n/siblings + 1;
  int end = i*n/siblings;
  printf("\n");
  printf("Start: %d | End: %d\n", start, end);
  for (int i = start; i <= (int)end; i++)
    arr[i] = isPrime(i);
  show_primes(arr, start, end);
}

void main (void){
  int n = 20;
  int cc = 4;
  bool is_prime[n+1];
  // scanf("%d", &n);
  is_prime[0] = is_prime[1] = 0;
  for (int i = 1; i <= (int)n; i++) is_prime[i] = 0;

  pid_t process;
  for (int i = 1; i <= cc; i++){
    process = fork();
    if (process == 0){
      prime(is_prime, n, i, cc);
      exit(0);
    }
  }
  return;
}
```

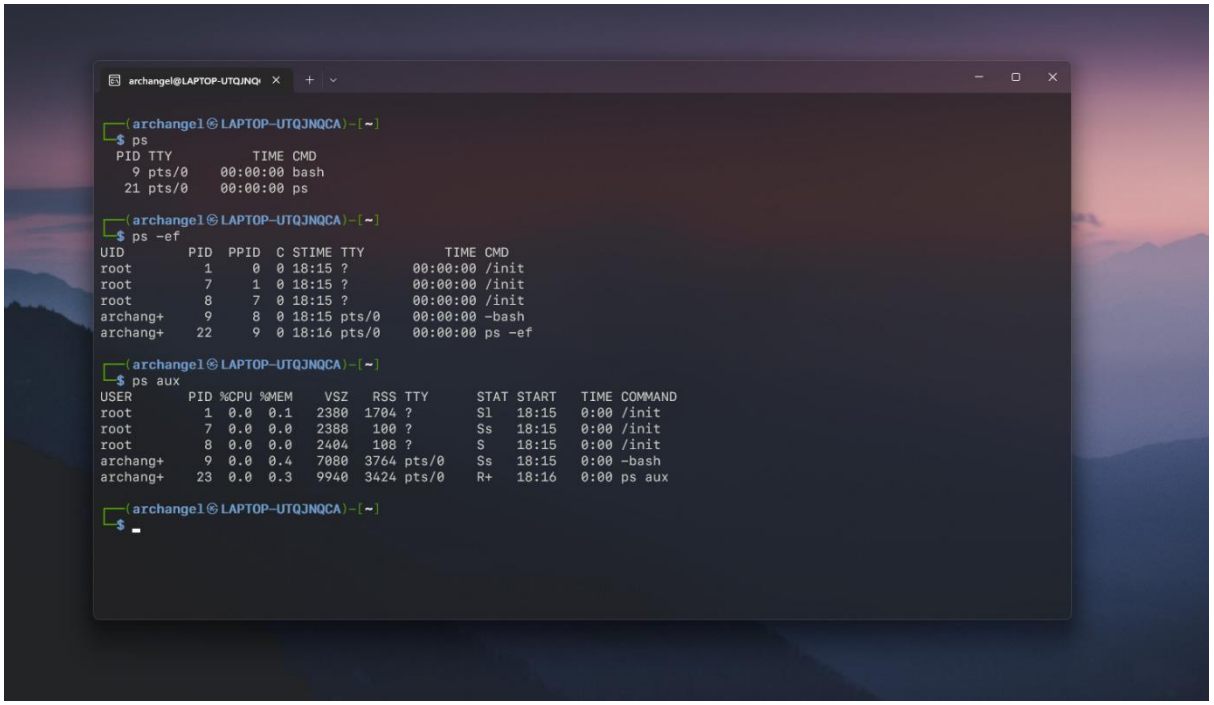
>> Output

```
goofynugtz@archangel:~/Downloads/OS Codes/Assignment 1$ ./a.out


Start: 1 | End: 5
1 - Not Prime
2 - Prime
3 - Prime
4 - Not Prime
5 - Prime

Start: 6 | End: 10
6 - Not Prime
7 - Prime
8 - Not Prime
9 - Not Prime
10 - Not Prime


Start: 11 | End: 15
11 - Prime
12 - Not Prime
13 - Prime
14 - Not Prime
15 - Not Prime


Start: 16 | End: 20
16 - Not Prime
17 - Prime
18 - Not Prime
19 - Prime
20 - Not Prime
```

Answer 4 – Utility commands

➔ `ps` – It's used to list running processes in system along with PIDs. Abberiviated for process status.

```
• goofynugtz@archangel:~$ ps
    PID TTY          TIME CMD
   4381 pts/0    00:00:00 bash
   4447 pts/0    00:00:00 ps
```



➔ `cat`: Preview contents of a file

```
• goofynugtz@archangel:~$ echo "Hello World" > file.txt
• goofynugtz@archangel:~$ cat file.txt
  Hello World
```

➔ `pwd`: Prints the current working directory

```
• goofynugtz@archangel:~$ pwd
  /home/goofynugtz
• goofynugtz@archangel:~$ cd Downloads/
• goofynugtz@archangel:~/Downloads$ pwd
  /home/goofynugtz/Downloads
```

➔ `kill`: Kills the process ID provided as argument.

➔ `chmod`: Used to change the access modes (access permissions) of a file.

Answer 5 – Race condition

```c
#include <stdio.h>
#include <sys/types.h>

void main (void){
  FILE* f1;
  int i, x = 2000;

  for(i = 0; i < 200; i++){
    // reads current balance from text file
    f1 = fopen("balance.txt", "r");
    fscanf(f1, "Account Balance: %d", &x);
    fclose(f1);
    x++;
    // writes updated balance to text file
    f1 = fopen("balance.txt", "w+");
    fprintf(f1, "Account Balance: %d", x);
    fclose(f1);
  }
}
```

balance.txt

```
Account Balance: 2000
```

>> Executing two separate times

```
● goofynugtz@archangel:~/Downloads/OS Codes/Assignment 1$ ./a.out
● goofynugtz@archangel:~/Downloads/OS Codes/Assignment 1$ ./a.out
○ goofynugtz@archangel:~/Downloads/OS Codes/Assignment 1$
```

>> Output

balance.txt

```
Account Balance: 2400
```

>> Executing twice parallel

```
● goofynugtz@archangel:~/Downloads/OS Codes/Assignment 1$ ./a.out & ./a.out &
  [1] 5530
  [2] 5531
```

>> Output

balance.txt

```
Account Balance: 2650
```