📘 **C Programming Basics**

Lab Session: System Calls in C Programming

Course: Operating Systems
Date: 23.05.2025 practical 02

Practical 1: Multi-process Tasks (Factorial, Fibonacci, Prime)

```
#include <stdio.h>

#include <unistd.h>


int X, Y, Z;


// factorial, fibonacci, is_prime functions


int main(){
   printf("Enter the number (X): ");
   scanf("%d", &X);
   printf("Enter the number (Y): ");
   scanf("%d", &Y);
   printf("Enter the number (Z): ");
   scanf("%d", &Z);


   int f1, f2;
   int p = getpid();
   printf("Process A (Parent) PID: %d\n", p);


   f1 = fork();
```

```
    if (f1 == 0) {
        printf("Process B (Child 1) PID: %d, Parent PID: %d\n", getpid(), getppid());
        // factorial logic
    } else {
        f2 = fork();
        if (f2 == 0) {
            printf("Process C (Child 2) PID: %d, Parent PID: %d\n", getpid(), getppid());
            // fibonacci and prime check logic
        }
    }
    return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ gcc process_calc.c -o process_calc
[2021ict108@fedora ~]$ ./process_calc
Enter the number (X): 5
Enter the number (Y): 7
Enter the number (Z): 11
Process A (Parent) PID: 3000
Process B (Child 1) PID: 3001, Parent PID: 3000
Factorial of 5 is 120
Process C (Child 2) PID: 3002, Parent PID: 3000
Fibonacci series: 0 1 1 2 3 5 8
11 is a prime number.
```

Practical 2: Fork Tree with wait()

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>

int main() {
    int n;
    printf("Enter number: ");
    scanf("%d", &n);

    int child1 = fork();

    if (child1 == 0) {
        // factorial
        int child5 = fork();
        if (child5 == 0) {
            // fibonacci
        } else {
            wait(NULL);
        }
    } else {
        int child2 = fork();
        if (child2 == 0) {
            // power of 2
        } else {
            int child3 = fork();
```

```
        if (child3 == 0) {
            // square
        } else {
            wait(NULL);
            wait(NULL);
        }
    }
  }

  return 0;
}
```

Fedora Output:

```
[2021ict108@fedora ~]$ gcc fork_tree.c -o fork_tree
[2021ict108@fedora ~]$ ./fork_tree
Enter number: 4
Child1 -> PID: 3010     PPID: 3009
Factorial of 4 is 24
Child5 -> PID: 3012     PPID: 3010
Fibonacci term at position 4 is 3
Child2 -> PID: 3011     PPID: 3009
2 to the power of 4 is 16
Child3 -> PID: 3013     PPID: 3009
Square of 4 is 16
```