CL205 - Operating Systems Lab

Task#05

1. A Fibonacci series is a set of numbers where every nth number is the sum of the nth-1 and nth-2 numbers. The only exception to this rule is the 1st and 2nd numbers which are 0 and 1 respectively. The following code can find the nth number in the Fibonacci sequence:

```
int fib(int n)
{
    if (n<=0)
        return 0;
    else if (n==1)
        return 1;
    else
        return fib(n-1)+fib(n-2);
}
int main()
{
    int find = 40;
    printf("Element No. %d in series is: %d", find,
fib(find));
    exit(0);
}
```

Note that the call to fib() function is recursive. Modify the above code so that each fib() call is implemented in a separate thread.

2. Given two matrices, A and B, where matrix A contains M rows and K columns and matrix B contains K rows and N columns, the **matrix product** of A and B is matrix C, where C contains M rows and N columns. The entry in matrix C for row i, column j ($C_{i,j}$) is the sum of the products of the elements for row i in matrix A and column j in matrix B. That is,

$$C_{i,j} = \sum_{n=1}^{k} A_{i,n} \times B_{n,j}$$

For example, if A is a 3-by-2 matrix and B is a 2-by-3 matrix, element $C_{3,1}$ is the sum of $A_{3,1}$ x $B_{1,1}$ and $A_{3,2}$ x $B_{2,1}$. Calculate each element $C_{i,j}$ in a separate thread. This will involve creating M x N threads. The main thread will initialize the matrices A and B and allocate sufficient memory for matrix C, which will hold the product of matrices A and B. These matrices will be declared as global data so that each thread has access to A, B, and C. Matrices A and B can be initialized statically, as shown below:

```
#define M 3
#define K 2
#define N 3

int A[M][K] = {{1, 4},{2, 5},{3, 6}};
int A[K][N] = {{8, 7, 6},{5, 4, 3}};
int C[M][N];
```