4. The Fibonacci sequence is the series of numbers 0, 1, 1, 2, 3, 5, 8, .... Formally, it can be expressed as: f ib0 = 0 f ib1 = 1 f ibn = f ibn−1 + f ibn−2 Write a multithreaded program that generates the Fibonacci sequence. This program should work as follows: On the command line, the user will enter the number of Fibonacci numbers that the program is to generate. The program will then create a separate thread that will generate the Fibonacci numbers, placing the sequence in data that can be shared by the threads (an array is probably the most convenient data structure). When the thread finishes execution, the parent thread will output the sequence generated by the child thread. Because the parent thread cannot begin outputting the Fibonacci sequence until the child thread finishes, the parent thread will have to wait for the child thread to finish.

**Code: -**

#include<iostream>//this header file include(library) standard input output functionalities

#include<thread>//this header file include (library) all methods related to threads,creating and joining etc.

#include<chrono>//this header file include (library) functionalities related to date and time

using namespace std;//standard input output

using namespace std::chrono;//standard date and time

void fib(int n)

//threaded process to calculate and store fibonacci in array

{

int \*arr=new int [n];

arr[0]=0;//initialising first term

arr[1]=1;//initialising second term

if(n==1){

//if input number of terms in one

cout<<"element"<<arr[0]<<" is generated and stored in array"<<"\n";

}

else if(n==2){

//if input number of terms is 2

cout<<"element "<<arr[0]<<" is generated and stored in array"<<"\n";

cout<<"element "<<arr[1]<<" is generated and stored in array"<<"\n";

}

else

{ // if input number of terms is greater than 2

cout<<"element "<<arr[0]<<" is generated and stored in array"<<"\n";

cout<<"element "<<arr[1]<<" is generated and stored in array"<<"\n";

for(int i=2;i<n;i++)//loop to calculate other fibonacci term greater than second term

{

int element=arr[i-1]+arr[i-2];//storing each element in a temporary variable

arr[i]=arr[i-1]+arr[i-2];//storing that element in array

//showing generation of number in sequence

cout<<"element "<<element<<" is generated and stored in array"<<"\n";

}

cout<<"============================================================================="<<endl;

//printing the fibonacci equence

cout<<"The resultant fibonacci series is"<<endl;

for(int i=0;i<n;i++)//loop to print whole array

{

cout<<arr[i]<<" ";// printing each element

}

cout<<endl;

cout<<"============================================================================="<<endl;

}

}

int main()// main thread

{

int n;//variable to store number of terms to be printed

cout<<"enter a number to print fibonacci"<<"\n";

cout<<"============================================================================="<<endl;

cin>>n;

cout<<"============================================================================="<<endl;

cout<<"============================================================================="<<"\n";

//storing the current or starting thread creation time child thread

auto starttime = high\_resolution\_clock::now();

std::thread t1(fib,n);//creating child thread using thread keyword with name t1 and passing function as argument

t1.join();//after executing thread it will join child thread to main thread

//until and unless the thread will not be joined int the main process the execution of child thread cant produce output

auto stoptime=high\_resolution\_clock::now();//storing the terminating time of child thread

auto duration=duration\_cast<microseconds>(stoptime-starttime);//finding total time of execution

cout<<endl;

//printing total time of execution.

cout<<"============================================================================="<<endl;

cout<< "Time taken in execution in sec: "<<duration.count()/1000000<<endl;

cout<<"============================================================================="<<endl;

return 0;

}