Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

\_\_\_\_\_\_\_10\_\_\_\_\_\_\_

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 1 | **Create two thread that must be affected by one thread as event. Initial state of Event must be un-signaled.**  **First thread open notepad.exe**  **Second thread open calculator.exe**  **output must be display current Event first and with running thread.** |

Submitted On:

28-12-2022

(Date: DD/MM/YY)

**Task # 01: Create two threads that must be affected by one thread as event. Initial state of Event must be un-signaled.**

**First thread open notepad.exe**

**Second thread open calculator.exe**

**Output must be display current Event first and with running thread.**

**Solution:**

#include<iostream>

#include<Windows.h>

using namespace std;

HANDLE hEvent;

DWORD WINAPI MyThread\_Fun1(LPVOID lpparam){

    STARTUPINFOW si;

    PROCESS\_INFORMATION pi;

    ZeroMemory (&si,sizeof(si));

    ZeroMemory(&pi,sizeof(pi));

    BOOL bCreateProcess=CreateProcessW(L"C:\\Windows\\System32\\notepad.exe",NULL,NULL,NULL,FALSE,0,NULL,NULL,&si,&pi);

    if(bCreateProcess==false){

        cout<<"Failed"<<endl;

    }

    cout<<"Notepad open successfully"<<endl;

    return 0;

}

DWORD WINAPI MyThread\_Fun2(LPVOID lpparam){

    STARTUPINFOW si;

    PROCESS\_INFORMATION pi;

    ZeroMemory (&si,sizeof(si));

    ZeroMemory(&pi,sizeof(pi));

    BOOL bCreateProcess=CreateProcessW(L"C:\\Windows\\System32\\calc.exe",NULL,NULL,NULL,FALSE,0,NULL,NULL,&si,&pi);

    if(bCreateProcess==false){

        cout<<"Failed"<<endl;

    }

    cout<<"Calculator open successfully"<<endl;

    return 0;

}

int main(){

    HANDLE hThread1,hThread2;

    DWORD ThreadID1,ThreadID2;

    hEvent=CreateEventW(NULL,FALSE,FALSE,L"My\_EVENT");

    if(hEvent==NULL){

        cout<<"Event not created successfully"<<endl;

    }

    cout<<"Event created successfully"<<endl;

    hThread1=CreateThread(NULL,0,MyThread\_Fun1,NULL,0,&ThreadID1);

    if(hThread1==NULL){

        cout<<"Thread 1 not created"<<endl;

    }

    hThread2=CreateThread(NULL,0,MyThread\_Fun2,NULL,0,&ThreadID2);

    if(hThread2==NULL){

        cout<<"Thread 2 not created"<<endl;

    }

    WaitForSingleObject(hThread1,INFINITE);

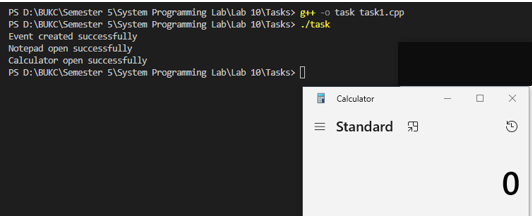
    WaitForSingleObject(hThread2,INFINITE);

    CloseHandle(hThread1);

    CloseHandle(hThread2);

    CloseHandle(hEvent);

}

**Output:**