Лабораторна робота №4

Тема: Синхронізація потоків за допомогою об’єктів Mutex і Semaphores.

Мета роботи: навчитися синхронізувати потоки за допомогою об’єктів синхронізації мютексів (Mutex) і семафорів (Semaphores).

Порядок виконання.

Для прикладу розглянемо дві програми, котрі працюють з одним файлом. Перша програма записує дані у файл, а друга зчитує. Задача полягає в тому, щоб уникнути конфліктів в даній ситуації, тобто зробити так щоб друга програма немала доступу до програми поки перша не закінчить свою роботу.

Створимо два проекти *ConsoleAplication***.**

Код програми для запису даних у файл:

```
#pragma hdrstop
#include "windows.h"
#include "fstream.h"
#include "iostream.h"

//---------------------------------------------------------------------

#pragma argsused
int main()
{
        HANDLE hShared = CreateMutex(NULL, TRUE, "WriteData");
        ofstream ofs("d:\\write.txt");
        cout << "Writing to file: " << endl;
        for (int i = 256; i <= 65535; ++i)
        {
          cout<< i << endl;
          ofs << i << ' ';
        }
        int i;
        cout << "Press Key and Enter for access to file " << endl;
        cin >> i;
        ofs.close();
        ReleaseMutex(hShared);
        CloseHandle(hShared);
        return 0;
}
//---------------------------------------------------------------------
```

Дана програма виводить у файл числа від 256 до 65535

Для повільнішого виводу даних можна використати функцію **Sleep()**

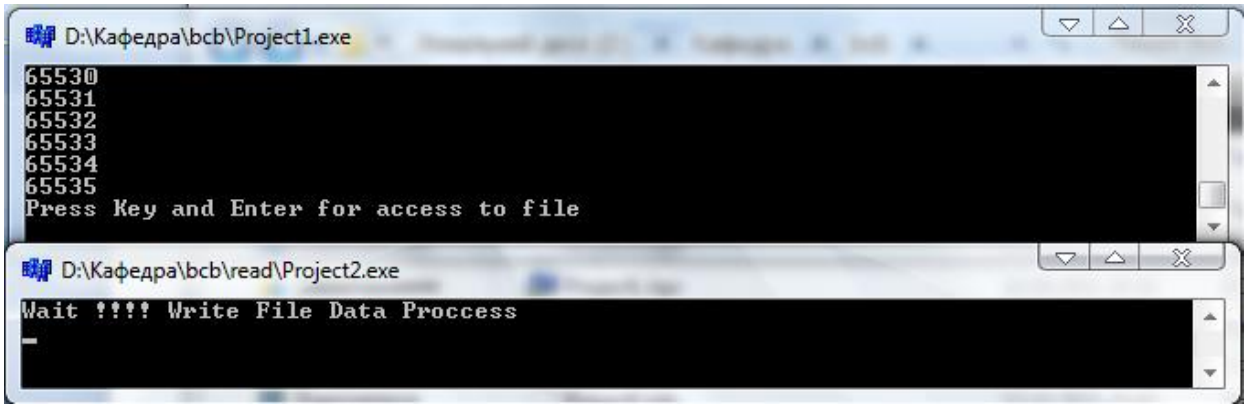
Код програми для зчитування даних з файлу:

```
#include "windows.h"
#include "fstream.h"
#include "iostream.h"

void main()
{
        HANDLE hShared = OpenMutex(MUTEX_ALL_ACCESS, FALSE, "WriteData");
        cout << "Wait !!!! Write File Data Proccess " << endl;
        WaitForSingleObject(hShared, INFINITE);
        ifstream ifs("d:\\write.txt");
        char buffer[100];
        ifs >> buffer;
        cout << "Read File Data  -  " << buffer << endl;
        int i = 0;
        cin>>i;
        CloseHandle(hShared);
```

В даному коді програми для доступу до **Mutex** його потрібно спочатку відкрити з тим же ім'ям, що і в першому коді, і задати **MUTEX_ALL_ACCESS**. Функція **WaitForSingleObject** буде чекати поки доступ не буде відкритий.

Запустіть одночасно дві програми. Програма зчитування буде чекати поки програма запису не дозволить доступу. Так як показано на скріншоті:



Введіть букву в програму запису і **Enter** , і програма зчитування розпочне читати дані. Приблизно так виглядає синхронізація. Даний механізм можна використовувати не тільки для запису/читання файлів, але і для будь-якої синхронізації між потоками.

Індивідуальні завдання

Задано **хп=-1**, **хк=1**, **Δx=0.2**, **y=4.7**, **z=1.32**. Відповідно до номера варіанту потрібно розробити програму для одновимірного табулювання функцій **a=f[x,y,z,b]** і **b=f[x,y,z]** за незалежною змінною **x** згідно з математичними виразами, результати табулювання вивести у файл:

1) $$a[x,y,z,b]=\frac{\sqrt{\left|x^2-z\right|^{0.3}}-\sqrt[3]{y+2b}}{1+\frac{x^1}{1!}+\frac{y^2}{2!}+\frac{z^3}{3!}}.\quad b[x,y,z]=x\left(\frac{y+\operatorname{arctg}\left|x^2+z\right|^{0.1}}{3+\sin^2 y^3}+e^{-\frac{x+z}{y}}\right):$$

2) $$a[x,y,z,b]=\left|x\right|^{0.43}+\frac{e^{y-x}+\sqrt{\left|y^2+b\right|^{0.22}}}{1+x^2\left|y-tg^2 z\right|}.\quad b[x,y,z]=\frac{2z+\cos\left|y-3x\right|^{1/3}}{2.1+\sin^2\left|z^3-y\right|^{0.2}}+\ln^2\left|z-x\right|:$$

3) $$a[x,y,z,b]=(x+y)^2+\frac{x+z^3/\left(b^2+y\right)^2}{1+e^{-(x-y)}+\left|z\right|^{0.34}}.\quad b[x,y,z]=1+\frac{\left|y-x\right|^2}{\left|z\right|^{1.34}}+\frac{(z-x)^2}{\sin^2 y}+\frac{\left|y-z\right|^3}{\cos y^2}:$$

4) $$a[x,y,z,b]=y^2+\frac{x^2+\sin^2 b}{y^2+\left|\frac{x^2}{y+x^3/3}\right|-\ln\left|x\right|}.\quad b[x,y,z]=\left|\frac{1}{\left|z\right|^{0.6}}+\sin^2\frac{x+z^2}{2x-y}\right|^{1/3}-ze^{\frac{x^2-y}{z}}:$$

5) $$a[x,y,z,b]=\frac{2\cos\left|x\right|^{1/3}-x^2/6}{1-b+\sin^2 y^3}+\ln^2\left|z\right|^{0.6}.\quad b[x,y,z]=\frac{x^2+z^2/tg^2\left|x\right|^{0.3}}{3+x+y^2/2!+z^3/3!}+\ln^{0.3}\left|\frac{z}{x}\right|^{\frac{1}{3}}:$$

6) $$a[x,y,z,b]=\frac{1+\sqrt{\sin^2\left|x+y\right|^{0.4}}}{2+b^2+\sin^2 y^3}+tg\frac{3x}{y}.\quad b[x,y,z]=\cos^2\left(\operatorname{arctg}\frac{x^2+y}{z+1}\right)+\frac{x}{y}e^{3x+y}:$$

7) $$a[x,y,z,b]=\ln\left|\left(y-\sqrt{\left|x^2-b\right|}\right)\frac{y-x^2}{z+4y^2}\right|^{\frac{2}{3}}.\quad b[x,y,z]=1-\frac{x+y}{\left|z\right|^{0.34}}+\frac{y^2}{3!}+\frac{z^3}{5!}+\frac{e^{x-y}}{z+y}:$$

8) $$a[x,y,z,b]=\frac{\sqrt{\left|x^2-1\right|^{0.3}}-\sqrt[3]{\left|y+2b\right|}}{1+\frac{x^1}{1!}+\frac{y^2}{2!}+\frac{z^3}{3!}}.\quad b[x,y,z]=\frac{\left|y+x\right|^{0.2}}{\left|z\right|^{1.34}}+\frac{(y-z)^2}{1+\sin^2 y}+\frac{\left|z-y\right|^3}{1-\cos x^2}:$$

9) $$a[x,y,z,b]=\frac{3+e^{y-x}+\sqrt{\left|y^2+b\right|^{0.3}}}{1+x^2\left|y-tg^2 z^2\right|}.\quad b[x,y,z]=y\left|\frac{\left|z\right|^{0.3}}{x}+tg^2\frac{x+z^2}{2x-1.4}\right|^{\frac{1}{3}}-ze^{x^2-y}:$$

) $$a[x,y,z,b]=(1+y)^2\frac{\left|x+y\right|^{0.3}/b^2+z}{1+e^{-(x-z)}+\left|y\right|^{0.43}}.\quad b[x,y,z]=x\left(\frac{y+\operatorname{arctg}\left|x^2+z\right|^{0.1}}{2+\sin^2 y^3}+e^{-\frac{x+z}{y}}\right):$$

1
к". Для

Приклад виконання завдання з використанням об'єкту синхронізації потоків Semaphore

```
#include <windows.h>
#include <iostream.h>
#include <conio.h>

#include <vcl.h>
#include <math.h>
volatile int a[10];
HANDLE hSemaphore;

DWORD WINAPI thread(LPVOID)
{
  for (int i = 0; i < 10; i++)
  {
    a[i] = i + 1;
    ReleaseSemaphore(hSemaphore,1,NULL);
    Sleep(500);
  }

  return 0;
}

int main()
{
  int i;
  double x=0,b=0;
  cout<<"Vvedit x = ";
  cin>>x;
  cout<<"Vvedit b = ";
  cin>>b;
  HANDLE  hThread;
  DWORD  IDThread;

  cout << "An initial state of the array: ";
  for (i = 0; i < 10; i++)
    cout << a[i] <<' ';
  cout << endl;
  // ??????? ???????
  hSemaphore=CreateSemaphore(NULL, 0, 10, NULL);
  if (hSemaphore == NULL)
    return GetLastError();

  hThread = CreateThread(NULL, 0, thread, NULL, 0, &IDThread);
  if (hThread == NULL)
    return GetLastError();

  cout << "A final state of the array: ";
  for (i = 0; i < 10; i++)
  {
    WaitForSingleObject(hSemaphore, INFINITE);
    cout << sin(a[i]+x+b) << " \a" << flush;
  }
  getch();
  cout << endl;

  CloseHandle(hSemaphore);
  CloseHandle(hThread);

  return 0;
}
```