Лабораторна робота №2

Тема: Програмування потоків в ОС MS Windows з використанням прикладного програмного інтерфейсу WinAPI.

Мета роботи: навчитися розробляти багатопотокові аплікації (програми) в ОС Windows з використанням прикладного програмного інтерфейсу WinAPI.

Порядок виконання.

Створюється потік функцією CreateThread, яка має наступний прототип:

```
HANDLE CreateThread (
        LPSECURITY_ATTRIBUTES lpThreadAttributes, / / атрибути захисту
        DWORD dwStackSize, / / розмір стека потоку в байтах
        LPTHREAD_START_ROUTINE lpStartAddress, / / адреса виконуваної функції
        LPVOID lpParameter, / / адреса параметра
        DWORD dwCreationFlags, / / прапори створення потоку
        LPDWORD lpThreadId / / ідентифікатор потоку
) ;
```

При успішному завершенні функція CreateThread повертає дескриптор створеного потоку і його ідентифікатор, який є унікальним для всієї системи. В іншому випадку ця функція повертає значення NULL.

Коротко опишемо призначення параметрів функції CreateThread. Параметр lpThreadAttributes встановлює атрибути захисту створюваного потоку. До тих пір поки ми не вивчимо структуру системи безпеки в Windows, тобто розділ Windows NT Access Control з інтерфейсу програмування додатків Win32 API, ми будемо встановлювати значення цього параметра в NULL при виклику майже всіх функцій ядра Windows. Це означає, що атрибути захисту потоку збігаються з атрибутами захисту створив його процесу. Про процеси буде детально розказано в наступному розділі.

Параметр dwStackSize визначає розмір стека, який виділяється потоку при запуску. Якщо цей параметр дорівнює нулю, то потоку виділяється стек, розмір якого дорівнює

за замовчуванням 1 Мб. Це найменший розмір стека, який може бути виділений потоку. Якщо величина параметра dwStackSize менше, значення, заданого за замовчуванням, то все одно потоку виділяється стек розмірів у 1Мб. Операційна система Windows округлює розмір стека до однієї сторінки пам'яті, що звичайно дорівнює 4 Кб.

Параметр lpStartAddress вказує на виконувану потоком функцію. Ця функція повинна мати наступний прототип:

DWORD WINAPI ThreadProc (LPVOID lpParameters) ;

Параметр lpParameter є єдиним параметром, який буде переданий функції потоку. Параметр dwCreationFlags визначає, в якому стані буде створений потік. Якщо значення цього параметра дорівнює 0, то функція потоку починає виконуватися відразу після створення потоку. Якщо ж значення цього параметра 2 одно CREATE_SUSPENDED, то потік створюється в підвішеному стані. Надалі цей потік можна запустити викликом функції ResumeThread. Параметр lpThreadId є вихідним, тобто його значення встановлює Windows. Цей параметр повинен вказувати на змінну, в яку Windows помістить ідентифікатор потоку, який унікальний для всієї системи і може надалі використовуватися для посилань на потік.

Наведемо приклад програми, яка використовує функцію CreateThread для створення потоку, і продемонструємо спосіб передачі параметрів виконуваної потоком функції.

/ / Приклад створення потоку функцією CreateThread

```
# include <windows.h>
# include <iostream.h>
volatile int n ;
 DWORD WINAPI Add ( LPVOID iNum )
{
        cout << " Thread is started. " << endl ;
        n + = ( int ) iNum ;
        cout << " Thread is finished. " << endl ;
        return 0 ;
}
 int main ( )
```

```
{
        int inc = 10;
       HANDLE hThread ;
       DWORD IDThread ;
        cout << " n = " << n << endl ;
        hThread = CreateThread(NULL,0,Add,(void*)inc,0,&IDThread);
       if ( hThread == NULL)
       return GetLastError ();

       // Чекаємо поки потік Add закінчить роботу
       WaitForSingleObject ( hThread, INFINITE ) ;
       // Закриваємо дескриптор потоку Add
       CloseHandle ( hThread ) ;
        cout << " n = " << n << endl ;
       return 0 ;
}
```

Зазначимо, що в цій програмі використовується функція WaitForSingleObject, яка чекає завершення потоку Add.

Завдання

1. Вивчити програму для консольного процесу, який складається з двох потоків: main і worker.Потік main повинен виконати наступні дії :

1. Створити масив цілих чисел, розмірність і елементи якого вводяться з консолі.
2. Створити потік worker.
3. Знайти мінімальний і максимальний елементи масиву і вивести їх на консоль. Після кожного порівняння елементів «спати» 7 мілісекунд.
4. Дочекатися завершення потоку worker.
5. Підрахувати кількість елементів у масиві, значення яких більше середнього значення елементів масиву, і вивести його на консоль.
6. Завершити роботу.

Потік worker повинен виконати наступні дії :

1. Знайти середнє значення елементів масиву. після кожного підсумовування елементів «спати» 12 мілісекунд.
2. Завершити свою роботу.

Примітки.

1. Для очікування завершення роботи потоку worker використовувати функцію:

```
DWORD WaitForSingleObject (
        HANDLE hHandle, / / дескриптор об'єкта
        DWORD dwMilliseconds / / інтервал очікування в  Мілісекундах
) ;
```

де другий параметр встановити рівним INFINITE. наприклад

WaitForSingleObject (hThread, INFINITE) ; / / чекати завершення потоку

тут **hThread** - дескриптор потоку worker.

2. Для «засипання» використовувати функцію:

```
VOID Sleep (
        DWORD dwMilliseconds / / мілісекунди
) ;
```

наприклад,

Sleep (12); / / спати 12 мілісекунд

2. Модифіковані та налагодити програму відповідно до свого варіантом.

Варіанти

1). Потік worker повинен знайти значення факторіала елементів масиву.

2). Потік worker повинен знайти значення суми парних елементів масиву.

3). Потік worker повинен знайти значення кількість парних елементів масиву.

4). Потік worker повинен знайти значення кількість непарних елементів масиву.

5). Потік worker повинен знайти значення суми непарних елементів масиву.

6). Потік worker повинен знайти значення середнє значення парних елементів масиву.

7). Потік worker повинен знайти значення середнє значення непарних елементів масиву.

8). Потік worker повинен знайти значення факторіала парних елементів масиву.

9). Потік worker повинен знайти значення факторіала непарних елементів масиву.

10). Потік worker повинен знайти значення середнє значення елементів масиву, виключаючи максимальний елемент.

11). Потік worker повинен знайти значення середнє значення елементів масиву, виключаючи мінімальний елемент.

12). Потік worker повинен знайти значення факторіала елементів масиву, виключаючи

максимальний елемент.

13). Потік worker повинен знайти значення факторіала елементів масиву, виключаючи

мінімальний елемент.

14). Потік worker повинен знайти значення суми непарних елементів масиву і мінімального елемента.

15). Потік worker повинен знайти значення суми парних елементів масиву і мінімального елемента.

16). Потік worker повинен знайти значення факторіала елементів масиву.

17). Потік worker повинен знайти значення суми парних елементів масиву.

18). Потік worker повинен знайти значення кількість парних елементів масиву.

19). Потік worker повинен знайти значення кількість непарних елементів масиву.

20). Потік worker повинен знайти значення суми непарних елементів масиву.

21). Потік worker повинен знайти значення середнє значення парних елементів масиву.

22). Потік worker повинен знайти значення середнє значення непарних елементів масиву.

23). Потік worker повинен знайти значення факторіала парних елементів масиву.

24). Потік worker повинен знайти значення факторіала непарних елементів масиву.

25). Потік worker повинен знайти значення середнє значення елементів масиву, виключаючи максимальний елемент.

26). Потік worker повинен знайти значення середнє значення елементів масиву, виключаючи мінімальний елемент.

27). Потік worker повинен знайти значення факторіала елементів масиву, виключаючи

максимальний елемент.

28). Потік worker повинен знайти значення факторіала елементів масиву, виключаючи

мінімальний елемент.

29). Потік worker повинен знайти значення суми непарних елементів масиву і мінімального елемента.

30). Потік worker повинен знайти значення суми парних елементів масиву і мінімального елемента.

Приклад:

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

volatile int n ;
int O[200];
int p11=200;
int p=0;
int ch1=0;

DWORD WINAPI worker ( LPVOID iNum )
{
    int *A=new int[p11];
    for(int i=0;i<p11;i++)
    {
         A[i]=O[i];
     }
    int ch=ch1;
    float  fact=1;
    for(int i=1;i<ch;i++)
   {
      fact*=A[i];
      Sleep(12);
    }
    float ser;
    float sum;
```

```
  for(int i=0;i<ch;i++)
  {
       sum+=A[i];
   }
   ser=sum/ch;
   int ch2=0;
   for(int i=0;i<ch;i++)
 {

   if (A[i]>ser)
   {
    ch2++;
   }
 }

 cout<<"factorial \n";
 cout<<fact<<"\n";
 cout<<"serednye znachennya elementiv\t";
 cout<<ser<<"\n";
 cout<<"kilkist elementiv masivy > za seredne znachennja \t";
 cout<<ch2<<"\n";
 int h=0;
  for(int i=0;i<ch;i++)
 { if(A[i]%2!=0)
       {
         p++;
       }
 }

 cout<<"kilkist neparnuch elementiv";
 cout<<p;
 cout<<"\n";

// Sleep(8000);

 return 0 ;
}
 DWORD WINAPI main1 ( LPVOID iNum )
 {int *A=new int [ch1];
 int ch;
 ch=ch1;
  for(int i=0;i<ch1;i++)
     {
       A[i]=O[i];
```

```
      }
 int max=A[0],min=A[0];
  for(int i=0;i<ch;i++)
  {if (max<A[i])
  {max=A[i];
  Sleep(7);
  }
  }
  for(int i=0;i<ch;i++)
  {if (min>A[i])
  { min=A[i];
  Sleep(7);

  }
  }
  cout<<"max element masivy \t" ;
  cout<<max; cout<<"\n";
  cout<<"min element masivy \t";
  cout<<min;
  cout<<"\n";

  for(int i=0;i<ch;i++)
  {
   O[i]=A[i];
  }
  Sleep(1000);
  return 0;
  }

int main ( )
{
    HANDLE hThread ;
    DWORD IDThread ;
    int *A;
     int ch;
     cout<<"vvedit rozmir masuvy";
     cin>>ch;
     A=new int [ch];
     for(int i=0;i<ch;i++)
    {
        cout<<"vvedit elementu masuvy \n";
        cin>>A[i];
    }
    for(int i=0;i<ch;i++)
```

```
  {
      O[i]=A[i];
  }
  ch1=ch;
  p11=ch;
  hThread = CreateThread(NULL,0,main1,(void*)1,0,&IDThread);
  hThread = CreateThread(NULL,0,worker,(void*)1,0,&IDThread);
  // Закриваємо дескриптор потоку worked
  CloseHandle ( hThread ) ;
  WaitForSingleObject ( hThread, INFINITE ) ;
  // Закриваємо дескриптор потоку worked
 //CloseHandle ( hThread) ;
  system("pause");
  return 0 ;
}
```