Лабораторна робота №1
“Об’єкти для синхронізації процесів у ОС Windows”

Мета роботи

Розробити прикладну програму із застосуванням типових об’єктів, що застосовуються для синхронізації у ОС Windows (можна використати аналоги для ОС Linux/UNIX).

Завдання

Спроектувати архітектуру розподіленої багатопроцесної прикладної програми із застосуванням специфіки ОС Windows та WinAPI (для ОС Linux/UNIX згідно стандарту POSIX знайти та використати аналоги). У розробці логіки прикладної програми використати типові об’єкти синхронізації:

- ☐ м’ютекс (mutex)
- ☐ семафор (semaphore)
- ☐ таймер (timer / waitable timer)
- ☐ подія (event)
- ☐ критична секція (critical section)

У ОС Windows із застосуванням WinAPI обов’язково використати: для організації очікування переходу об’єктів у сигнальний стан використовувати функції WaitForSingleObject() та/або WaitForMultipleObjects(); для ідентифікації стану об’єктів та роботи з ними використати функцію GetLastError(); для ілюстрації синхронізації процесів створити кілька процесів з допомогою функції CreateProcess() та застосувати передачу між процесами об’єктів синхронізації по спадковості (тобто використати анонімні, неіменовані об’єкти і застосувати передачу дескриптора об’єкта через командний рядок).
Примітка. Замість Linux можна використати середовище WSL2 під Windows для розробки.

Варіанти завдань

Для усіх варіантів однаково: реалізувати процес-предок, що передасть по спадковості процесу-нащадку неіменований м’ютекс; реалізувати головний процес, що може запускатися лише один раз водночас (тобто копії процесу повинні припиняти свою роботу, якщо вже є одна запущена версія цього процесу – можна використати іменований м’ютекс); реалізувати головний процес, що ініціює створення та запускає 10 процесів, кожен з яких виводить на екран призначений йому номер, але ці процеси мають бути синхронізовані з допомогою семафора так, щоб одночасно працювало не більше трьох процесів; також головний процес, запустивши згадані 10 інших процесів, із допомогою таймера зачекавши 5 секунд, має перевірити стан виконання усіх створених ним процесів та вивести на екран повідомлення про завершення їх роботи (тобто перевірити, що об’єкти відповідних процесів вже перейшли в сигнальний стан).

Лабораторна робота №2
“Об’єкти для синхронізації потоків у ОС Windows”

Мета роботи

Розробити прикладну програму із застосуванням типових об’єктів, що застосовуються для синхронізації у ОС Windows (можна використати аналоги для ОС Linux/UNIX).

Завдання

Спроектувати архітектуру розподіленої багатопотокової прикладної програми із застосуванням специфіки ОС Windows та WinAPI (для ОС Linux/UNIX згідно стандарту POSIX знайти та використати аналоги). У розробці логіки прикладної програми використати типові об’єкти синхронізації:

- ☐ подія (event)
- ☐ критична секція (critical section)

У ОС Windows із застосуванням WinAPI обов’язково використати: для організації очікування переходу об’єктів у сигнальний стан використовувати функції WaitForSingleObject() та/або WaitForMultipleObjects(); для ідентифікації стану об’єктів та роботи з ними використати функцію GetLastError(); для ілюстрації синхронізації потоків використати функцію CreateThread() та призначити кожному потоку окрему кучу пам’яті створену з допомогою функції CreateHeap(); для ілюстрації одночасної роботи зі спільними даними використати об’єкт проекції файлу у пам’ять, створений з допомогою функції CreateFileMapping().
Примітка. Замість Linux можна використати середовище WSL2 під Windows для розробки.

Варіанти завдань

Для усіх варіантів однаково: реалізувати головний потік, що створить три пари потоків у призупиненому стані та з різними пріоритетами виконання потоків у парі; у кожній парі потоків один потік в парі має виводити на екран та записувати у проекцію файлу додатні числа (від 1 до 500), а другий – від’ємні (від -1 до -500); логіка пар повинна передбачати різний підхід до синхронізації: 1) без синхронізації, 2) синхронізація з допомогою об’єкта подія (з ручним або автоматичним оновленням), 3) синхронізація з допомогою об’єкта критична секція (без або зі спін-блокуванням); головний потік має відновити роботу створених ним у призупиненому стані потоків та синхронізувати їх роботу через семафор, щоб дозволяти одночасно працювати не більше ніж двом потокам і саме таким чином обмежити виконання у відповідних парах.

Лабораторна робота №3
“Клієнт-серверна прикладна програма”

Мета роботи

Розробити прикладну програму із застосуванням типових об’єктів, що застосовуються для мережевої передачі даних у ОС Windows (можна використати аналоги для ОС Linux/UNIX).

Завдання

Спроектувати архітектуру розподіленої багатопотокової клієнт-серверної прикладної програми із застосуванням специфіки ОС Windows та WinAPI (для ОС Linux/UNIX згідно стандарту POSIX знайти та використати аналоги). У розробці логіки прикладної програми використати типові об’єкти передачі даних:

- ☐ іменовані канали (named pipe) і поштові скриньки (mailslot)
- ☐ сокети (socket) і бродкастинг (UDP broadcasting)

Примітка. Замість Linux можна використати середовище WSL2 під Windows для розробки.

Варіанти завдань

1. Реалізувати дві прикладні програми, що функціонують відповідно як багатопотоковий сервер і клієнт для взаємодії із цим сервером. Застосувати технологію **іменованих каналів та поштових скриньок**. Загальна задача клієнт-серверної прикладної програми – спільний обмін текстовими повідомленнями (форум) для клієнтів. Поштові скриньки (mailslot) використати для виявлення сервера у мережі.
2. Реалізувати дві прикладні програми, що функціонують відповідно як багатопотоковий сервер і клієнт для взаємодії із цим сервером. Застосувати технологію **сокетів та бродкастингу**. Загальна задача клієнт-серверної прикладної програми – спільний обмін текстовими повідомленнями (форум) для клієнтів. Бродкастинг (broadcasting) використати для виявлення сервера у мережі.

Примітка. Варіант завдання обирати відповідно до спискового номеру у журналі групи за таким алгоритмом: *до остачі від ділення номеру у списку на кількість варіантів додати одиницю*.

Лабораторна робота №4
“Багатопотокова прикладна програма”

Мета роботи

Розробити прикладну програму із застосуванням принципів паралелізму, використовуючи багатопотоковість.

Завдання

Спроектувати архітектуру багатопотокової прикладної програми із застосуванням специфіки ОС (на вибір):

- ОС Windows – із застосуванням WinAPI
- ОС Linux/UNIX – згідно стандарту POSIX

У розробці архітектури прикладної програми використати моделі створення і функціонування потоків та відношення координації потоків відповідно до варіанту.
Примітка. Замість Linux можна використати середовище WSL2 під Windows для розробки.

Для усіх варіантів завдань використати синхронізацію з допомогою м’ютексів та/або семафорів, а також передавання інформації з допомогою каналів (pipe та/або named pipe).

Для усіх варіантів завдань обрати із наступного переліку таку задачу, що може бути ефективно розв’язана із застосуванням відповідного підходу згідно свого варіанту:

1. Обчислення значення многочлена за схемою Горнера.
2. Обчислення коефіцієнтів многочлену, що є добутком многочленів великого порядку.
3. Розв’язування СЛАР (довільного порядку, більше третього) методом Гауса.
4. Обчислення визначників (довільного порядку, більше третього).

Примітка. Відношення координації та модель не слід розглядати як взаємопов’язані елементи – варто обирати задачу, що відповідає заданій у варіанті моделі, та можна організувати логіку синхронізації лише деяких потоків прикладної програми, що відповідає заданому у варіанті відношенню координації.

Варіанти завдань

Номери варіантів подано у клітинках

Модель / Координація	Старт–Старт	Фініш–Старт	Старт–Фініш	Фініш–Фініш
“Делегування”	1	2	3	4
“Рівноправні вузли”	5	6	7	8
“Конвеєр”	9	10	11	12
“Виробник-споживач”	13	14	15	16

Примітка. Варіант завдання обирати відповідно до спискового номеру у журналі групи за таким алгоритмом: *до остачі від ділення номеру у списку на кількість варіантів додати одиницю*.

Лабораторна робота №5
“Прикладна програма із міжпроцесною взаємодією”

Мета роботи

Розробити прикладну програму із застосуванням принципів паралелізму, використовуючи розбиття логіки роботи між кількома процесами.

Завдання

Спроектувати архітектуру прикладної програми, що розділяє логіку роботи між кількома процесами із застосуванням специфіки ОС (на вибір):

- ОС Windows – із застосуванням WinAPI
- ОС Linux/UNIX – згідно стандарту POSIX

У розробці архітектури прикладної програми сформулювати задачу поділу робіт між процесами з можливістю проілюструвати на прикладі проблеми координації паралельної/розподіленої організації.

Варіанти завдань

1. “Перегони даних” (data race)
2. Нескінченне відтермінування (indefinite postponement)
3. Взаємне блокування (dead lock)
4. Труднощі організації зв’язку

Примітка. Варіант завдання обирати відповідно до спискового номеру у журналі групи за таким алгоритмом: *до остачі від ділення номеру у списку на кількість варіантів додати одиницю*.