Список вариантов заданий по синхронизации

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант | Объект | Область действия | Код данных | Объект | Код данных |
| 1 | K | Процесс | К11 | WaitableTimer | T1 |
| 2 | K | Процесс | К15 | WaitableTimer | T2 |
| 3 | K | Процесс | K16 | WaitableTimer | T3 |
| 4 | M | Процесс | M22 | WaitableTimer | T4 |
| 5 | M | Процесс | M23 | WaitableTimer | T5 |
| 6 | M | Система | M15 | WaitableTimer | T1 |
| 7 | M | Система | M13 | WaitableTimer | T2 |
| 8 | M | Система | M14 | WaitableTimer | T3 |
| 9 | S | Система | S12 | WaitableTimer | T4 |
| 10 | S | Система | S14 | WaitableTimer | T5 |
| 11 | S | Система | S15 | WaitableTimer | T1 |
| 12 | M | Процесс | M24 | WaitableTimer | T4 |
| 13 | M | Система | M16 | WaitableTimer | T3 |
| 14 | M | Система | M12 | WaitableTimer | T4 |
| 15 | M | Система | M11 | WaitableTimer | T5 |
| 2-1 | M | Система | M12; | WaitableTimer | T1 |
| 2-2 | M | Система | M12; | WaitableTimer | T2 |
| 2-3 | S | Процесс | S13; | WaitableTimer | T5 |
| 2-4 | K | Процесс | K15; | WaitableTimer | T5 |
| 2-5 | K | Процесс | K12; | WaitableTimer | T3 |
| 2-6 | M | Процесс | M22; | WaitableTimer | T4 |
| 2-7 | S | Система | S12; | WaitableTimer | T4 |
| 2-8 | K | Процесс | K11; | WaitableTimer | T1 |
| 2-9 | M | Процесс | M23; | WaitableTimer | T4 |
| 2-10 | K | Процесс | K13; | WaitableTimer | T2 |
| 2-11 | S | Система | S15 | WaitableTimer | T1 |
| 2-12 | M | Процесс | M24 | WaitableTimer | T4 |
| 2-13 | M | Система | M16 | WaitableTimer | T3 |
| 2-14 | M | Система | M12 | WaitableTimer | T4 |
| 2-15 | M | Система | M11 | WaitableTimer | T5 |

Лабораторная работа

СИНХРОНИЗАЦИЯ ВЗАИМОДЕЙСТВУЮЩИХ ПОТОКОВ В

WIN32 API

Цель работы: Изучить возможности согласования работы

взаимодействующих потоков с целью исключения гонок и обеспечения

необходимой последовательности их выполнения, а так же используемые для

этой цели системные средства. Приобрести практические навыки

синхронизации работы потоков и создания безопасных, с точки зрения

многопоточности, приложений Windows.

Задание: Изучить представленные в папке работы примеры многопоточных

приложений и реализованные в них возможности синхронизации работы.

Разработать собственные безопасные с точки зрения многопоточности

приложения в соответствии с индивидуальными заданиями.

Выполнение:

Предварительный шаг -ознакомление:

Изучите примеры использования объектов синхронизации в

многопоточных приложениях, представленные в папке лабораторной работы

2-2, реализуйте проекты, сформируйте исполнимые файлы, запустите на

выполнение, ознакомьтесь с исходным тестом и рассмотрите использование

объектов синхронизации.

Получение практических навыков синхронизации потоков:

1. Используйте проект предыдущей лабораторной работы 2-2.

2. Добавьте PopUp меню “Синхронизация” с командами “Включить

согласование”, “Выключить согласование”.

3. Добавьте в приложение код обработки новых команд меню и

средства синхронизации работы в соответствии с индивидуальным заданием

1.

4. Выполните компиляцию, компоновку и отладку программы.

5. Выполните тестирование добавленных функций ( работа с синхронизацией

и без).

Дайте пояснение полученной информации. 6. Создайте и протестируйте приложение с применением для синхронизации

объекта «ожидающий таймер» (WaitableTimer Object) в соответствии с

индивидуальным заданием 2.

Варианты заданий по теме

"Синхронизация потоков в Win API приложении"

(задание 1 с объектами синхронизации и задание 2 с ожидающим

таймером)

Варианты задания 1.

Вариант К1. Синхронизирующий объект – критическая секция.

В приложении, имеющем несколько потоков осуществляющих вывод

"бегущей строки", с помощью критической секции организовать работу

потоков таким образом, чтобы каждый из потоков получал возможность

неразрывно выполнить N шагов вывода символов строки. На каждом шаге

строка продвигается на один символ. Неразрывно означает, что в это время

никакой другой аналогичный поток вывод строки выполнять не может.

Отображать количество шагов вывода.

Код

варианта

Значение N Код

варианта

Значение N

К11 N = длинна строки К14 N = длинна строки +

60

К12 N = длинна строки +

20

К15 N = 2 длинны строки

К13 N = длинна строки +

40

К16 N = 4 длинны строки

Вариант М1. Синхронизирующий объект – мьютекс.

Создать приложение, осуществляющее вывод бегущей строки в отдельных

вторичных потоках (не менее двух). Используя синхронизирующий объект

мьютекс организовать работу потоков таким образом, чтобы при запуске

одного или более экземпляров приложения каждый поток, выводящий

строку, получал исключительное право атомарного (неразрывного)

выполнения операции N кратного вывода строки. Под однократным

выводом строки понимается последовательный со сдвигом от левого края

окна вывод всех символов строки. После этого строка выводится снова от

левого края окна. Атомарность здесь означает, что пока один поток

выполняет указанную операцию вывода, другие аналогичные потоки должны

ждать.

Код варианта Значение N Код варианта Значение N

M11 N = 1 M14 N = 4

M12 N = 2 M15 N = 5

M13 N = 3 M16 N = 6

Вариант М2. Синхронизирующий объект – мьютекс. Создать приложение, осуществляющее вывод «бегущей строки » в

отдельных вторичных потоках (не менее двух). Используя

синхронизирующий объект мьютекс организовать работу потоков таким

образом, чтобы каждый поток, выводящий строку, получал исключительное

право неразрывного N кратного вывода строки и это право не должно

зависеть от действий потоков других процессов. Под однократным выводом

строки понимается последовательный со сдвигом от левого края окна вывод

всех символов строки. После этого строка выводится снова от левого края

окна. Неразрывность здесь означает, что пока один поток приложения

выполняет вывод, другие аналогичные потоки этого приложения должны

ждать. Отображать счетчик количества “пробежек” строки.

Код варианта Значение N Код варианта Значение N

M21 N = 1 M24 N = 4

M22 N = 2 M25 N = 5

M23 N = 3 M26 N = 6

Вариант S1. Синхронизирующий объект – семафор.

Имеется приложение, осуществляющее вывод «бегущей строки » в

отдельных вторичных потоках (не менее двух). Организовать работу потоков

таким образом, что бы не зависимо от количества запущенных аналогичных

приложений по экрану одновременно бежало не более N строк.

Код варианта Значение N Код варианта Значение N

S11 N = 1 S14 N = 4

S12 N = 2 S15 N = 4

S13 N = 3 S16 N = 6

==================================================================== Варианты заданий с ожидающим таймером (задание 2)

Т1. Ждущий таймер.

Создать консольное приложение, которое через n = 2 минуты после запуска и

затем каждые K = 11 секунд будет выводить в консольное окно фразу:

«Сигнал № <X>», где X – порядковый номер сигнала.

Т2. Ждущий таймер.

Создать консольное приложение , которое через N =7секунд после запуска и

затем через каждые k=3 секунд будет выводить на консоль фразу «С момента

запуска прошло <X> секунд», где Х=n + i \* k (i =0,1,2, … ,20), где i – номер

срабатывания таймера.

Т3. Ждущий таймер.

Создать консольное приложение, которое через каждые n =15 секунд будет

выводить диалоговое окно MessageBox с текстом «Вызов номер <i>

Продолжить – RETRY. Завершить работу - ABORT». После нажатия кнопки

RETRY или ABORT в MessageBox приложение будет соответствующим

образом реагировать. Если ответ задерживается более чем на 15 секунд, то

это воспринимается как ответ “ RETRY ”

Т4. Ждущий таймер.

Создать консольное приложение, запускающее через 15 секунд после старта

программу Notepad.exe, в окне которой отображается номер срабатывания

таймера: «Запуск № i», i=1,2,…10. Через 10 секунд после запуска программа

Notepad закрывается.

Т5. Ждущий таймер.

Консольное приложение типа «Будильник» с функциями «Соня». Время

срабатывания и период напоминания вводятся в командной строке при

запуске программы.