Отчет по ИДЗ 4

Подготовил Балабанов Максим Михайлович БПИ212

Вариант 4

4. ***Задача о читателях и писателях.*** Базу данных разделяют два типа процессов – читатели и писатели. Читатели выполняют транзакции, которые просматривают записи базы данных, транзакции писателей и просматривают и изменяют записи. Предполагается, что в начале БД находится в *непротиворечивом* состоянии (например, если каждый элемент — число, то они все отсортированы). Каждая отдельная транзакция переводит БД из одного *непротиворечивого* состояния в *другое*. Для предотвращения взаимного влияния транзакций процесс-писатель должен иметь исключительный доступ к БД. Если к БД не обращается ни один из процессов-писателей, то выполнять транзакции могут одновременно сколько угодно читателей. Создать многопоточное приложение с потоками-писателями и потоками-читателями. Реализовать решение, используя семафоры.

4 балла:

- Модель параллельных вычислений, используемая при разработке многопоточной программы: **Производители и потребители**. Это парадигма взаимодействующих неравноправных потоков. Одни потоки «производят» данные, другие их «потребляют». Часто такие потоки организуются в конвейер, через который проходит информация. Каждый поток конвейера потребляет выход своего предшественника и производит входные данные для своего последователя.
- Реализовано консольное приложение, решающее поставленную задачу с использованием одного варианта синхропримитивов.
- Ввод данных в приложение реализован с консоли.
- Результаты работы:

  Формат ввода:
  На вход подается длина начального массива n и сам массив a1 a2 … an
  Затем k – количество писателей и читателей и сами профессии (R – писатель, V - читатель)

```
4 1 2 3 4
3 R V V
viewer join
viewer find 2 in data. viewer's job done for 20%
redactor join
viewer join
viewer blocked
viewer blocked
redactor start working
redactor add 7 to data
redactor's job done!
viewer unblocked
viewer unblocked
viewer find 3 in data. viewer's job done for 20%
viewer find 4 in data. viewer's job done for 40%
viewer find 4 in data. viewer's job done for 40%
viewer find 4 in data. viewer's job done for 60%
viewer find 1 in data. viewer's job done for 80%
viewer find 4 in data. viewer's job done for 60%
viewer find 1 in data. viewer's job done for 80%
viewer find 7 in data. viewer's job done for 100%
viewer's job done!
viewer find 7 in data. viewer's job done for 100%
viewer's job done!

Process finished with exit code 0
```

```
5 5 4 3 2 1
6 R R R R R R
redactor join
redactor start working
redactor add 1 to data
redactor's job done!
redactor join
redactor start working
redactor add 4 to data
redactor's job done!
redactor join
redactor start working
redactor add 4 to data
redactor's job done!
redactor join
redactor start working
redactor add 4 to data
redactor's job done!
redactor join
redactor start working
redactor add 4 to data
redactor's job done!
redactor join
redactor start working
```

```
2 1 1
2 V R
viewer join
viewer find 1 in data. viewer's job done for 20%
viewer find 1 in data. viewer's job done for 40%
viewer find 1 in data. viewer's job done for 60%
viewer find 1 in data. viewer's job done for 80%
redactor join
viewer blocked
redactor start working
redactor add 4 to data
redactor's job done!
viewer unblocked
viewer find 1 in data. viewer's job done for 100%
viewer's job done!

Process finished with exit code 0
```

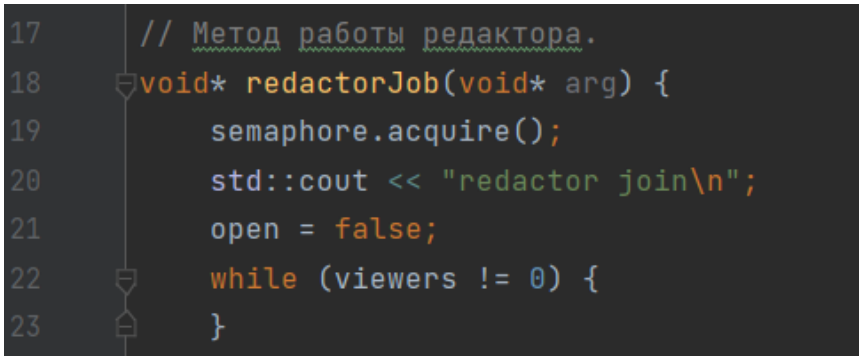
5 баллов:

- В программу добавлены комментарии, поясняющие выполняемые действия и описание используемых переменных.
- **Сценарий**, описывающий одновременное поведение представленных в условии задания сущностей в терминах предметной области:

Можно обойтись обычным мьютексом, но это не оптимально — компьютерная память, как правило, устроена так, что несколько потоков могут свободно читать и писать её (единственная проблема — нет гарантии, что в процессе обработки переменная внезапно не изменится). У этой проблемы есть несколько вариантов, разные и решения. Кому отдавать приоритет — читателю или писателю — остаётся за программистом.

В моем решении, реализованном при помощи **семафора**, абсолютный приоритет отдается писателям (редакторам).

То есть при подключении потока-редактора к ресурсу, доступ к нему закрывается и поток-редактор ожидает, пока все потоки-читатели не заблокируются:



Активные потоки-читатели же доделывают последнюю итерацию, после чего их доступ блокируется

до момента, пока поток-редактор не отключится от ресурса.

6 баллов:

- Реализован ввод данных из командной строки.
- Алгоритм работы:

«Читатели выполняют транзакции, которые просматривают записи базы данных»

Создаются потоки-читатели, которые выводят случайные значения из базы данных в консоль.

«Транзакции писателей и просматривают, и изменяют записи.»

Создаются потоки-редакторы, которые могут менять базу данных (добавлять в неё случайные значения)

«Предполагается, что в начале БД находится в непротиворечивом состоянии (например, если каждый элемент — число, то они все отсортированы).»

В моей решении база данных – отсортированный массив.

«Каждая отдельная транзакция переводит БД из одного непротиворечивого состояния в другое.»

После каждой итерации он сортируется вновь.

«Для предотвращения взаимного влияния транзакций процесс-писатель должен иметь исключительный доступ к БД.»

При помощи семафора регулируется количество активных потоков-писателей и читателей. Если активен один писатель, то доступ к базе данных закрыт.

«Если к БД не обращается ни один из процессов-писателей, то выполнять транзакции могут одновременно сколько угодно читателей.»

Если ни один писатель неактивен, то доступ к базе данных открыт.