

ИДЗ 4

Исполнитель: Чапурина Валерия
Сергеевна

Группа: БПИ237

Вариант: 4

Условие задания

Цели и задачи:

Изучить работу с потоками. Научиться разбивать задачу на части, для последующего их выполнения различными потоками.

Общие сведения

Потоки предоставляют возможность проведения параллельных или псевдопараллельных (в случае одного ядра) вычислений. Они могут порождаться во время работы программы, процесса или другого потока. Основное отличие потоков от процессов заключается в том, что различные потоки имеют различные пути выполнения, но при этом пользуются общей памятью. Таким образом, несколько потоков, могут пользоваться глобальными

переменными, и любое изменение данных одним потоком, будет доступно и для всех остальных. Существует несколько моделей построения многопоточных приложений.

Портфель задач. Один из распространенных способов динамического распределения работ. Как правило, реализуется с помощью разделяемой переменной, доступ к которой в один момент времени имеет только один поток или процесс.

4 **Задача о Винни-Пухе и мстительных пчелах.** Неправильные пчелы, подсчитав в конце месяца убытки от наличия в лесу ВинниПуха, решили разыскать его и наказать в назидание всем другим любителям сладкого. Для поисков медведя они поделили лес на участки, каждый из которых прочесывает одна стая неправильных пчел. В случае нахождения медведя на своем участке стая проводит показательное наказание и возвращается в улей. Если участок прочесан, а Винни-Пух на нем не обнаружен, стая также возвращается в улей. Там она получает информацию об еще неисследованных участках и снова улетаёт. Требуется создать многопоточное приложение, моделирующее действия пчел. При решении использовать парадигму **«портфель задач»**. Каждая стая пчел — отдельный поток.

Текст программы

лежит тут

Соответствие требованиям на оценку: 8

1 Оценка многопоточного приложения

Ниже представлены требования которым должно удовлетворять задание по многопоточному программированию для получения соответствующей оценки.

При отсутствии корректной реализации взаимодействия потоков, то есть, при некорректном поведении программы, задача считается нерешенной (0 баллов).

Программа должна быть разработана на языке программирования C или C++ с использованием для работы с потоками функций POSIX Thread. Использование более высокоуровневых библиотек классов и функций не допускается.

- ок, всё работает, POSIX Thread используем, более высокоуровневые библиотеки классов или функций не используются

Сценарий, описывающий одновременное поведение сущностей

4–5 баллов

- Соблюдены общие требования к отчету.
- В отчете должен быть приведен сценарий, описывающий одновременное поведение представленных в условии задания сущностей в терминах предметной области. То есть, описан сценарий, задающий ролевое поведение субъектов и объектов задачи (интерпретация условия с большей степенью детализации происходящего), а не то, как это будет реализовано в программе.

Давайте опишем этот сценарий:

Лес разделён на несколько участков, которые необходимо прочесать в поисках Винни-Пуха. Каждую стаю пчёл представляет отдельный поток. Все

стай используют общий портфель задач, где хранятся номера участков леса, которые пчёлы ещё не исследовали. Алгоритм действий следующий:

1. Каждая стая забирает очередной участок из портфеля задач для проверки
2. Если на участке обнаружен Винни-Пух, стая завершает свою работу, сообщая об успехе
3. Если Винни-Пух не найден, стая возвращается в улей и берёт следующую задачу из портфеля
4. Программа завершается, когда все участки прочёсаны или Винни-Пух найден

Модель параллельных вычислений

- Описана модель параллельных вычислений, используемая при разработке многопоточной программы.
- Каждая стая пчёл реализована как поток с использованием POSIX Threads → pthread в C++
- Портфель задач представлен очередью, защищённой мьютексом → pthread_mutex_t
- Для синхронизации вывода в консоль и файл используем отдельный мьютекс, чтобы в выводе строки не перемешивались
- Все потоки работают в рамках одного процесса, взаимодействуя через общую память

Входные данные

- Описаны входные данные программы, включающие вариативные диапазоны, возможные при многократных запусках.
1. Командная строка
 - `-f <output_file>`: имя выходного файла для записи результатов

- `-i <input_file>` : имя файла с номерами участков леса (опционально)
 - `-s <num_sections>` : количество участков леса (игнорируется, если указан файл ввода)
 - `-n <num_swarms>` : количество стай пчёл
2. если указан файл ввода, программа считывает номера участков из него. В противном случае участки генерируются автоматически (от 1 до заданного числа)
- Реализовано консольное приложение, решающее поставленную задачу с использованием одного варианта изученных синхропримитивов.
 - Ввод данных в приложение реализован с консоли во время выполнения программы (без использования аргументов командной строки).
 - Для используемых генераторов случайных чисел описаны их диапазоны и то, как интерпретируются данные этих генераторов.
 - реализовано
 - сразу использовались аргументы командной строки из критериев на оценку 8

Генераторы случайных чисел

Пчёлы находят Винни-Пуха на участке с шансом 5%. Генерируем при помощи `std::uniform_int_distribution` в диапазоне [1, 100]. Если получилось число > 95 - Винни-Пух найден, иначе - не найден, продолжаем поиск

- Вывод программы должен быть информативным, отражая все ключевые протекающие в ней события в терминах предметной области. Наблюдатель на основе вывода программы должен понимать, что в ней происходит в каждый момент времени ее работы.
- В программе присутствуют комментарии, поясняющие выполняемые действия и описание используемых объектов и переменных.
- Результаты работы программы представлены в отчете.

При невыполнении этих требований оценка снижается.

- ага, отражает, вот пример:
(тут 2 стаи и 5 участков, то есть
`-s 5 -n 2` в качестве аргументов из командной строки)

```
Swarm 2 is searching section 1...
Swarm 1 is searching section 2...
Swarm 1 found nothing in section 2. Returning to the hive.
Swarm 1 is searching section 3...
Swarm 2 found nothing in section 1. Returning to the hive.
Swarm 2 is searching section 4...
Swarm 1 found nothing in section 3. Returning to the hive.
Swarm 1 is searching section 5...
Swarm 2 found nothing in section 4. Returning to the hive.
Swarm 1 has found Winnie the Pooh in section 5!
All bee swarms have returned to the hive.
```

- поясняющие комментарии есть
- вот выше пример + ниже будет

Подробный обобщённый алгоритм

6–7 баллов

В дополнение к требованиям на предыдущую оценку и, в некоторых случаях вместо них:

- В отчете подробно описан обобщенный алгоритм, используемый при реализации программы исходного словесного сценария. В котором показано, как на программу отображается каждый из субъектов предметной области.

1. Инициализация

- Считываются входные параметры (командная строка или файл)
- Создаётся портфель задач с номерами участков
- Инициализируются мьютексы для синхронизации потоков и вывода

2. Работа потоков (стай пчёл)

- Каждый поток забирает участок из портфеля, проверяет его и возвращается
- Если Винни-Пух найден, поток обновляет глобальный флаг `winnie_found` и завершает работу
- Если на участке Винни-Пуха не оказалось, поток сообщает об этом и переходит к следующему заданию

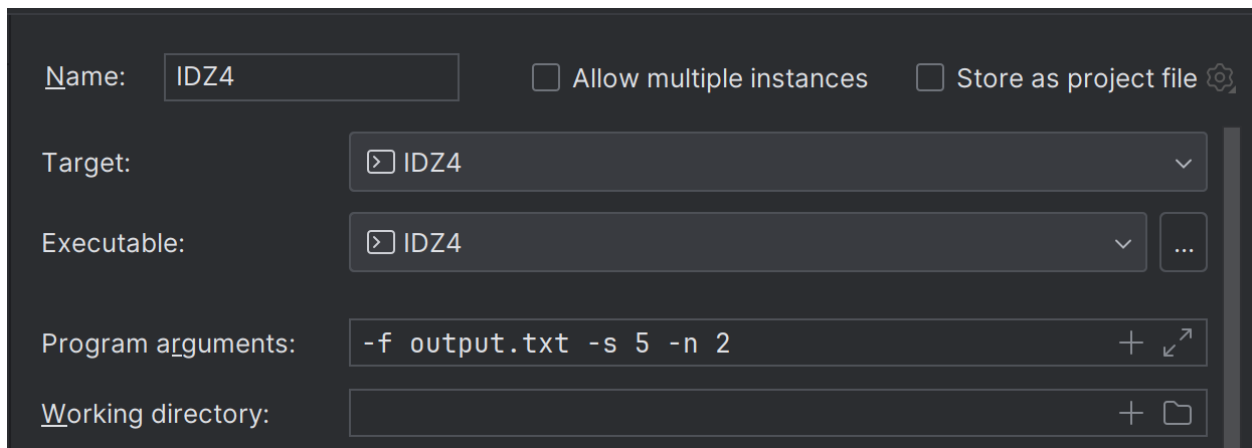
3. Завершение работы

- Главный поток ожидает завершения всех потоков через `pthread_join`
- Результаты работы записываются в файл и выводятся в консоль

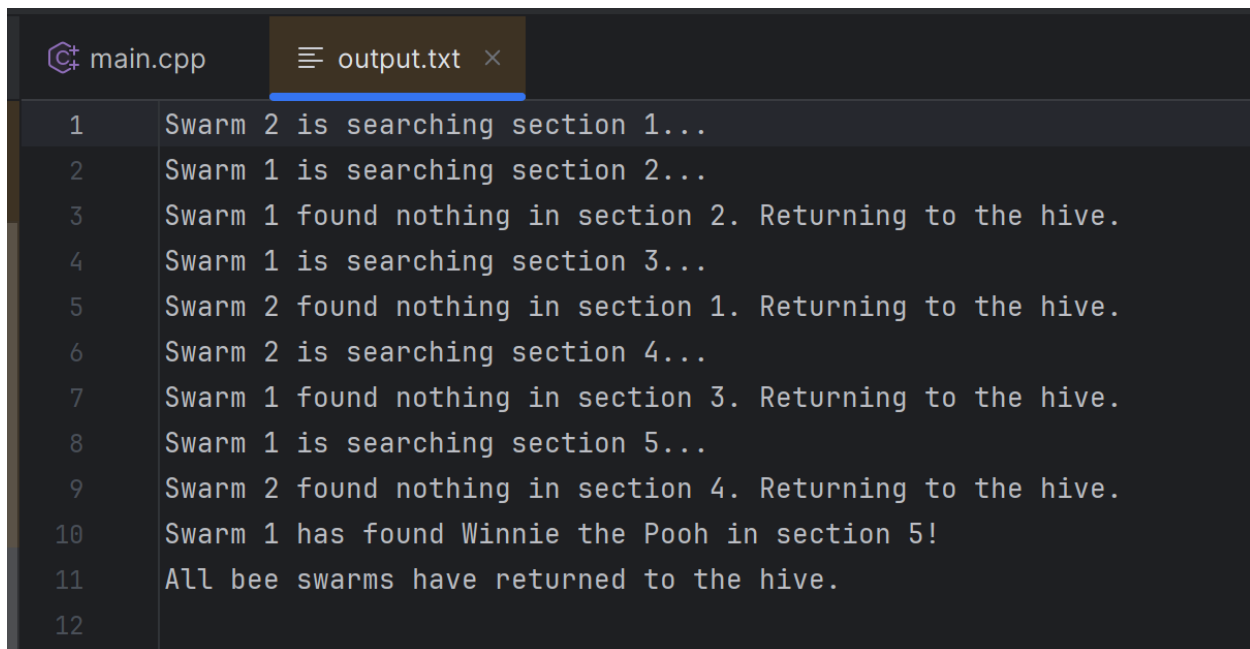
- В программу добавлена генерация случайных данных в допустимых диапазонах.
- Реализован ввод исходных данных из командной строки при запуске программы вместо ввода параметров с консоли во время выполнения программы.
- Результаты изменений отражены в отчете.

При невыполнении этих требований оценка снижается.

- угу
- да, вот так вводим исходные данные через командную строку (при помощи IDE CLion)



Результат выполнения программы записался в файл output.txt в той же директории, где хранится исполняемый код. То есть, в cmake-build-debug



```
1 Swarm 2 is searching section 1...
2 Swarm 1 is searching section 2...
3 Swarm 1 found nothing in section 2. Returning to the hive.
4 Swarm 1 is searching section 3...
5 Swarm 2 found nothing in section 1. Returning to the hive.
6 Swarm 2 is searching section 4...
7 Swarm 1 found nothing in section 3. Returning to the hive.
8 Swarm 1 is searching section 5...
9 Swarm 2 found nothing in section 4. Returning to the hive.
10 Swarm 1 has found Winnie the Pooh in section 5!
11 All bee swarms have returned to the hive.
12
```

8 баллов

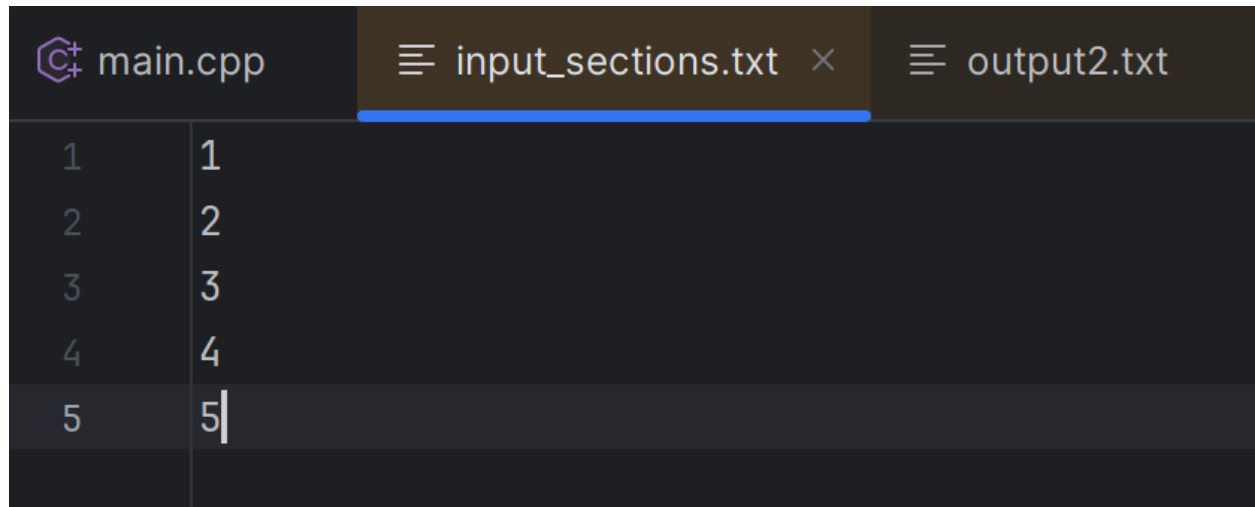
В дополнение к требованиям на предыдущую оценку

- В программу, наряду с выводом в консоль, добавлен вывод результатов в файл. Имя файла для вывода данных задается в командной строке как один из ее параметров.
 - Результаты работы программы должны выводиться на экран и записываться в файл.
-
- вот как раз выше пример
-
- Наряду с вводом исходных данных через командную строку добавлен альтернативный вариант их ввода из файла, который по сути играет роль конфигурационного файла. Имя этого файла задается в командной строке вместо параметров, которые в этом случае не вводятся. Управление вводом в командной строке осуществляется с использованием соответствующих ключей.

Такой вариант добавлен. Например, вот такие аргументы командной строки

```
Program arguments: -f output2.txt -i input_sections.txt -n 5
```

Пример содержания файла input_sections.txt:



```
1 1
2 2
3 3
4 4
5 5
```

Результат работы программы:

```
Project >
  > CMakeFiles
  > Testing
    .ninja_deps
    .ninja_log
    build.ninja
    cmake_install.cmake
    CMakeCache.txt
    IDZ4.exe
    input_sections.txt
    output.txt
    output2.txt
    .clang-format
    CMakeLists.txt

Run IDZ4 x
C:\Users\xiaom\CLionProjects\HSE\AKOS\ACS\IDZ4\cmake-build-debug\IDZ4.exe -f output2.txt -i input_sections.txt -n 5
Swarm 1 is searching section 1...
Swarm 2 is searching section 2...
Swarm 3 is searching section 3...
Swarm 4 is searching section 4...
Swarm 5 is searching section 5...
Swarm 5 found nothing in section 5. Returning to the hive.
Swarm 4 found nothing in section 4. Returning to the hive.
Swarm 3 found nothing in section 3. Returning to the hive.
Swarm 2 found nothing in section 2. Returning to the hive.
Swarm 1 found nothing in section 1. Returning to the hive.
All bee swarms have returned to the hive.

Process finished with exit code 0
```

Примеры

- Приведено не менее трех вариантов входных и выходных файлов с различными исходным данными и результатами выполнения программы.
- Ввод данных из командной строки расширен с учетом введенных изменений.
- Результаты изменений отражены в отчете.

При невыполнении хотя бы одного из требований оценка снижается на балл.

- 2 уже было показано в отчёте выше

- результат для `-f output3.txt -s 20 -n 4` (побольше участков чтобы было)

```
↓
Swarm 1 is searching section 1...
⇌
Swarm 2 is searching section 2...
⇌
Swarm 3 is searching section 3...
⇌
Swarm 4 is searching section 4...
⇌
Swarm 4 found nothing in section 4. Returning to the hive.
⇌
Swarm 1 found nothing in section 1. Returning to the hive.
⇌
Swarm 1 is searching section 6...
⇌
Swarm 4 is searching section 5...
⇌
Swarm 3 found nothing in section 3. Returning to the hive.
⇌
Swarm 3 is searching section 7...
⇌
Swarm 2 found nothing in section 2. Returning to the hive.
⇌
Swarm 2 is searching section 8...
⇌
Swarm 2 found nothing in section 8. Returning to the hive.
⇌
Swarm 4 found nothing in section 5. Returning to the hive.
⇌
Swarm 1 has found Winnie the Pooh in section 6!
⇌
Swarm 4: Winnie the Pooh has already been found. Returning to the hive.
⇌
Swarm 2 is searching section 9...
⇌
Swarm 3 found nothing in section 7. Returning to the hive.
⇌
Swarm 3: Winnie the Pooh has already been found. Returning to the hive.
⇌
Swarm 2 found nothing in section 9. Returning to the hive.
⇌
Swarm 2: Winnie the Pooh has already been found. Returning to the hive.
⇌
All bee swarms have returned to the hive.

Process finished with exit code 0
```

Все файлы с выводом лежат в репозитории в папке tests