Вариант 8

Задание

Военная задача. Анчуария и Тарантерия — два крохотных латиноамериканских государства, затерянных в южных Андах. Диктатор Анчуарии, дон Федерико, объявил войну диктатору Тарантерии, дону Эрнандо. У обоих диктаторов очень мало солдат, но очень много снарядов для минометов, привезенных с последней американской гуманитарной помощью. Поэтому армии обеих сторон просто обстреливают наугад территорию противника, надеясь поразить минометы противника. Стрельба ведется хаотично до полного 3 уничтожения всех минометов противника, размещенных на некоторой прямоугольной площадке размером N*N. То есть координаты целей, хоть и случайные, согласуются между разными минометчиками одной армии. Интервалы между выстрелами (задержки) зависят от расстояния до точки попадания, формируемой случайно. Повторная стрельба по одним и тем же координатам не производится. Создать приложение, моделирующее военные действия. Каждый миномет задается отдельным процессом. Их количество у каждой стороны одинаково и задается аргументом командной строки. Размер поля также задается в командной строке. Расположение минометов порождается случайно.

Отчет :

Преведено решение задачи на языке C на отметки 4, 5, 6, 7 соответственно.

Все решения протестированы и успешно работают, каждая версия программы была протестирована на 3 тестовых наборах, и получену результаты победы обеих команд. Падений программы и других пробелм в ходе тестирования не выявлено.

Для компиляции использовать:

```
gcc main.c -o main
```

Для запуска программы использовать:

```
./main "size of the battlefield" "count of the mortars"
Пример запуска:
./main 5 2
```

Поясню решение для следующих оценок:

4 балла:

Описание работы:

```
0) !!! Использую именованные семафоры !!!
1) Инициализирую семафоры, и подготавливаю пустое поле боя.
2) От изначального процесса создаю n дочерних(делаю это все в цикле), заставляю
дочерние процессы ждать завершения создания всех дочерних процессов, также при
создании каждого дочернего процесса заношу его данные на поле боя(сохраняю его Pid и
номер команды), здесь же происходит генерация его местоположения.
3) После создания всех процессов добавляю поле боя в общую память всех процессов.
```

(field)
4) Далее главный поток засыпает, до завершения всех потоков-детей.(Потоки-дети представляют собой орудия, которые перестреливаются друг с другом на поле боя).
5) Далее использую семафоры для синхронизации перестрелки(leftMortars, rightMortars), перестрелка основана на принципе, что 2 команды могут одновременно вести огонь друг по другу, но при этом из каждой единовременно команды может стрелять только 1 орудие(это реализация механизма синхронизации между различными орудиями одной команды).
Если какое-то орудие выпустило снаряд, то пока он "летит", орудие другой команды может успеть выстрелить, а может и не успеть.
Также использую printData семафор, для корректного вывода в консоль(чтобы сразу несколько потоков не печатали информацию), здесь можно было использовать mutex, но задание на семафоры, так что пришлось использовать не совсем верный подход.
После завершения перестрелки уничтожаю все процессы оружия, а далее в основном потоке после пробуждения очищаю потребовавшиеся ресурсы.

5 баллов:

Логика точно такая же, единственное изменение - это замена именованных на неименованные семафоры.

6 баллов

Логика точно такая же, единственное изменение - это использование стандарта SYSTEM POSIX V.

7 баллов

На 7 баллов программа была разделена на 2 части : main.c и weapon.c, где weapon отвечает за действия оружия.

P.S. Также стоит отметить, что в коде присутсвует большое количество комментариев, так что хорошей проверки)