Галкина Таисия Олеговна БПИ213
27 вариант

Условие:
Задача про экзамен. Преподаватель проводит экзамен у группы студентов. Каждый студент получает свой билет, сообщает его
номер и готовит письменный ответ. Подготовив ответ, он передает его преподавателю. Преподаватель просматривает ответ и сообщает студенту оценку. Студент, дождавшись результата, уходит с
экзамена. Требуется создать приложение, моделирующее
действия преподавателя и студентов, каждый из которых представлен отдельным процессом.

Оценка 4 - именованные POSIX семафоры. Обмен данными также ведется через разделяемую память в стандарте POSIX.

В данном сценарии исходные сущности - студенты и преподаватель - отображаются в виде процессов.
Каждый процесс-студент выполняет следующие действия:

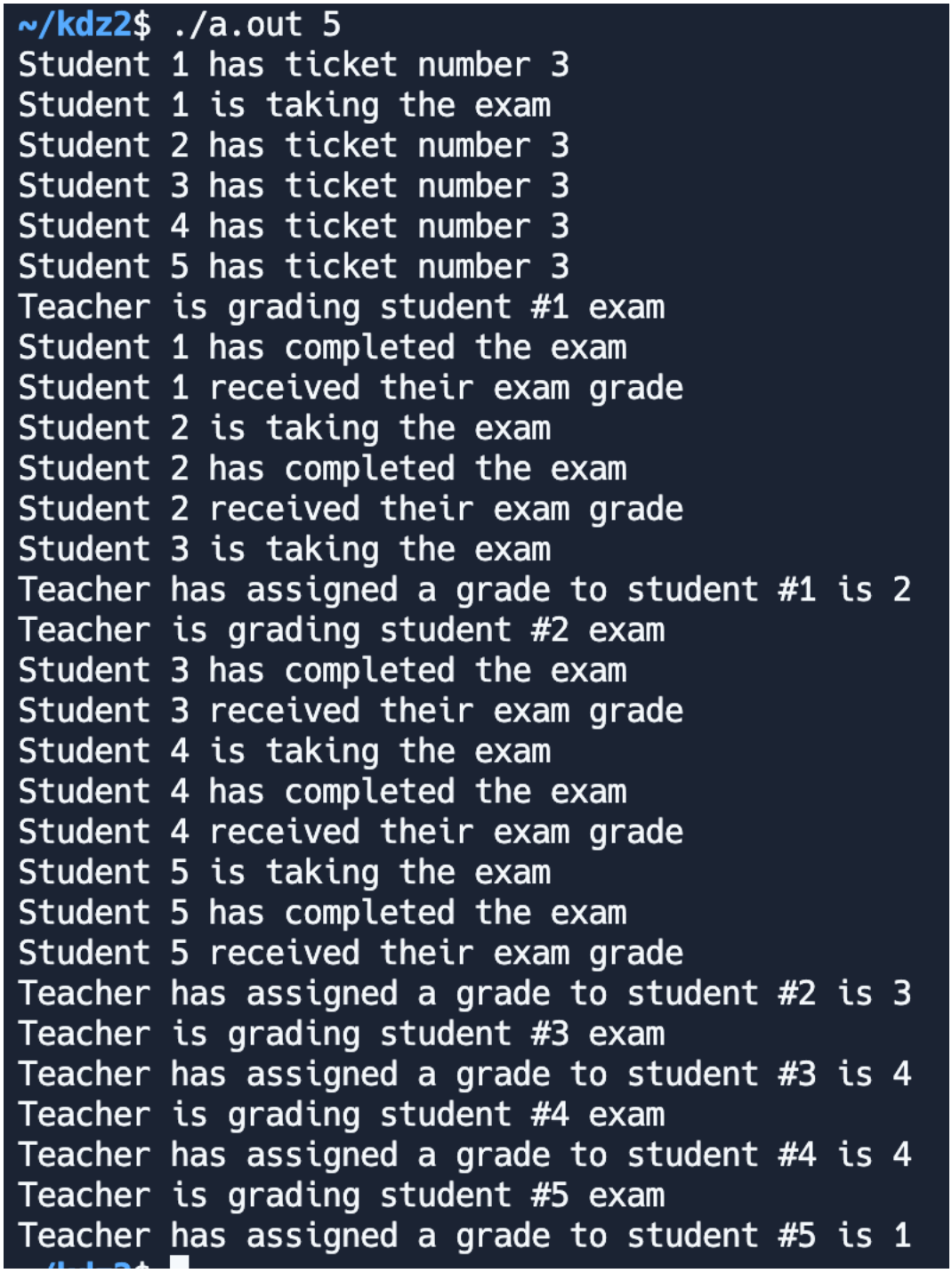
1. Генерирует случайный номер билета.
2. Ждет, пока будет свободен семафор экзамена.
3. Проходит экзамен, затрачивая на это случайное время.
4. Ждет, пока будет свободен семафор результатов.
5. Получает свою оценку.
6. Освобождает семафор экзамена и завершает свою работу.

Процесс-преподаватель выполняет следующие действия:

1. Ждет, пока будет свободен семафор результатов.
2. Выставляет оценку для работы, для которой семафор был освобожден.
3. Освобождает семафор результатов.

Для взаимодействия между процессами используются два семафора: семафор экзамена и семафор результатов. Семафор экзамена используется для того, чтобы ограничить количество студентов, которые могут одновременно сдавать экзамен. Семафор результатов используется для того, чтобы преподаватель мог проверять работы студентов только после того, как они сдали экзамен и получили свою оценку.

Также у нас предусмотрено завершение программы с помощью команды ctrl + c. А значит 5 пункт тоже выполняется. Итого все условия выполняются.
Пример выполнения программы:
Мы в консоль вводим количество студентов



Оценка 5 - неименованные POSIX семафоры. Обмен данными также ведется через разделяемую память в стандарте POSIX.

Вся логика точно такая же, однако тут мы используем неименованные семафоры.

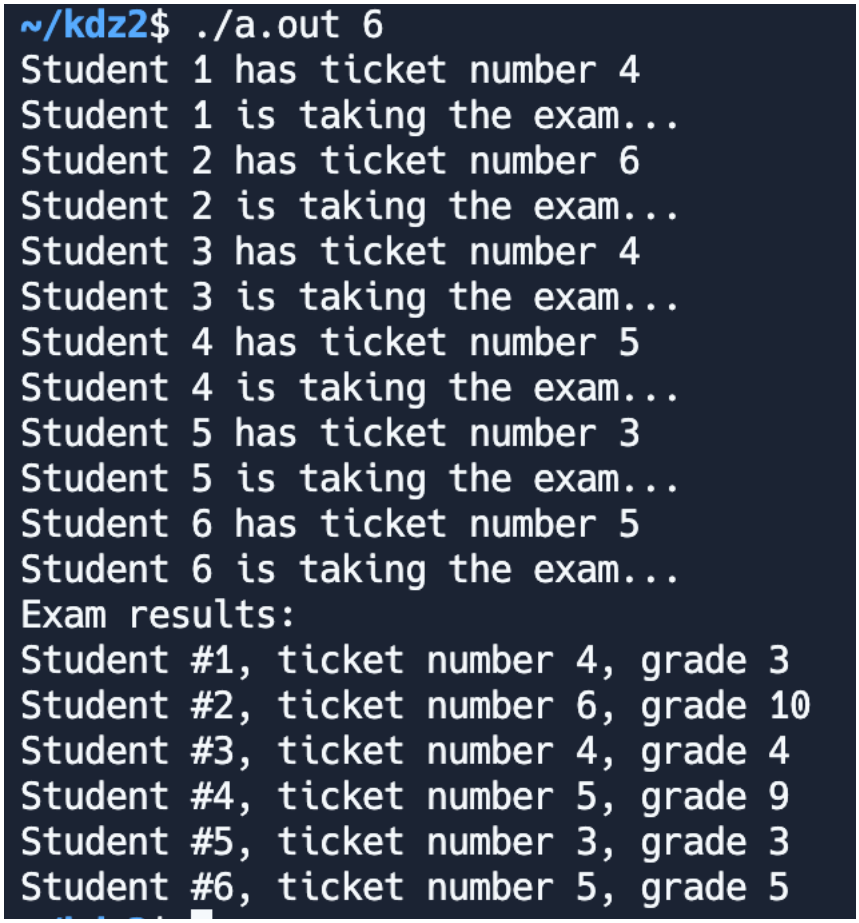
Пример выполнения программы:

```
~/kdz2$ ./a.out 5
Student 1 has ticket number 2
Student 1 is taking the exam
Student 4 has ticket number 2
Student 4 is taking the exam
Student 3 has ticket number 2
Student 3 is taking the exam
Teacher is grading student #1 exam
Student 5 has ticket number 2
Student 5 is taking the exam
Student 2 has ticket number 2
Student 2 is taking the exam
Teacher has assigned a grade to student #1 is 5
Teacher is grading student #2 exam
Student 4 has completed the exam
Student 4 received their exam grade
Student 2 has completed the exam
Student 2 received their exam grade
Student 1 has completed the exam
Student 1 received their exam grade
Student 3 has completed the exam
Student 3 received their exam grade
Student 5 has completed the exam
Student 5 received their exam grade
Teacher has assigned a grade to student #2 is 1
Teacher is grading student #3 exam
Teacher has assigned a grade to student #3 is 3
Teacher is grading student #4 exam
Teacher has assigned a grade to student #4 is 1
Teacher is grading student #5 exam
Teacher has assigned a grade to student #5 is 2
```

Оценка 6 - UNIX SYSTEM V семафоры. Обмен данными ведется через разделяемую память в стандарте UNIX SYSTEM V.

Вся логика точно такая же, однако тут мы используем UNIX SYSTEM V семафоры.

Пример выполнения программы:



Оценка 7 - именованные POSIX семафоры. Обмен данными также ведется через разделяемую память в стандарте POSIX.

Также можно экстренно закончить программу через ctrl + c.

В данном случае у нас две программы :

1. st.c – программа, в которой выполняется вся логика студентов, то есть оттуда мы узнаем(передаем программе учителя через разделяемую память количество студентов) сколько всего студентов. После создаем ощущение реального экзамена.
2. tch.c – программа, в которой выполняется логика учителя. Через разделяемую память мы узнаем сколько у нас студентов, а после генерируем им какие-то оценки. Также с помощью sleep мы создаем ощущение, что учитель проверяет оценку.

Результаты выполнения программы:

Выполнение программы st.c:

```
^C~/kdz2$ ./st 5
Student 1 has ticket number 5
Student 1 is taking the exam
Student 2 has ticket number 5
Student 4 has ticket number 5
Student 3 has ticket number 5
Student 5 has ticket number 5
Student 1 has completed the exam
Student 1 received their exam grade
Student 2 is taking the exam
Student 2 has completed the exam
Student 2 received their exam grade
Student 4 is taking the exam
Student 4 has completed the exam
Student 4 received their exam grade
Student 3 is taking the exam
Student 3 has completed the exam
Student 3 received their exam grade
Student 5 is taking the exam
Student 5 has completed the exam
Student 5 received their exam grade
```

Выполнение программы tch.c:

```
~/kdz2$ ./t
Teacher is grading student #1 exam
Teacher has assigned a grade to student #1: 1
Teacher is grading student #2 exam
Teacher has assigned a grade to student #2: 8
Teacher is grading student #3 exam
Teacher has assigned a grade to student #3: 1
Teacher is grading student #4 exam
Teacher has assigned a grade to student #4: 3
Teacher is grading student #5 exam
Teacher has assigned a grade to student #5: 4
```

Сначала нужно запустить программу со студентами, а только потом с учителем, поскольку надо сначала записать в память количество студентов.