ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

ЛАБОРАТОРНАЯ РАБОТА № 3

Выполнил:

Студент группы: ИП-712

Алексеев С.В.

Проверил: профессор кафедры ПМиК

Фионов А.Н.

Новосибирск – 2020

1. Электростанция состоит из следующих элементов: хранилище топлива (1 шт.), транспортное средство (1 шт.), котлы (4 шт.). Элементы станции работают параллельно, каждый по своей программе (что может быть реализовано с помощью нитей). Транспортное средство доставляет топливо из хранилища к котлам. Топливо имеет различные марки (от 1 до 10). Топливо марки 10 горит в котле 10 с (условно), в то время как топливо марки 1 горит всего 1 с. Необходимо написать программу, моделирующую работу электростанции и показывающую на экране процесс ее функционирования. Используя функции библиотеки VinGraph, нарисовать абстрактную картину, которой представлены (почти) все доступные графические элементы.

Для реализации данного задания необходимо написать программу с обменом сообщениями (между потоками) и использованием POSIX Threads (pthread.h).

Сообщения не передаются между потоками напрямую, поэтому для обмена сообщениями между потоками используются каналы и соединения. Канал можно создать с помощью функции ChannelCreate(). После этого можно вызывать MsgReceive() и MsgReply().

ChannelCreate(unsigned flags) принимает один аргумент-флаг или комбинацию из них.

MsgReceive(int chid, void \* msg, int bytes, struct \_msg\_info \* info) принимает 4 аргумента. chid, ID, который мы получаем после вызова ChannelCreate(), msg – указатель на буфер с данными, bytes – размер буфера, info – NULL или указатель на структуру \_msg\_info, где может хранится дополнительная информация о сообщении.

Принимающий поток может создать соединение к каналу другого потока используя ConnectAttach(), а потом уже вызывать MsgSend(). Получается своеобразная схема “клиент-сервер’’.

ConnectAttach(uint32\_t nd, \_t pid, int chid, index, int flags) устанавливает соединение между потоком и каналом. nd – дескриптор, pid – ID процесса-владельца каналом, chid - chid, ID, который мы получаем после вызова ChannelCreate(), index – наименее возможный ID соединения, flags – флаги.

MsgSend(int coid, const void\* smsg, int sbytes, \* rmsg, int rbytes) отправка сообщения в канал. coid – ID, полученный от ConnectAttach. smsg – указатель на буфер сообщения, sbytes – кол-во байтов на отправку, rmsg – указатель на буфер с ответом, rbytes – размер буфера ответа.

Деятельность транспорта (carThread), склада топлива (fuelTankThread) и котлов (boilerRoom) реализована в разных потоках. Графика реализована с помощью VinGraph и таймера для отображения анимации.

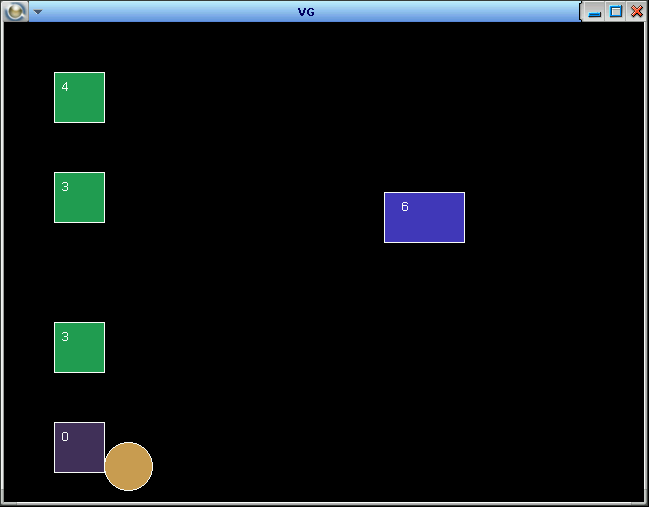


Рис.1 – Процесс работы программы с одним транспортным потоком

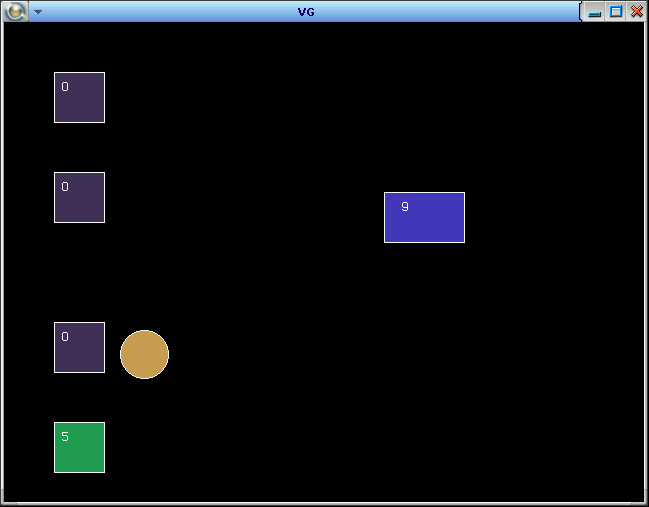
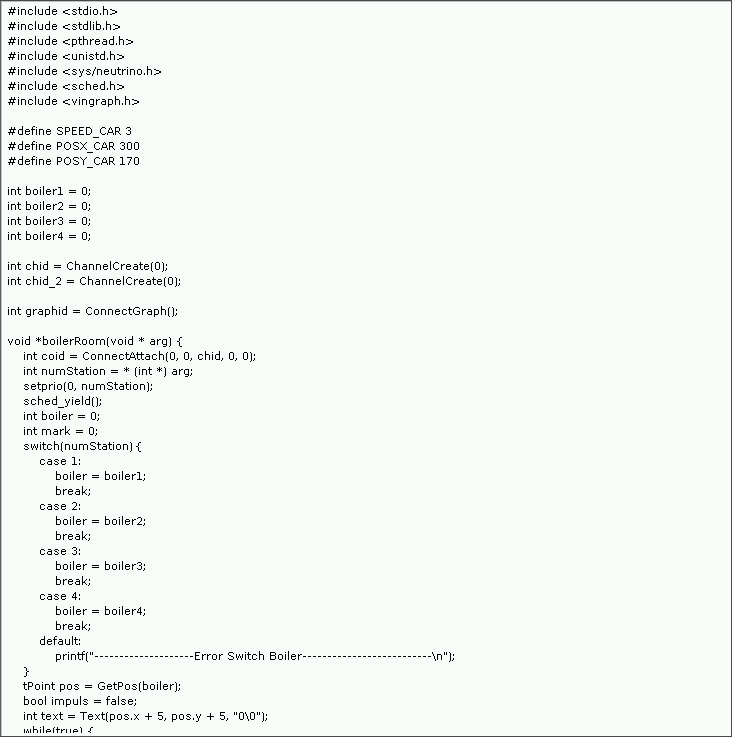
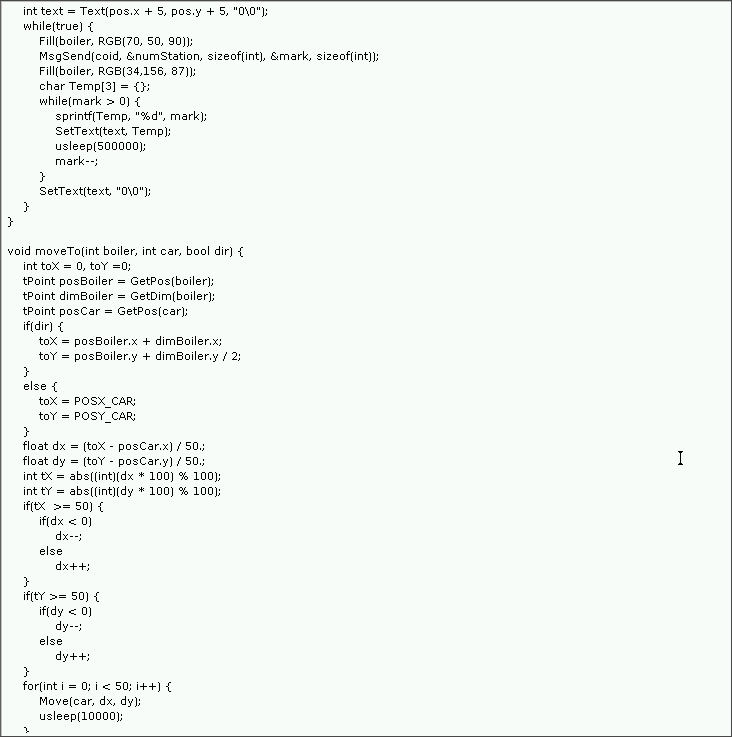
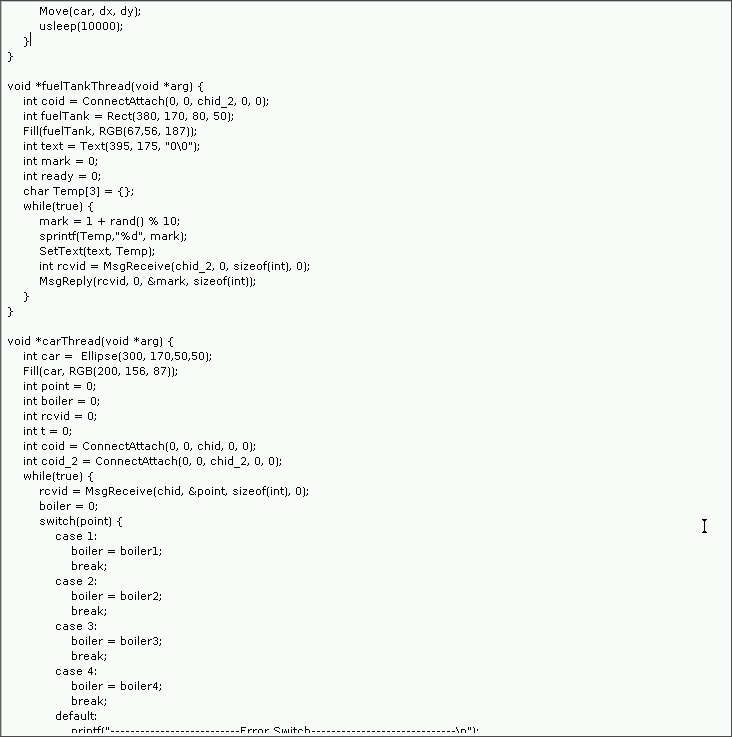
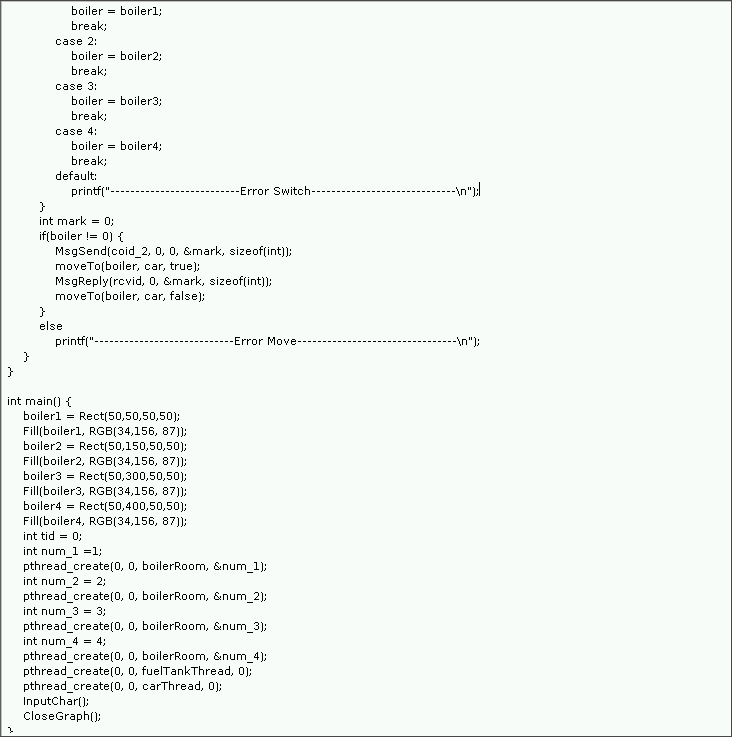


Рис.2 – Процесс работы программы с одним транспортным потоком









1. А теперь добавьте второе транспортное средство.

Второе задание отличается от первого лишь добавлением дополнительного потока с транспортом.

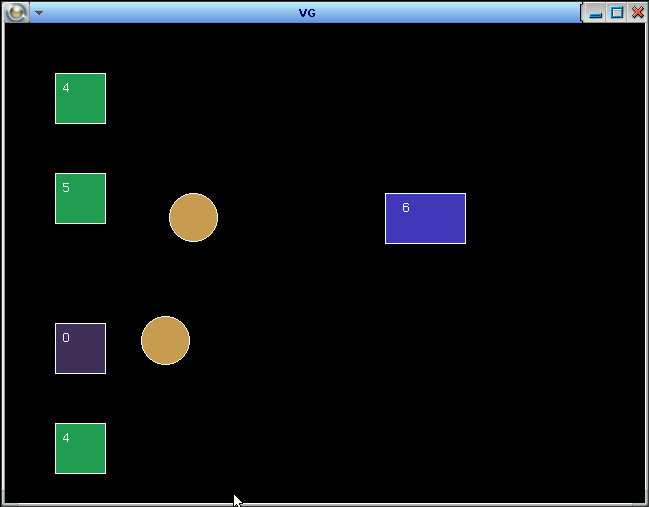
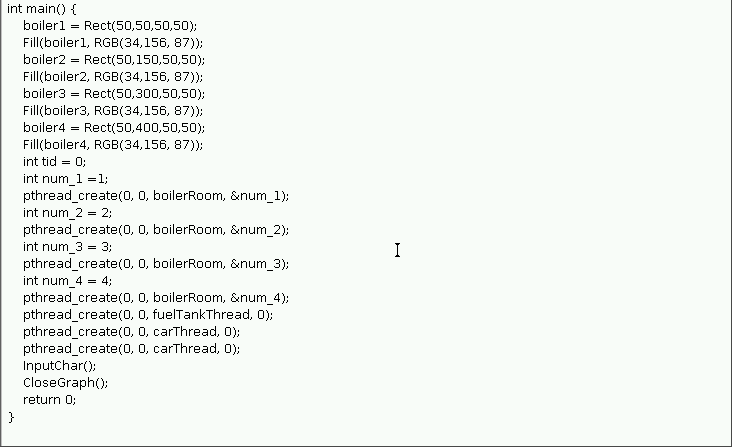


Рис.3 – Процесс работы программы с двумя транспортными потоком



1. (использование импульсов) Регулируя скорости работы элементов электростанции, вы можете создать ситуацию, когда котлы будут простаивать из-за низкой скорости подвоза топлива. Создайте такую ситуацию. Теперь сделайте так, чтобы топливо подвозилось к котлам заранее, до момента их полной остановки. Это можно реализовать, если котлы будут сообщать о том, что топливо скоро кончится (например, его осталось на 2 с работы). Ясно, что котлы могут это сделать с помощью импульса, т.к. обычное сообщение их заблокировало бы, в то время как они должны продолжать работать.

Программа реализована с использованием pulse-методов. Они оповещают потоки о необходимости поставки.

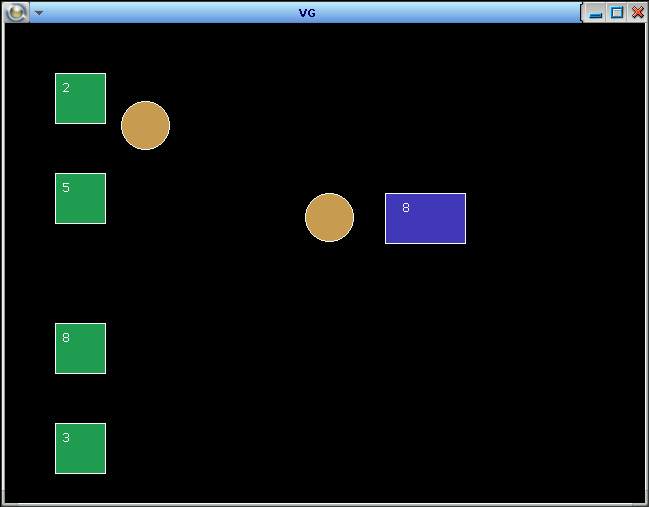


Рис. 4 – Процесс работы программы с импульсами (транспорт двигается к котлу с топливом < 3)

MsgSendPulse(int coid, int priority, int code, int value) – отправляет импульс к потоку.

