

CS 33211 Operating Systems

Project #1

Due via e-mail by 11:59pm on Monday, March 5, 2007

Preliminaries

In the project you will develop programs that manipulate threads. You write your programs in the environment and on the platform of your choice. However, your submission should compile with the gcc compiler on the department Linux machines: `neptune` or `poseidon`. For program development you can use either C or C++. Your program would use POSIX thread library (supply `-pthread` option to the compiler). Your code should have good programming style and comments. Programs written with poor programming style may receive lower grade. The project assignment contains two parts.

Programming Assignment

1. Printing letters. Write a program that accepts a phrase of unspecified length on the command line. For example:

```
prompt% letterprint Operating Systems Class at Kent-State
```

The `main()` in this program has to create two threads running functions (`vow` and `cons`). The threads should take turns printing the respective words of the phrase supplied on the command line. The `vow` thread should print all the words that start with a vowel and the `cons` thread should print all words starting with a consonant. Note that the main thread (running `main()`) should not print anything itself – the output should be supplied by the threads that it creates. The order of words in the phrase should not be changed in the printout. Your program should work for any phrase of any reasonable length, not just the one given in the example. The output of your program should look similar to the following

```
prompt% letterprint Operating Systems Class at Kent-State
vow:      Operating
cons:     Systems
cons:     Class
vow:      at
cons:     Kent-State
```

In this part you are not allowed to use synchronization primitives such as mutexes for thread coordination. You can use `sched_yield()` to relinquish control of the CPU.

2. Road construction. Ohio Department of Transportation was not satisfied with the results of the recent construction project in the Main Street next to Kent State University and started the construction there again. Only one lane is left operational during the construction work. The width of the lane is 10 meters and the length of the construction zone is 100 meters. The cars going in the opposite directions have to coordinate their movement. Besides, the students from the university like to go to the Susan's coffee shop across the street for lunch. All this created quite an inconvenience for everybody. ODOT installed traffic lights on both sides of the one-lane section of the road and by the pedestrian crosswalk. ODOT also hired you to write a program to coordinate the work of the traffic lights according to the following rules:

- neither car nor pedestrians should wait if the intersection is empty;
- cars cannot go in the opposite directions simultaneously on the one-lane section;
- a pedestrian cannot cross the street while there is a car in the one-lane section, but multiple pedestrians can cross the street at the same time;
- a car may enter the one lane section if there is a car there going in the same direction, however, a car is not allowed to pass another car;
- a car does not wait for more than two cars going in the opposite direction;

- a pedestrian has to yield to cars BUT a pedestrian should not wait for more than tree cars (in either direction).

Write a program `constructionzone` that coordinates the traffic in the construction zone. Your program should read a file called `traffic.txt` which contains the information about the traffic arriving at the construction zone. The example file is as follows

```
0 E1 10
1 P1 1
4 E2 15
5 W1 10
```

The file has the following format. Each line corresponds to either car or pedestrian. Each entry is a space separated list of three fields. First entry – number of seconds after the previous participant arrived (the first line always has 0). The number of seconds is always an integer. Second entry consists of the letter and a number. The letter is: E – car going East, W – car going West, or P – pedestrian (the letters are always capital). The number is the sequence number of the participants. The last entry is the speed (in meters per second) of the participant inside the construction zone. This entry is needed to calculate the amount of time (in seconds) each participant spends in the zone. Note that pedestrians have to cross the street (10 meters) and the cars have to ride along it (100 meters). The number of seconds in the construction zone should be rounded up.

Create three functions: `pedestrian()`, `carWest()`, `CarEast()`. Your program should read the input file, and launch a thread per each traffic participant the appropriate time (consider using `sleep()` system call). Each thread should execute corresponding function.

Each function should contain three sections: zone entrance, zone crossing, zone exit. Use mutexes and condition variables to ensure that the traffic crosses the construction zone according to the ODOT rules. After the entrance and exit the thread should announce (print) its action. Use `sleep()` system call to implement crossing the zone. The amount of time that each participant remains in the zone corresponds to his speed.

The output of your program should look as follows:

```
prompt% constructionzone
E1 entering construction
E2 entering construction
E1 exiting construction
E2 exiting construction
P1 entering construction
P2 exiting construction
W1 entering construction
W1 exiting construction
```

Getting Help

Help is available from the instructor (Mikhail Nesterenko) and the TA (Swetha Vasudevan). The easiest way to reach us is through e-mail. Both of us have our office hours listed on the course's webpage. If you need a consultation outside office hours please make an appointment. The office hours may be extended as the project deadline approaches:

- for clarifications on the assignment itself contact the instructor;
- with questions on thread operation contact either the instructor or the TA;
- for help with your code or debugging, contact the TA.

Submitting Your Project

You have to submit your project to the teaching assistant by e-mail. She will acknowledge the receipt of your project within a day. Send her only the source files (you may include your test input file for the second part). Pack the files into a compressed file archive. Here is an example command line:

```
prompt% tar czvf project1.tgz letterprint.c constructionzone.c  
        traffic.txt
```

Grading

The programming project is worth 50 points. Late projects are accepted. Late policies are outlined in the syllabus. The grading will be such that the working program submitted a few days late will receive a higher grade than the program that does not work but is submitted. If your program still does not work, submit a text file specifying how the program fails (does not compile, fails test examples, etc.), what you think the reason is and how it can be fixed. You may be given partial credit.