Northwestern University
Department of Electrical Engineering and Computer Science

Assignment 4

EECS 231 Advanced Programming
Fall Quarter 2006

DUE: Monday 11/13, 11:59pm

Goals of this assignment

- Practice file I/O
- Apply your knowledge of inheritance and abstract classes
- Learn about time-driven simulations

Time-driven simulations

A time-driven simulation is based on the simulated ticking of a clock. At each step of the simulation, time is incremented by a fixed amount. After each step, we check whether any event happens and if so, handle it. A time-driven simulation typically ends when the time has reached a particular value or when a certain event has occurred.

In this assignment you will use a time-driven simulation to model part of a company's product distribution process, collect data on distribution times and calculate statistics.

Project details

We are interested in the transportation of goods from a factory to a distribution center, and in particular how long it takes from the time an item is produced to the time it arrives to the distribution center. The company uses a fleet of trucks and trains for the transportation. You may assume that the fleet will never have more than 10 vehicles.

Details on how the operation runs follow.

Trucks

All trucks are initially stationed at the factory. Each truck has an initial capacity and will take a certain amount of time, k to travel between the factory and the distribution center.

A truck waits at the factory until it is fully loaded. At that point, it starts traveling towards the distribution center. A truck that is loaded at time t will depart the factory immediately and arrive at the distribution center at time $t + k$.

When the truck reaches the distribution center, it is unloaded and then driven back to the factory. Travel in the two directions takes the same amount of time. Unloading the truck takes one time unit. For example, a truck that arrives at the distribution center at time t will leave at time $t + 1$ and return to the factory at time $t + k + 1$.

At any time the truck is either at the factory, or at the distribution center or in transit towards one of these two destinations.

Trains

All trains are initially stationed at the factory. Assume there are enough tracks. Each train has an initial capacity and will take a certain amount of time to travel between the factory and the distribution center.

Furthermore, each train has a fixed departure time and runs on a 24-hour shedule. The train always departs the factory at the specified time, regardless of its load.

When the train reaches the distribution center, it is unloaded and immediately travels back to the factory. Travel in the two directions takes the same amount of time. Unloading a train takes one time unit.

At any time the train is either at the factory, or at the distribution center or in transit towards one of these two destinations.

Packages

All packages are assumed to have the same volume and weight. The capacity of each vehicle is measured in "package units". The only information we want to keep track of is the time each package was ready for shipping and the time it arrived at the distribution center. Note that the time a package is ready for shipping is not necessarily the same as the time the package is loaded on a vehicle. If there is no vehicle at the factory, then packages pile up until a truck or train arrives.

Simulation

The simulation will run as follows:

- Initialize the fleet (see "Input" section below)
- At each time unit:
  - Examine how many packages are ready for shipping.
    - If there is a vehicle available, load the packages. If not all packages fit, load the rest in another vehicle, if available. Traverse the vehicles in the same order they were "entered" in the fleet.
    - If there are no vehicles at the factory at this time, do not load the packages yet.
  - Then, examine all vehicles. If a train at the factory is scheduled to depart, then it should. If a truck at the factory is full, have it depart for the distribution center.

    If a train or truck has reached the distribution center, unload it and prepare to turn it back.

    If a train or truck has reached the factory, make it available for loading (but do not start loading at this time unit).

    When unloading a vehicle, go through the packages and mark their arrival time. We are interested in the following data:
    - minimum elapsed time between production and arrival at the distribution center.

    * maximum elapsed time between production and arrival at the distribution center.
    * average elapsed time between production and arrival at the distribution center.
- At some point, the factory will stop producing packages. When that happens, the simulation should go on until all packages have reached the distribution center. (It does not matter if at that time there empty vehicles still in transit.)

  You may assume that the input data will be such that there won't be any trucks left waiting half-loaded at the factory.

  When the simulation stops and the statistics should be printed out. See also the section titled "Debugging Output", below.

Input

The data file consists of two sections:

The first section contains the data you need to initialize the simulation. This is a sequence of lines that contain the type of vehicle (TRAIN or TRUCK), the capacity of the vehicle, the time it takes for it to travel between the factory and the distribution center and, for trains only, the scheduled departure time (on a 24-hr scale). The end of this list is marked with the word END.

The second section contains a sequence of integers representing the number of packages generated at each time unit. The end of the sequence is marked with a negative number.

For example, a sample input file could read:

```
TRAIN 10 3 6
TRUCK 1 2
TRUCK 2 2
TRUCK 4 2
END

3
0
5
4
2
0
3
1
-1
```

Debugging output

If the user wishes, your program should also create a file that contains debugging output. Here is some sample debugging output:

```
TIME: 4
    4 package(s) have been produced.
    A total of 4 package(s) are ready for loading.

    Preparing to load...
```

```
2 package(s) have been loaded on vehicle 0
1 package(s) have been loaded on vehicle 1
1 package(s) have been loaded on vehicle 2
Finished loading

Updating state of vehicles...
Vehicle 0 (train) is at the warehouse but is not ready to leave yet.
Vehicle 1 (truck) is at the warehouse and is leaving NOW for the distribution center.
Vehicle 2 (truck) is at the warehouse but is not ready to leave yet.
Vehicle 3 (truck) is at the warehouse but is not ready to leave yet.
```

An executable will be provided so that you can see exactly what output your program is expected to generate. You must match the debug output EXACTLY!

The name of the output file should be the same as the name of the input file but with extension `.debug`. For example, if the input file is called `data1.in` then the output file should be called `data1.debug`. You will provided sample code for this.

Execution

A switch `-d` that turns on debugging as well as the input file name should be passed as command-line arguments.

Designing your simulation

You will need a Package class for the packages, as well as a Vehicle abstract base class with Train and Truck derived classes. Use the description above to decide what attributes each type of object should have.

In addition, create a Simulation class to control and run the simulation. An object of this class will be in charge of "moving" vehicles and collecting statistical information. We will discuss this further in class.

START EARLY! This assignment is much more involved than previous ones.

Files

Make sure you create separate `.h` and `.cpp` files. Furthermore, you must provide a makefile. You may use the one from the gdb lab as a blueprint.

Follow the usual commenting guidelines.

Submitting your files

tar or zip your `Makefile`, `.h` and `.cpp` files and email them to `ido715@ece.northwestern.edu` as usual. Do not neglect to cc yourself and verify that the file was attached properly and was not corrupted in any way.