2003/2004 SOUTHERN CALIFORNIA REGIONAL
ACM INTERNATIONAL COLLEGIATE PROGRAMMING CONTEST

Problem 1
Stale Data Removal

Modern data acquisition systems obtain and record digitized data from multiple instruments that are connected via a communications bus. The data acquisition system essentially acts as a polling device, requesting sampled data values from digitizing instruments. The instruments sample and digitize their values only at specific, periodic times, and simply report the most recently sampled values to the data acquisition system. The effect of this technique reduces a continuous, time-varying signal to a sample-and-hold model, as shown in Figure 1-1.

A straightforward method of acquiring data polls every instrument at the same rate as the instrument with the highest sample rate. The polling of the acquisition system may then be seen as sampling the already sampled-and-held signal. Figure 1-2 depicts the polling of the instrument for Figure 1-1, which samples more slowly than the polling rate, wherein lies the problem: The act of polling any sensor that samples slower than the polling rate produces some repeated values, known as stale data. For instance, an onboard instrument may sample a wing flap position 10 times per second as in Figure 1-1; however, the data acquisition system may poll the instrument 16 times per second, depicted in Figure 1-2, producing exactly six stale values per second. Removal of stale data is often crucial to the correct analysis of the data (frequency analysis is one such example). A simple-minded approach of removing all duplicates ignores the possibility that the instrument may genuinely sample two equal consecutive values.

Although the the instrument and data acquisition clocks may keep perfect time (do not drift), the clocks are usually free-running. That is, there is no synchronization between instrument and data acquisition system. The first polled value in the sample input (which corresponds to Figures 1-1 and 1-2) occurs with an offset delay of 1/160 (0.00625) seconds. The polling continues every 1/16 second thereafter. For this polled sequence, it is possible to reconstruct the instrument sequence uniquely.

Figure 1−1

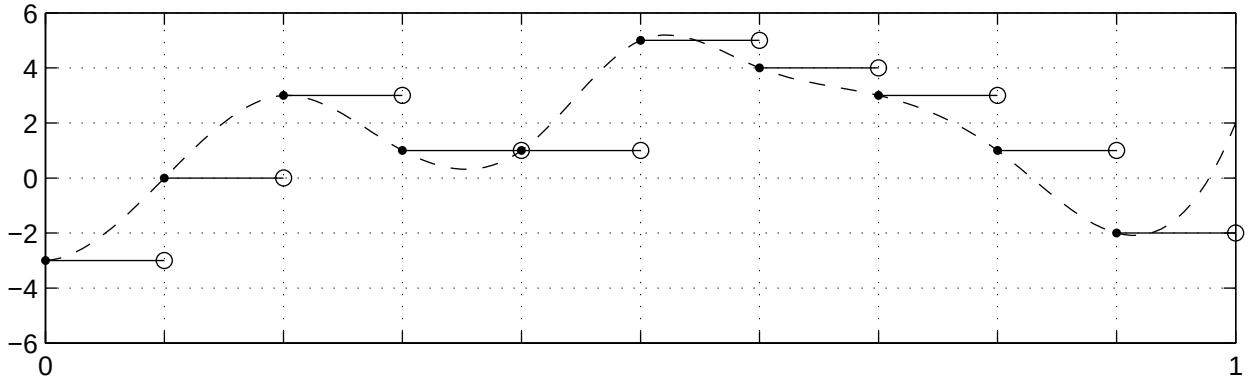
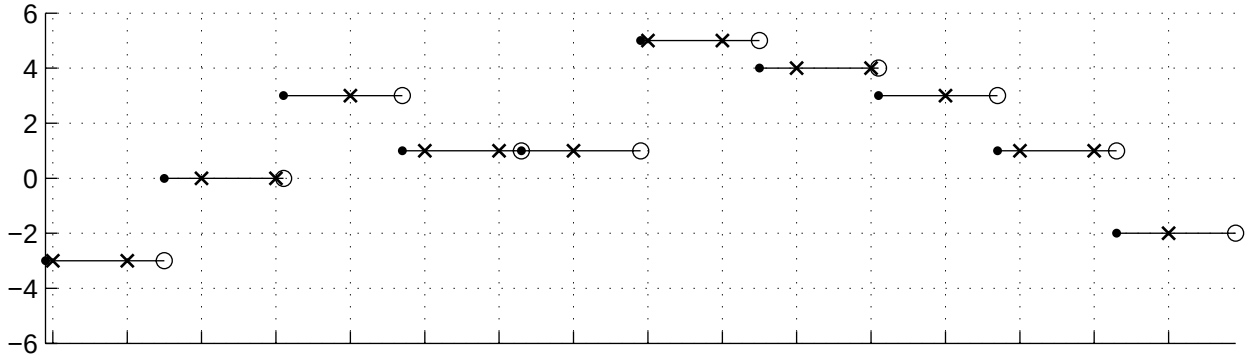


Figure 1−2



Problem 1
Stale Data Removal (continued)

Uniquely reconstructing the instrument sequence from the polled values is not always possible. Consider the 10 sample-per-second (10 sps) instrument sequence $\{0, 0, 0, 0, 0, 0, 0, 0, 1, 0\}$ that is polled 11 times per second with an offset delay of 0.09 seconds. The resulting polled values are $\{0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\}$. Unfortunately, the 10 sps instrument sequence $\{0, 0, 0, 0, 0, 0, 0, 1, 0, 0\}$ polled 11 times per second with an offset delay of 0 seconds generates the same polled values, $\{0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\}$.

Your program should take a given instrumentation sample rate, F_i, the data acquisition system polling rate, F_p, and a list of polled values. From this input, your program should either reconstruct the instrument sampled sequence, or determine that a unique reconstruction is not possible.

Input is a series of integers, one per line. The first line contains the instrument sampling rate, F_i, in samples per second. The second line contains the acquisition polling rate, F_p, again in samples per second. $F_i \leq F_p$. All remaining lines contain the acquired (polled) data sequence, one value per line, until end-of-file. The number of polled values will always be an integer multiple of F_p, with at least F_p values of acquired data. Sampled and polled values always fall within the range [-32678,32767]. You will never be asked to analyze more than 256 polled values.

Output must contain either the instrument sample stream, one value per line, or the line:
`insufficient context`
when a unique reconstruction does not exist. The files *ambiguous.in* and *ambiguous.out* are available via the *getdata* command.

Sample Input

```
10
16
-3
-3
 0
 0
 3
 1
 1
 1
 5
 5
 4
 4
 3
 1
 1
-2
```

Output for the Sample Input

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