Homework 6: CLU and Program Verification

This homework is due April 19, 1991, (by 5:00 PM).

Test your programs using the `clu` command on Unix, which is the interpreter described in Appendix F (with `load` added). Put your CLU code for the standard problems in `hw6.l`. Hand in a print-out of the code along with your answers to other problems, and please put your name on both. (If you do extra-credit problems, please write the names of the file(s) where the source code for your solutions can be found on your print-out.)

1. (15 points) Samuel Kamin remarks on the bottom of page 209 that the interpreter code is influenced by the idea of data abstraction. For this problem you will describe two (2) such data abstractions found in the interpreter of chapter 1 (appendix B) of the book. For each of the two data abstractions you find in the code, describe (i) the abstract values of the objects, (ii) a notation for a typical abstract value, (ii) and the operations on the objects, giving for each operation its name, type, and a English description of its effect, using the notations of (ii). The idea is to try to be as precise as you can in your descriptions. For examples, see the CLU lecture notes handout.

2. (extra credit only) Implement one or both of the data abstractions you described in the previous problem by a CLU cluster.

3. (10 points) Implement the `note` data type described in class as a CLU cluster. Be sure to include for each operation of the type a comment describing, in English, the effect of the operation.

4. (30 points) Implement the 4-part voice leading problem as described below in CLU. For this problem, we use the following terminology. A *chord* is (abstractly) a tuple of 4 notes (b t a s, representing the voices bass, tenor, alto, and soprano) such that for each $x \in \{s, a, t\}$, $x = b + p + k \times 7$, where $p \in \{0, 2, 4\}$ and k is an integer and $s > a > t > b$. A *note* is an integer (1 is tonic, 8 is octave above, etc.). A *bass line* is a list of notes.

   The program's input is an *initial chord* and a *base line*. The program's output is a list of chords such that: the bass note of the ith output chord is the ith note of the bass line, the first chord of the output is the initial chord of the input, and the following rules are satisfied:

   (a) Notes are repeated if possible in all voices. For example, consider the $i-1$st and ith chords of the output. The tenor (or alto or soprano) of the ith chord should repeat the same note as the tenor (or alto or soprano) of the $i-1$st chord, if possible subject to the conditions above.

   (b) Each voice moves to closest note from previous chord. For example, consider the $i-1$st and ith chords of the output. The difference between the tenor (or alto or soprano) of the ith chord and the tenor (or alto or soprano) of the $i-1$st chord should be as small as possible, subject to the conditions above. To break ties in this rule (i.e., two notes equally close that are possible), the sign of the difference should be opposite to that for the bass.

For this problem you should use your note cluster from the previous problem. You should also use a chord cluster, either the one discussed in class or one you design yourself. (It's not clear that the one discussed in class is best for this problem.) You can also use the any of the clusters in the textbook (e.g., List).

5. (extra credit only) Do problem 5 on page 267 of the text (rational numbers).

6. (extra credit only) Do one or more of problems 3, 4, or 7 on page 267.

7. (extra credit only) Read the Unix manual page for CLU and write one or more of the above problems using the "real CLU" (accessed by the command `real-clu` from your account). See Gary for help with syntactic or other problems specific to "real CLU".

8. (extra credit only) Do one or more of the problems in section 6.9.3 of the text (program verification).

9. (extra credit only) Do (one or more of) the problems 8 or 9 in section 6.9.2 of Kamin's book. You will find the source for the clu interpreter in

   `/home/cs342/public/src/chap.6/clu.p`

   Make a copy using the Unix `cp` command and then edit your copy. Turn in differences (using the Unix command `diff`) as before.