MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.5150/6.5151—Large-Scale Symbolic Systems
Spring 2025

Pset 4

Issued: 5 March 2025 Due: 14 March 2025

Reading: SDF Chapter 3, Section 3.5: Efficient user-defined types

Documentation: The MIT/GNU Scheme documentation online at
`http://www.gnu.org/software/mit-scheme/`

Note: This problem set builds on the previous problem set. Please be sure to start early and ask for help if you need guidance. The software has been augmented with many new files since the previous problem set.

To Do

The "flavor" for this problem set is `user-defined-types`, so to load the code use:
`(manage 'new 'user-defined-types)`.

To modify the entities in the adventure game you must edit stuff in the following files:

`</path/to>/sdf/user-defined-types/adventure-substrate`

`</path/to>/sdf/user-defined-types/adventure-objects`

`</path/to>/sdf/user-defined-types/adventure-world`

Problems are on SDF pages 153–154:

Exercise 3.16: Adventure Warmup

Exercise 3.17: Health

Exercise 3.18: Medical Help

New Exercise: Food for health
It would be nice for there to be a few vending machines on the campus that have "food" that can be purchased with "money". Eating food may increase one's health (both the avatar and other characters). This can decrease mortality due to troll bites. So each character has a "wallet" with an inital amount of money. There should be a mechanism where money in a wallet is incrementally increased with time, perhaps "interest" on the amount in the wallet. Perhaps money can be exchanged among characters? Invent and implement such an elaboration to the adventure game.

Exercise 3.21: Your Turn

Exercise 3.22: Multiple Players (optional)
Augmenting this game, with a video interface and supporting multiple players over a network could be the basis of a group term project. The interesting part is preventing reader/writer bugs that can occur with concurrent modifications to the database.

Another good hack would be to make a distributed game so there is no central database – each player maintains a local database describing the state of play, which is made consistent with other players in some interesting way. That way, if a player's machine fails the other players can proceed.