Northwestern University
Department of Computer Science

Programming Assignment 4

CS 211 Fundamentals of Computer Programming II
Spring Quarter 2006

Due : Monday 05/22/06 at 11:59pm

Goals of this assignment

The goals of this assignment are to learn how to use inheritance, polymorphism, and to save Roquefort the Rat from a horrible fate. *Again! Jeez what's with this rat?!*

Part 1: Roquefort the Rat and the Island of Doom

In this sequel, Roquefort the Rat makes it to an island inhabited by Comet the Cat. The island is full of dangers both for Roquefort and Comet.

As usual, the cat always tries to catch the rat (or at least get it killed). At each step, it moves towards the rat's direction. The only ways for Roquefort to survive are:

- He keeps moving until the cat gets tired and gives up the hunt. He may take advantage of a number of tunnels on the island which lead him to random locations (it is, however, possible that a tunnel gets him closer to Comet than he was before). Comet always jumps over the tunnels, as she is too big to fit in one.
- He lures the cat into one of the Curiosity Pits found on the island. If the cat falls into one, she dies. Roquefort is smart enough to never fall in a Curiosity Pit.

Roquefort's movements are controlled by the player, using the numeric keypad (1: SW, 2: S, 3: SE, etc.) Comet tries to dispatch Roquefort in any one of the following ways:

- Catch up to him.
- Force him into the sea where he will drown (seeing as how he always ends up on islands, perhaps Roquefort should invest in some swimming lessons...)

The cat always moves (stupidly) towards the rat. The movement is decided by the computer. If the cat encounters a tunnel, it jumps over it. If it encounters a Curiosity Pit, it falls in and dies. It doesn't matter whether it can swim or not since it will never move towards the sea. However, it does have limited stamina. Every time it makes a move its stamina decreases. When it becomes zero, the cat stops hunting and the game terminates with Roquefort as the winner.

The game

Provided code

The base class representing items that can be found on the island is `Contents`. There are two types of contents on the island: animate and inanimate objects. The class that represents animate objects is called `Animal` and has two subclasses, `Cat` and `Rat`. The class that represents inanimate objects is called `Thing` and has the subclasses `Tunnel`, `Sea`, `Pit`.

You will be given very basic definitions of these classes. You'll have to add functions and possibly additional data members to them.

Finally, there is the `World` class which represents the game's universe. It contains the following objects:

- A two-dimensional array of pointers to `Contents`. This is the island, and its size is predetermined (15×15)
- A pointer to the cat. This allows us immediate access to the array element that holds the cat (otherwise, we'd have to traverse the whole island to find it)
- A pointer to the rat.
- A pointer to a `Sea` object. We do not need to allocate space for each and every array slot that corresponds to the sea, since all `Sea` objects can be seen as just one.
- A pointer to a `Tunnel` and a `Pit`, for the same reason we have a pointer to the `Sea`.
- The number of tunnels and pits on the island. These are determined randomly.
- The termination state. This is an enumeration whose value represents the state of the game. The default value is `AllNormal`, meaning that the game should continue. All other values terminate the game and they represent the reason for termination (the rat has drowned or been eaten, the cat is tired or has died).

You are only given the implementations of the following functions:

- `create()` which creates the board. STUDY THIS FUNCTION to see how it's done.
- `display()` which displays the board.
- The `World` destructor.
- `endGame()` which prints out the reason the game has ended. Feel free to modify this if you like.

You are also given the prototype for the main gameplay function, but you have to implement it.

You'll need to come up with several functions of your own. For example, the `Rat` and `Cat` classes will need functions to determine the new location. The `World` class will need a function to update the board with the new locations of the rat and the cat. Be careful of memory issues and don't forget to set the terminating state as necessary.

Study the provided files and design your game before you start writing code. THIS PROJECT IS NOT EASY. Start early.

What to do

Study the given files and the provided executable (game under Linux, game.exe for Windows), to get an idea of what has been done for you and what is expected. Think about what additional data members and functions may be needed. *Then* start writing code.

Feel free to add more features to your game if you wish. If you would like to modify something (e.g. the way Comet the Cat handles tunnels), send an email to bmd with your suggestion.

What to submit

Write your name and email in the form of comments at the beginning of every file you submit. A plain text `README` file explaining your approach and changes will also be helpful.

Combine **all** of the files for this assignment into **one** tarball (compressed tar file) or zip file called `pa4.tar.gz` or `pa4.zip`. Please keep all of the Roquefort files in their own subdirectory.

Start your work early and take advantage of the discussion board and office hours.