Exercise 6: OOP

N. Berger (nberger@physi.uni-heidelberg.de)

7.11.2011

Please send your solutions to nberger@physi.uni-heidelberg.de until 21. 11. 2011, 12:00. Put your answers in an email (subject line *SMIPP:Exercise06*). Test your program before sending it off...

1. **Gift exchange** Write code to play a very simple game, used in social sciences to detect altruism. Player A is given an amount of money M (e.g. 10 EUR), he can then give some part of the money $x \leq M$ to player B, who will receive a sum of kx, with $k > 1$ (we will use $k = 1.7$). Player B then gets to send some amount of money y, $y \leq kx$ back to A. There are various strategies for A and B to determine x and y which are optimal if the game is played just one way, both ways, once or many times.

   The idea is that you write a class that plays this game (against itself and eventually against the classes of the other students). This class has to implement the interface specified in the `BasePlayer` abstract base class available from the course website. So write your own class inheriting from `BasePlayer` and implementing all the virtual methods (and any additional methods you might need). Give the class an unique name somehow referring to your name. Compile it (a sample makefile is available on the course website, just fill in your class there). Get rid of all error and warning messages from the compiler. Test it using a `main()` program (template (exercise6.cpp) also available on the website). Execution is via `./exercise6`. Then send in the code files.

   All the classes received in time will play against all others in four modes:

   - Single game round, once as A and once as B.
   - Back and forth game round, once starting as A, once starting as B.
   - 100 rounds back and forth, once starting as A, once starting as B.
   - 10000 rounds back and forth, once starting as A, once starting as B.

   Classes will be ranked for each mode, the class with the lowest ranking points wins. The classes will not be told in which mode they are going to play. Classes have to keep track of the money they earned so far (which will be checked). The k factor is handled by the extarnal program, but will always be 1.7.
   (Attach one or more .cpp and .h files)