Pac-Man

Joseph Cordina

January 2, 2011

This is the description for Assignment 1 of unit CSA 2205, System Programming, for the year 2010/2011. This assignment is worth 13 of the total mark (30) for this part of the unit. The deadline for this project is 8th May 2011. You may do this assignment in a group of maximum 2 people, yet you may do this assignment individually also. A single mark will be allocated to the whole group, so choose your partner carefully and make sure that the work gets evenly distributed. Under no circumstances should code be shared outside your group. You will submit your assignment making use of the ASS system[1]. Those of you without login access, please contact me on joseph.cordina at um.edu.mt in due time (thus make sure you check you can submit at least two weeks before the deadline). Each team might be asked to present their implementation of your assignment during which your program will be executed and the design explained. Please be reminded that you cannot copy and plagiarise to ease your way to a final submission. While you may discuss ideas with others, do not steal. You can find more information at the University main website.

Read th plagiarism information well since after-wards no excuses will be accepted. Anyone resorting to such methods will be considered as trying to cheat and will even risk the removal from the degree program.

1 Submission Contents

You will submit your C files together with any accompanying header files (C language has to be used obviously) separately. Please do not use any tar or zip file formats. Additionally include any libraries you might have used. The compilation can assume the presence of 'gcc' on which your system will be compiled. Your code has to work on kronos.cs.um.edu.mt and poseidon.cs.um.edu.mt. If your code works only on a Linux machine, you will be penalised (and you have to explicitly mention that it only works on Linux in your report). Include with your submission a *makefile* that can compile your system[2]. You will need a makefile for the client code. If you need to have a server setup, also include a makefile to build the server. With your submission, you will provide a file called report.txt which will be a very **brief** description of how your system works and provides the required functionality. Make sure you mention any special techniques or tools you made use of and also any considerations you have made to improve efficiency. Additionally, please note down any bugs your system might have. Listing bugs is common practise, and will benefit the marking process. This file should also contain references to resources you have made use of in your program. Finally, you will submit a file 'architecture.jpg' containing an image that describes in the most concise and clear manner possible your system's architecture (including any important communication paths and structures used).

Before submitting, please do clean up your code. Remove any dead code and useless comments. Every system calls output should be validated for errors and when appropriate, an error message reported to standard output. Also do place some comments to

[1] www.cs.um.edu.mt/~jcord/ass

[2] You can find plenty of tutorials on the net that teach you how to set one up

describe your structures in the code, at least to show that you understand how your own program works. Remember C is only unreadable if you make it unreadable. I cannot emphasise enough to be careful with your naming conventions and visual formatting.

2 Pac-Man

The aim of your assignment is to build a two-player Pac-Man game. It should be pretty easy to find out how to play this game[3] or you can can find many online versions of this game[4]. You may use a static map for the game, i.e. one that never changes between games. Every user that runs the command *pac-man* will be placed in the game (note that each instance can be run by a different Unix user). If a game is already present with one player, then the second user will be placed as the second player in the game. Each player will start in a position that is not occupied by any other player or ghost. Subsequent users running this command will be placed in a *view-only* mode until the current game finishes, after which the user gets to play (in a FIFO fashion). Each user gets three lives and the winner in a two player game is the one that has eaten the most *dots*. For your implementation, I suggest you make use of the *ncurses* library, which allows fine control over a text terminal. Every half a second, the ghosts and the pac-mans will move one step (feel free to tweak the speed to make game playable). You can change the direction of movement by using the keys w,a,d and x.

The players might be joining the game as a different user in the system, but you can assume that all players will be running on the same system. Each player will see a current true picture of the map drawn on his terminal. There should be no limit of the number of viewing players allowed.

3 What might matter

Most of the marks will be allocated to a good design, resulting in a solution free of race conditions, deadlocks and still considerably efficient. For your own good, first design your system before starting the coding. It will result in a better system and a quicker turn around time. For this assignment you cannot make use of the socket API, but you may use any other IPC structure. The processes involved can be assumed to run on the same system, and will be run under the same or different users (having non-root privileges). You cannot assume that the communicating processes are related to parent-child relationships. You can also assume that no deadline extensions will be given. I will compile your code using your provided make files and then play against your implementation a standardised test run which will dictate your marks.

4 And To Conclude

Feel free to make use of any optimisations you can think of. Optimisations you should consider are implementation optimisations (better data structures, less indirect references, etc) and also algorithmic optimisations. Keep the allocated time you spend on this project in perspective with the allocated marks. As a suggestion, first sit down and design your solution well before implementing it since this will aid you a lot in your programming.

Good luck and if you need anything, contact me on joseph dot cordina at um.edu.mt. Even better, make use of the discussion group to discuss ideas and ask questions. You can find a link to the discussion group on my website. Other students might help you out in the solution. Just make sure you do not post any code or given concrete solutions.

[3] http://en.wikipedia.org/wiki/Pac-Man
[4] http://www.google.com/pacman/

Note that you will be doing a lot of design work and scrapping lots of ideas, so do not expect to do this assignment in the last few days before the deadline. Do it slowly and the ideas will mature. Remember this is mainly a design assignment (like all good system tools) and you only do the coding after you are sure your design is correct.

I might post on the website a list of designs that impressed me (lets hope there are a couple).