P6

Latin Squares on the Raspberry Pi

At the end of this laboratory, you should have had a lot of practice implementing simple C++ classes and manipulating pointers.

You should have implemented a fully working Latin square solver on the Raspberry Pi implemented as a console application that accepts a number on the command line and outputs all distinct Latin squares of that size, using Knuth's *Dancing Links* algorithm.

There is a very helpful presentation from Rob Beezer on this topic.

Do not let yourself get stuck trying to understand the ideas here; ask for help in the lab.

Preparation time	:	3 hours
Lab time	:	3 hours

Items provided

Tools	:	None
Components	:	None
Equipment	:	DVI-D capable monitor
Software	:	Win32 disk imager

Items to bring

- Essentials. A full list is available on the Laboratory website at https://secure.ecs.soton.ac.uk/notes/ellabs/databook/essentials/
- Raspberry Pi boot image
- Raspberry Pi
- Raspberry Pi power supply
- Raspberry Pi SD card
- Raspberry Pi HDMI to DVI-D cable
- Raspberry Pi keyboard
- Raspberry Pi mouse

Before you come to the lab, it is essential that you read through this document and complete ***all*** of the preparation work in section 2. If possible, prepare for the lab with your usual lab partner. Only preparation which is recorded in your laboratory logbook will contribute towards your mark for this exercise. There is no objection to several students working together on preparation, as long as all understand the results of that work. Before starting your preparation, read through all sections of these notes so that you are fully aware of what you will have to do in the lab.

Academic Integrity – *If you undertake the preparation jointly with other students, it is important that you acknowledge this fact in your logbook. Similarly, you may want to use sources from the internet or books to help answer some of the questions. Again, record any sources in your logbook.*

Aims, Learning Outcomes and Outline

This laboratory exercise aims to:

- Develop confidence in the use of C++ pointers.
- Develop the ability to convert a formal algorithm into a working program.

Having successfully completed the lab, you will be able to:

- Develop C++ programs on the *Raspberry Pi*.
- Implement and use a significant computational algorithm in a working C++ program.
- Make good use of classes to structure a pointer-based algorithm.
- Write effective test software to verify your output.

To prepare this lab you will need to familiarise yourself with the *Dancing Links* algorithm for Sudoku, using Knuth's original paper and other internet sources.

1 Preparation

Find out about Latin Squares and the Dancing Links algorithm; the algorithm was described in the last lecture on 7th March. In particular, by using web and other sources:

- Ensure you clearly understand how the *Dancing Links* algorithm works.

- Design appropriate C++ classes for the algorithm implementation.

Record what you have learnt in you logbook.

---

2 Laboratory Work

2.1 Subsection 1

On your Raspberry Pi, build a console program which accepts a single number between one and nine as input on the command line using **`argc`** and **`argv`**) and uses the *Dancing* Links algorithm to output to the console a complete set of Latin squares of that size, in the format (here with an input of **`3`**):

```
123
231
312

231
123
312
```

...several more squares

Note that most of the credit for this laboratory is for this section; it is particularly important that you

1. correctly implement the generic *Dancing Links* algorithm for perfect match problems and then construct appropriate initialisations for the *Latin Square* problems,
2. make effective use of C++ classes, and
3. use pointers correctly.

2.2 Subsection 2

Write a separate program which checks the correctness of your output. In particular, ensure that every square has the correct number of rows and columns, contains only the correct digits, and contains no repeated digits.

---

3 Optional Additional Work

Marks will only be awarded for this section if you have already completed all of Section 3 to an excellent standard and with excellent understanding.

1. Modify your program to print only Latin Squares in which the digits appear in ascending order in the first row and the first column.

2. Many such ordered *Latin Squares* form the multiplication table of a mathematical *group*. What is the smallest such square that does not?

3. What is the smallest *Latin Square* that forms the multiplication table of a *non-abelian* group?

Revision History

March 11 2014	Denis Nicole (dan)	2014 version of this lab created