## 1B Advanced Physics: Computing in C++

### Summary

This course will consist of 6 lectures and 6 practical classes. The best way to learn computing is by doing it, so much of the material will be taught through self-guided study; lectures will focus on concepts that are suited to the lecture format. The introductory course will teach only the 'C' subset of C++ ('but it better motivates the students when we called it C++' (Roberto Cipolla, CUED)). Classes and object-oriented programming would be introduced in the part II course. The course will be in part modelled on CUED's C++ course successfully used by all first-year undergraduates (300 per year) since 1997.

http://mi.eng.cam.ac.uk/∼cipolla/undergrad.html

The general structure of the course will be:

- Three lectures (one per week) in the first three weeks of the Michaelmas term. There would be three weeks of practical classes (weeks 2-4) where the students would work through the self-study guide along the lines of that currently used in the part II FORTRAN course. The aim of the first half of the course is for every student to become familiar with linux, gnuplot, a text editor, elementary C++ programming, and a C++ debugger. Each practical session would have a specific programming task related to the previous lecture.

- Three lectures (one per week) in the first three weeks of Lent term and three weeks of practical classes in the PWF. In the second half of the course, C++ programming concepts will be covered with the help of examples drawn from the 1A and 1B Physics courses. The students will each complete *three* mini-projects, each requiring two−three hours of work. The students would be given a program framework for each of the mini-projects (main, relevant includes, empty functions etc.). Each mini-project will consist of a core task which all students will have to complete and optional parts introducing more interesting computational/physics ideas.

### Assessment

Assessment of the course will use a simple system aimed at ensuring that steady progress is being made every week. All marking will be pass/fail, and the aim will be for every student to achieve full marks. The total mark would be incorporated into the Advanced Physics practical mark. For each task the students will be required to upload their program files using a web interface. The students could also be required to answer one or two simple questions using the web interface related to the task e.g. how did you test the program? The questions would be designed to make the students think carefully about certain aspects of the mini-project. Full marks would be awarded on submission of each task/mini-project. The students would be told this and that the examiners could take the work into account for students at class boundaries.

**Course Synopsis** *(draft 1.2)*

The synopsis below indicates the general structure of the course and the style and level of the set tasks/mini-projects.

---

**Week 1.** (First Thursday of Michaelmas term):

**Computing concepts** How numbers are represented in the computer. What a `C++` program looks like.

**Computing skills** Using linux, bash, text editor, `C++` compiler, and `C++` debugger.

**Self-assessed task** Explore a working program with a `C++` debugger. Modify the program and recompile.

**Week 2.** (Second week of Michaelmas term):

**Computing concepts** Functions, loops. Monte Carlo methods.

**Self-assessed task** Write a program to measure $\pi$ by Monte Carlo sampling from the unit square.

**Week 3.** (Third week of Michaelmas term):

**Computing concepts** Pointers, conditionals.

**Computing skills** Using gnuplot.

**Self-assessed task** Write a program to generate some random walks and print them to file. Plot the random walks using gnuplot.

---

**Week 4.** (First week of Lent term):

**Computing concepts** Functions, memory allocation.

**Mini-project** Write a function to compute the result of a collision between two particles of masses $m_1$ and $m_2$ and velocities $v_1$ and $v_2$.

**Week 5.** (Fourth week of Lent term):

**Mini-project** Use the collision function from week 4 to simulate the motion of 20 different-massed particles in a one-dimensional box. Spit out the state of the system every $\delta t$ seconds. **Requirements:** function to decide what event comes next (a collision, or a clock tick); a working collision function. Show that the average velocity of a particle is inversely proportional to $\sqrt{m}$.

**Optional Part** Show that the particles have a Gaussian momentum distribution and that particle positions and momenta are independent.

**Payoff** Successful completion of this project yields startling results: a) particles' momenta have Gaussian distributions; b) more massive particles move slower ($\sqrt{m}!$) c) particles' locations and momenta are independent.

**Week 6.** (Second week of Lent term):

**Computing concepts** Likelihood functions

**Mini-project** Write a program to read in some data and perform a straight-line fit using likelihood maximization (also known as $\chi^2$ minimisation).

**Optional Part** Make some random Poisson distributed data from a simple distribution (in `C++`), then fit a model to the data by maximising the likelihood (in gnuplot).

**Payoff** reinforces the ideas in the Experimental Methods course. The optional part introduces the idea of likelihood functions other the $\chi^2$.

---

---

*David J.C. MacKay + Mark Thomson  January 10, 2007*