AN ADAPTABLE COURSE OF ELEMENTARY, UNIVERSITY LEVEL, COMPUTER SCIENCE

P. Naur

Rapporteurs: Mrs. M. Hindmarsh
Mr. A. Hunter
Mr. B. Jones

1. Introduction

Professor Naur has been a member of a working group at the University of Copenhagen which has worked out a course plan to be used in constructing courses for specialists in fields other than computing. His first lecture gave an over-all view of this plan.

The working group consisted of six or eight members of several different faculties and included a biologist, a psychologist and an economist as well as a computing scientist - Professor Naur himself. The basic question to be answered by the group was whether to define a course to be taken by all students, regardless of their own particular interests and studies, or whether to decentralise and arrange different courses to accompany different subjects. In past years first-year students at Copenhagen took a compulsory course in philosophy. This was abolished a few years ago; should it be replaced by computing? It was finally decided that this was not advisable; teaching adjusted to the students' special interests was devised, but teaching that was, to some extent, unified. Professor Naur described the course as a frame with open holes each of which could be filled with illustrations and exercises from the students' own subject.

2. An outline of the course

The course consists of a basic part of 30 lectures with associated homework and guided laboratory work of about 15 hours, designed to last one semester. In addition a number of optional, supplementary subjects are specified.

The course stresses three things:

1. Data handling.
2. Data representation in the computer.
3. Simple programming (it was found by the economists that their students had difficulty in following the course without doing some elementary programming).

Details of the course are given in [1].

3. The basic course

This is described in the second chapter of [1], which starts with a set of 'goals' of the course, described by a number of skills that the successful students would have after having followed the course, as given below. The goals fall into four parts.

I Terminology and concepts.

Terms are chosen from standarised vocabularies. One very important course goal is to establish a common language of computer terms.

1. Hardware.
   There is no insistence on definitions but the student is expected to recognise, say, a card-reader shown in a picture.
2. Software and computer operations.
   The student should have a general idea of, for example, a compiler.
3. Data structures and their organisation.
   A student should be able to describe certain data structures and know how they relate to the physical structure of the computer and the media.
4. Programming.
   No specific language is insisted upon: Algol 60 and Fortran are used but only as examples. Terms for the elements of the programming language chosen - statements, variables,

do loops, and so on - are taught.

When given a term for such an element the student then should be able to

(a) write examples of such elements,
(b) given a program text, find examples of the element,
(c) describe how they work in the given program,
(d) give further examples of how these elements can be useful.

II Programming skill.

1. Program construction.
   The student should be able to transcribe data processes into a program section and describe briefly the variables in it. The section should be sufficiently accurate so that it would work if trivial errors were ignored, errors that could be corrected by a programmer who does not understand the problem.
2. Explaining a given program.
   The student should be able to
   (a) give a general explanation of a short, well-documented program,
   (b) determine how many times each statement is executed,
   (c) give a step by step description of the execution of the program.
3. Program testing.
   Given a simple program, the student should be able to point out obvious syntactic errors and construct a set of input data to activate all parts of it.

III Data media.

1. Quantity of data.
   Given a real life situation, preferably within his own field, the student should be able to estimate

(a) the amount of data to be handled, expressed, for example, in bits or characters,

(b) the time and cost involved in a manual transcription of the data to a machine-readable form,

(c) the likely numbers of errors in simple transcriptions.

2. Constructing the format.

The student should be able to suggest a format for representing the real life situation in a given medium.

3. Data processing.

Given a particular problem, the student should be able to suggest, in general terms, methods of solution. Knowledge of detailed sorting methods is not assumed.

IV An understanding of the potentialities and limitations of electronic data processing.

The student is asked to write an essay discussing the arguments for and against using electronic data processing for solving a particular problem.

4. Discussion

Professor Hamming, believing that there is no substitute for experience, considered it preferable to eliminate the minutiae of today and concentrate on those things that will still be relevant in 30 years time. There followed a lively discussion, with contributions from Professor Ercoli and Professor Naur, about the probable half life of the usefulness of university courses. Differences remained unresolved.

Professor Michaelson considered there to be no conflict between practical experience and future values, and suggested that the main difficulty lies in the non-uniformity of students, their abilities and interests. Professor Naur pointed out that this was precisely why the Copenhagen courses were not centralised.

5. A detailed course structure

Professor Naur, in his second lecture, expanded on his previous theme by giving a detailed breakdown of a course of 30 one-hour lectures, which could be tailored to the needs of a faculty giving it. He then suggested a set of supplementary lectures, a subset of which could be appended to the basic course as required according to the interests of students or staff.

5.1 Plan of work for the Basic Course

The first 13 lectures present a basic, formal introduction to computing. After this point however, the course should include comprehensive examples related to the field of the students. This was vital both to their understanding of the course content, and to maintain their interest.

The breakdown suggested in the course plan [1] is as follows:-

Lecture	Content
1. } 2. }	A simple example of a program; flow charting; punched card equipment; the practicalities of job submission.
3.	Character sets; basic symbols; keywords; literals; names; the concept of a variable.
4.	Error diagnostics; program translation by compilers; source and object listing; how to correct errors.
5.	Assignment statements; declarations; simple and subscripted variables.
6.	Simple arithmetic expression. General rules of program construction using them.
7. } 8. }	Calls of built-in subprograms and simple input/output.
9. } 10. }	Repeat and condition clauses; jumps; labels; Boolean expressions.

11,12 } Subprograms; large structures of programs;
13. } a summary of lectures 1 - 10.

During lectures 14 to 25 there is a parallel discussion of a comprehensive example of an application related to the students' field of study. Subjects for the discussion include: design, division into subprograms and detailed programming of selected parts.

14. } Testing; reliability; documentation of programs;
15. } a crude estimation of computer time.

16. } Programming of transfers to and from ancillary stores;
17. } further input/output.

18. } Elements of computer hardware; peripheral units;
19. } the configuration available.

20. } Organising of auxilliary storage devices;
21. } the operating system.

22. } Data; data structures and organisation;
23. } (files, records, ... etc.).

24. } Data transmission; media; capacity and
25. } access times of various media.

26. } The general problem of handling large amounts of
27. } data; specific example; a large system for handling a stream of transactions.

28. } A survey of problem oriented programming languages;
29. } potentialities and limitations of electronic data processing.

30. Questions; student criticism, possibly a quiz.

5.2. Supplementary Lectures

To make the course more flexible, the course plan given in [1] suggests that lecturers may wish to extend the course by adding one or more sessions from the following list of 27.

1. Number representations.
2. Logical and bit operations.
3. Representation and handling of alphabetic strings.
4. Numerical methods.
5. Monte Carlo methods; random numbers.
6. Supplementary programming languages.
7. Style and documentation.
8. Syntax and semantics.
9. Problem oriented programming languages.
10. Graphical output.
11. Data organisation, lists.
12. Searching and sorting.
13. Large files, databases.
14. Checking and redundancy.
15. Program libraries and collections of algorithms.
16. The computer - hardware details.
17. Process control, real time situations.
18. Interactive operation.
19. Data transmission and networks.
20. Analogue computers.
21. Selection of computers for an organisation or project.
22. Non-electronic data processing (card systems, etc.).
23. System design.
24. Operating systems and job control languages.
25. Design and programming as a team activity.
26. Legislation and software.
27. Data storage and processing in society.

Professor Naur indicated that the above list was intended mainly for guidance. It could be used positively to find subjects to include, or negatively to decide what to leave out.

6. Summary

As examples of successful use of this course at Copenhagen University, Professor Naur cited the Sociology department who had adjusted their course work to include it. In the case of the Medical Faculty, it had been hoped to introduce the course, but pressure of other courses made it impossible. Chemistry, Physics etc., had, of course, been using computers long before the course was developed and did not need to be "sold" the idea. Of non-science oriented departments, Sociology and Linguistics were the most active. Some reasonable syntactic analysis of French texts had been done on the computer to analyse sentence structure.

Organisation of the course was done by the computing department staff. Certain members of staff had specific responsibility to keep it alive and update it as necessary. A system of feedback from the course lecturers was maintained, and requests for detailed supplementary course material were also dealt with.

7. Discussion

Professor Rogers started the discussion by commenting on the difficulties of including computing courses in Medical Faculties, caused by pressure of time, and in some cases by prejudice against non-medical lecturers. He then raised the problem of course stagnation if the lectures were given by non-computer scientists. Professor Naur agreed with this, but said that it could be avoided by keeping in close touch with the faculties concerned.

Professor Randell suggested that it was perhaps too early to ask detailed questions on experience, but asked if the course terminated at a stable point such that a meaningful amount of information had been presented. Professor Naur answered that some students would forget all about computing in a very short time; others, however, would be stimulated to further work and become quite expert. Professor Randell commented that the lecture approach was

very different to Pat Goldberg's research. Professor Naur said that whichever the audience preferred depended on whether they believed in the concept of changing the computer interface to fit the user. At this point Professor Page commented that surely this was a case of what could be done now in the way of education, compared to possible future solutions. He went on to say that the feeling at Newcastle was that we preferred to have computing staff giving the courses. This would be difficult in some cases - for instance, no member of his staff was qualified to give the "Legislation and Software" lecture mentioned.

Professor Hamming asked why we didn't teach mathematics to non-specialists in this way, and suggested that it had been tried and found unsatisfactory. In reply, Professor Naur described current standards of mathematics teaching as 'frightful', and said he was certainly in favour of mathematics courses taking the form he had suggested for computing.

Dr. Lavington was concerned that professionalism should be passed on to students. He went on to point out the problems of finding out exactly what other departments required of the computer. It was, for instance, in the case of a physcist, difficult to separate the physics from the computing in a program. Professor Naur mentioned that work had been done on this during a three-year project at the University of Michigan with engineering lecturers from other centres invited to provide problems for a summer school. After finally commenting that extracting examples was never simple, the discussion ended.

8. Introduction to third lecture

Professor Naur's interest in the social problems raised by computers began many years ago as evidenced by his book, 'Computing Society', written in Danish, and published some eight years ago, the book being based on a series of lectures he had been invited to give on Danish radio.

Professor Naur described what has been happening within Copenhagen University during the past few years with student unrest having a profound effect on the organisation of the University. The students have a strong say on what will be taught and have votes on all the committees.

9. Computers and the Norwegian Trade Unions.

A few years ago, Kristen Nygaard, from the Norwegian Computing Centre - one of the authors of Simula and a politically active Social Democrat - became a consultant to the Norwegian Trade Unions and discussed with them the implications of the use of computers. In the long run the unions must find out for themselves, but he was there to offer assistance. Working groups looked into the consequence of using computers and came to several interesting conclusions. For instance one company had installed a centralised communicating system with terminals in many parts of the building, terminals through which information could be spread rapidly to the workers. This type of information spreading had previously been done by the workers' representatives who had chatted freely to the employees. This informal contract was lost.

The results of discussions with the working groups were

1. Teaching material was made available for the workers.
2. Collective agreements were drawn up between unions and management about prior consultation over the introduction of data processing equipment. This became a basic charter of rights of the workers to be informed.

10. Student involvement

Students in Denmark became interested in the social issues raised by computers in society and Nygaard visited the country for a lengthy period lecturing on these issues and discussing with the students how to move from vague ideas to specific projects. Nygaard

and some students became engaged with the Danish unions. These students supported the work of Nygaard and his end of the political spectrum.

Last January a seminar was organised at Aarhus University at which unions and the universities took part - employees declined the invitation to participate. Professor Naur took part with some reluctance, feeling that the whole issue was too politically motivated. He dislikes this type of activity in which the university is dragged into unpaid consultancy for one side of society only (the unions).

11. Political implications of teaching and research

Professor Naur believes that his job in teaching and research can be divorced from his political activities as a private citizen. Others do not agree. This difficulty may be lethal to the universities in the long run: politicians in Denamrk are unwise to allow the universities to become involved in active political work. The government has now cut down its grant to support university teaching by non-degree teachers (that is, student councillors), having now realised that although the universities are paid by the government, those universities are taking a very one-sided view of society.

These problems are impossible to avoid.

12. Discussion

Professor Rogers was surprised by the position taken by Government against University, and asked if there was a body like the British University Grants Committee acting as a buffer between University and Government. Professor Naur said there was no equivalent body in Denmark. Money was directly allocated by the Ministry of Education to individual Universities.

Professor Rogers asked if any other comparable organisation (to University) paid by the government was being pushed to take up political activity. Professor Naur replied No, Post Office and Railways are paid by government but do not take part in any political activity.

Professor Michaelson drew attention to Professor Naur's implicit assumption that all his previous activity had not been political, and he maintained that teaching computing, in itself, is a political activity, pointing out the desirability of teaching students to question the social order. Professor Naur replied that taking a teaching job makes him an employee of the government; one should try to be unbiassed but should be aware of the implications of the job.

Professor Hamming spoke of his experience at the Bell Laboratories starting some 20 years ago when he introduced computers which meant changing the structure of the organisation. There was a lot of staff unrest; they wanted to be involved with the implementation but soon lost interest. Now the management is back in control, all of this done without any political involvement.

Professor Griffiths and Professor Gillies both stressed the importance of proper criticism in research and further comments were made by Professors Ercoli, Hamming and Capriz, the last pointing out that the teacher who has no effect on society is ineffective as a teacher.

In response to Dr. Lavington's query as to the outcome of the Aarhus seminar, Professor Naur replied that there had been no direct results but he felt that it was not so much the unions who needed help in a computer-oriented society as the private citizen, who tended to be forgotten in the design of computerised systems.

Reference

[1] Authors: Andersen, J.D., Hilden, J., Leünbach, G., Naur, P., Rasmussen, J.B., Warnich-Hansen, T., Winkel, P.

Title: "Retningslinier for Universitetskurser i DATALOGI SOM HJAELPEFAG".

Subtitle: "Vejledning for Laerere ved tilrettelaegning af fagorienterbar undervisning".

Published: Datalogisk Institut, Kobenhavns Universitet, 1974.