1. ABSTRACT

The growing number of architecture and design students that take introductory computing courses justify the development of courses that are tuned to the specific needs of these disciplines. The importance of graphics has to be reflected in these courses and relationships that exist between structured programming and deterministic design problems must be demonstrated. This paper describes such a course - the software and the tutorial developed for it. It is both the introduction for architecture and design students to become competent program users and the foundation and prerequisite for more advanced courses in data structures and Artificial Intelligence for architectural tool building.

2. RATIONALE AND PAST EXPERIENCE

Mandatory computer courses have become part of the curriculum of architecture and design schools in recent years. Carnegie-Mellon University was one of the earliest institutions to require computing literacy from its graduating architecture students. Experiences through the past years show that the first contact with a programming course strongly influences the student's attitude towards architectural computing in later years.

For years, architecture and design students took the introductory computer course in the Computer Science department. The course concentrated on the Pascal language. It soon became clear that the course had to be modified somehow for architecture and design students. Thus began an interesting process which led to the Pascal course and software product described in this paper. The following assumptions were the basis for the development:

1. The learning of a programming language is a necessary part of an architect's education. This applies to future program developers as well as for program users. For software developers, the first course provides a foundation for advanced courses. For program users, the first course is a necessary step to understand and effectively use the widely
accepted business and architectural programs which all require some programming skills (for example: Lotus 1-2-3 and dBase III, and AUTOCAD).

2. Architecture and design contain a number of deterministic problems for which exist quantitative solutions. For all these, a structured programming language like Pascal provides a solution tool that can help the architect to understand the problem (by programming the solution) and speed up the execution. This applies for the numerical and graphical architectural problem (e.g., energy calculations and proportional or compositional problems).

3. At the present time, a structured programming language like Pascal is the appropriate introductory language. In the future, object oriented and artificial intelligence languages might become appropriate.

Based on these assumptions, the decision was made to teach Pascal in a form attractive to architecture and design students without giving up the teaching of basic programming principles. It was found that architecture students are willing to take on more complex programming tasks if they can see them related to architectural problems. The course developed at UCLA by Robin Ligget and Bill Mitchell [3] was an important source in this respect.

3. THE COMPUTING ENVIRONMENT

The previous introduction to Computing course at Carnegie-Mellon University was taught on a VAX II/780 based time sharing environment using Concept 100 terminals. Unfortunately, the terminals had both limited access times and graphics capabilities. To overcome some of these problems it was decided to switch to a Personal Computer Environment. The software was developed using the Turbo Pascal compiler and text editor on the IBM PC, XT, or AT. The reasons being the affordability of Turbo Pascal, as well as its graphic and sound capabilities. Shortcomings such as non-standard Pascal syntax (especially in the file input and output parts) had to be accepted.

The course can also be taught in a Macintosh Environment. For this purpose, the Turbo Pascal files are edited to run on a Macintosh based ALOE (A Language Oriented Editor) a system developed at Carnegie-Mellon University for the TOPS10 environment.

The decision to move from a time sharing system to a Personal Computer Environment increases the amount of control on the student side. With the growing number of students owning Personal Computers this is a desirable effect.

4. THE SOFTWARE PACKAGE

4.1 Characteristics

The Computer Science department at Carnegie-Mellon University had developed a sophisticated course framework including labs, programming exercises, and mastery exams. This framework was accepted in principle for the new course. The main task in the development of the new course was to
relate the principles of programming to architectural problem solving. The following describes examples of this effort:

4.2 Twelve Themes in Pascal and Architecture

The student has three sources of information and help during the course:
1. the instructor and the tutors
2. the textbooks [1,2]
3. a self-paced tutorial [4].

The self paced tutorial contains the description of each of the 12 labs, a set of programming exercises with descriptions, and a set of six mastery exams with descriptions. Each lab has a theme and is supported by two or three sample programs. Each theme is an important aspect of the Pascal language. A related problem in Architecture or design is added as an example.

Several applications can be chosen for the course - one being the construction of Palladio's Villa Rotonda with Pascal. This will be the main example described in the following twelve paragraphs.

Theme 1: Terminal Output

The student starts at "square one". After the login, one key stroke starts a Pascal program that draws a square - part of the Villa Rotonda's floor plan. The student then studies the program that produces this output and finds the information regarding how the program is composed, edited, compiled, and executed in the tutorial on the same diskette. The theme 1 tutorial also contains terminal specific information depending on the type of personal computer used. See figure 1.

Figure 1

PALLADIO

LA ROTONDA

121
Theme 2: Terminal Input

The student's role changes from passive to active. The square is explained as a special case of a rectangle. By multiplying length and width of a rectangle with given integers and by supplying the coordinates of its origin, the student adds to the floor plan. The effect of scaling factors is explained visually. See figure 2.

![Figure 2](image)

Figure 2

Theme 3: Procedures

Procedures are used to keep the program from growing too large and to give the student more control. The analogy is used in constructing different parts of the floorplan and adding them in a hierarchical order. See figure 3. The power of recursion is demonstrated for floor pattern design in the villa (not shown).

![Figure 3](image)

Figure 3
Theme 4: Functions

The difference between functions and procedures is shown. Pascal standard functions are used in combination with user defined functions. Area and volume calculations and circle drawing procedures using functions are presented. See figure 4.

![Figure 4](image)

Theme 5: Repetition

Repetition, one of the most used operations in Architecture, is introduced. FOR, WHILE, and REPEAT UNTIL structures are explored and used to draw the columns of the villa. See figure 5. Nested loops and their implications are shown. In addition, the x and y coordinates of the design are used to produce sounds by assigning them to frequencies, taking advantage of the sound capabilities of both the Macintosh and the IBM PC/AT.

![Figure 5](image)
Theme 6: Conditionals

Repetition and the adding of procedures make up only the first primitive design steps. They have to be controlled by external or internal restrictions. Conditionals are used as an appropriate Pascal structure. Sound is added as an aural signal for conditions met or not met. Stairs are added to the plan. See figure 6.

![Figure 6](image)

Theme 7: Call by Value and Call by Reference Parameters

The issue of shape and instances is introduced. Primitive shapes can be transformed into infinite numbers of instances by using parameters. The floorplan of the villa with its numerous circular and rectangular shapes is an excellent application. See figure 7.

![Figure 7](image)
Theme 8: Arrays

Arrays can be used as storage and organizational structures. One dimensional and multidimensional arrays and their applications are shown. The coordinates of the grid system of the Villa Rotonda are used as an example. Using arrays for other computations is explained. See figure 8.

Figure 8

Theme 9: Records

Advantages of the record structure over the array structure for certain architectural applications are explained. Initializing records and using arrays of records for storing coordinates and attributes of building parts are used as an example for the villa. See figure 9.

Figure 9
Theme 10: File Input and Output

So far, everything drawn on the screen was lost after the student logged off. File input and output introduces the notion of permanent storage and retrieval of computed data. The student experiences time advantages in retrieving computed data from storage over recomputing data. Initializing and updating of external files is practiced. See figure 10.

Figure 10

Theme 11: Dynamic Data: Pointers and Linked Lists

The advantage of dynamic over static variables for graphic programs is shown. Data structures that vary in form and size during execution are presented. The floorplan of the villa is used to point to certain elements (walls, rooms, etc.) with specified attributes. See figure 11.

Figure 11
Theme 12: Searching and Sorting

The importance of searching techniques for different applications including AI and Robotics is explained. Searching the solution space is a part of every problem solving process. Sequential, binary, and hash search techniques are introduced. For the villa application, a sequential search according to a certain search key is executed on a number of elements stored as records. See figure 12.

Figure 12

5. CONCLUSION

The above "twelve themes" present one option for the course. Another more numerically oriented option is available. See figure 13 as an example for an exercise using procedures, functions, repetition, and call by value parameter in the IBM PC version of the course.

Figure 13
The main concern in designing the course was transparency so that the source code of all programs, including the tutorial itself, can be studied by the student. A number of utilities - such as changing foreground and background colors for the Turbo Pascal version, or text windowing procedures - are included as source code and can be called from within any program. First experiences with individual students indicate that the course could increase their interest in building architectural tools rather than merely using canned programs. It certainly aroused their interest in more sophisticated graphics applications and lead to questions how the limitations of the simple programs could be overcome. An "Introduction to Computer Aided Design" course was specifically designed as a follow-up to this course and is presently being taught.

6. REFERENCES


