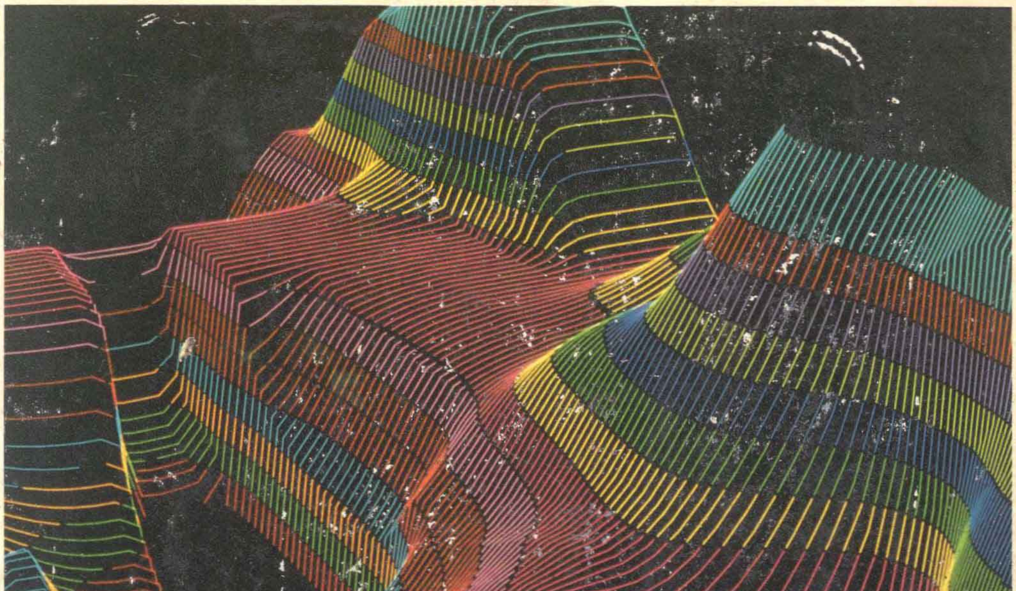
AN INTRODUCTION TO **NONPROCEDURAL LANGUAGES:** USING NPL™

THOMAS D. TRUITT
STUART B. MINDLIN



A BYTE BOOK

AN INTRODUCTION TO NONPROCEDURAL LANGUAGES

USING NPL™

Thomas D. Truitt
Stuart B. Mindlin
DeskTop Software Corporation

With contributions by
Tarralyn A. Truitt

McGRAW-HILL BOOK COMPANY
New York St. Louis San Francisco Auckland Bogotá
Hamburg Johannesburg London Madrid Mexico Montreal New Delhi
Panama Paris São Paulo Singapore Sydney Tokyo Toronto

This book was produced using WordStar on an Altos computer.
It was typeset by computer in Times Roman by AZTECH Document Systems Inc.
The editors were Charles E. Stewart and Joseph F. Murphy;
the production supervisor was Joe Campanella.
The cover was designed by Rafael Hernandez.
Halliday Lithograph Corporation was printer and binder.

Cover photograph courtesy of Melvin L. Prueitt, Los Alamos National Laboratory.

AN INTRODUCTION TO NONPROCEDURAL LANGUAGES
Using NPL™

1 2 3 4 5 6 7 8 9 0 HALHAL 8 9 8 7 6 5 4 3

ISBN 0-07-065301-1

Library of Congress Cataloging in Publication Data

Truitt, Thomas D.
An introduction to nonprocedural languages.

Bibliography: p.
Includes index.
1. Nonprocedural languages (Programming languages)
2. NPL (Computer program languages) I. Mindlin, Stuart B. II. Truitt, Tarralyn A. III. Title.
QA76.7.T78 1983 001.64'24 82-17315
ISBN 0-07-065301-1

FOREWORD

As Tom Truitt has noted in his Preface, I have been an advocate of the nonprocedural approach to computer application development for some years. Fortran and Cobol and Pascal will be around through our lifetimes, for a combination of good and bad reasons, but the excitement as we move through the eighties is in the nonprocedural arena.

Accordingly I was delighted to be asked by McGraw-Hill to review the manuscript of this book. I found the subject to be a familiar one, based on my own experience with a system that runs on the big mainframes, NOMAD, and I was astounded that something with a significant portion of the NOMAD capability could run on an Apple II Plus. The Apple is an impressive machine in its own right, but far from the capabilities—and price—of the machines in the IBM 370 line.

I expressed my enthusiasm to McGraw-Hill and gave Tom a call, asking if I could try out the system. No sooner said than done. I used NPL for a variety of record-keeping chores in connection with my college teaching, plus various other applications. A demonstration to a graduate database course resulted in a deeply impressed group of students and faculty.

You're going to be hearing a lot about NPL, especially as it becomes available on some of the newer and more powerful small computers. And the book is highly readable. A modest investment of reading and practice will have you doing things with a computer that would have been completely out of the reach of even the professional programmers a mere decade ago, and which even today are otherwise available only on very much larger computers.

Daniel D. McCracken
Author and Consultant
Professor of Computer Sciences
City College of New York

PREFACE

LANGUAGES FOR COMPUTERS AND PEOPLE

This is a book to teach you a whole new approach to coping with the lists of names and addresses, quantitative data, and other miscellaneous information that seem to clutter your life. There are five major sections. Three are tutorial in style, and two are reference material.

First-time users of computers may begin with Chapter 1, or with the Section Four tutorial. Choose the latter if you are already familiar with the Apple computer. This preface and Chapters 16 to 18 are written for more experienced computer users. The sections are organized as follows:

- Section One *Basic Language Features:* Chapters 1 through 13 present the nonprocedural language features required for data entry, for the query of datafiles, and for report writing.
- Section Two *Advanced Features for Database Management and Application Development:* Chapters 14 to 19 introduce the language features required for describing and using relational and hierarchical database structures, for file maintenance, and for executive level control of complex processes.
- Section Three *Language Reference Section:* A complete specification, in alphabetic order, of all features and keywords of the NPL language.
- Section Four *An NPL Tutorial for the Apple II Computer:* Three lessons offer step-by-step instructions for self-teaching the many features of the NPL language on a specific computer.
- Section Five *Appendixes:* Includes detailed language comparisons of five nonprocedural systems for IBM 370 computers, i.e., RAMIS II, FOCUS, NOMAD2, INQUIRE, and SQL.

PURPOSES

The book has three purposes:

- To introduce the first-time user of a microcomputer to an easier way of transforming raw data into useful information—i.e., easier than attempting to program with BASIC, COBOL, or other popular programming languages
- To provide a tutorial, a user's guide, and a reference manual for the NPL language, without the specific dimensions of the NPL software product intruding too much on the first purpose
- To suggest that microcomputers with nonprocedural software systems can become a natural training ground for people who must have access to data on very large computers, by showing the strong resemblances of

several large-computer, nonprocedural database systems and the methods taught throughout the book.

LANGUAGES

For better or worse there are hundreds of programming languages in use today. Some observers worry that a tower of Babel has been created by computer specialists and that as the number of languages grows the level of useful communications actually may be diminishing. Others are active in pursuing national and international standards for programming languages.

While the computer industry ponders these weighty matters, the practical people who have requirements for convenient access to their data and for methods of producing lists, tabulations, charts, and statistics have moved toward a different kind of standard, i.e., the English language. Computers are now beginning to listen in the language of their masters. This is an exciting trend, for it opens the doors for almost everyone to be a computer master, *without becoming a computer programmer.*

James Martin, in the preface to his *Application Development without Programmers,* writes, "Applications development did not change much for twenty years, but now a new wave is crashing in. A rich diversity of nonprocedural techniques and languages are emerging."

Daniel McCracken [1978] writes, "Most nonprocedural methods are used by people without data processing experience—which is another way of saying that the methods are used directly by the people with problems to solve, without going through the intermediary of a programmer at all. Taking all the factors together, I will venture to say that by the mid-'80s more than half of all computer time will be spent running applications programs constructed with nonprocedural oriented languages."

Software Terms and Tools

We think of *data* as numbers and words, which with appropriate sifting and sorting can be transformed into useful *information*—for accounting, financial planning, scheduling, and general support of decision making. Such data are stored in a *database,* which is generally a disk device with suitable software to provide convenient access. The most convenient of these software products employ a *query language* or an *application language* that is intended for people with a need for information but with no time or inclination to write conventional programs to get it. The table below lists selected database software products in four functional groups. Martin shows a larger list of software tools with additional classifications, and he discusses and compares the merit of many of the best products.

DATABASE SOFTWARE TOOLS FOR NONPROGRAMMERS

Software	Computer	Source
	Text retrieval systems	
INQUIRE	IBM 370	Infodata Systems, Inc.
STAIRS	IBM 370	IBM
	Database query languages	
ASI/INQUIRY	IBM 370	Applications Software Inc.
DATATRIEVE	DEC 11	DEC
EASYTRIEVE	IBM 370	Pansophic
MARK IV	IBM 370	Informatics General, Inc.
ON-LINE ENGLISH	IBM 370	Cullinane, Inc.
QUERY-BY-EXAMPLE	IBM 370	IBM
QWIK QUERY	several	CACI
SQL	IBM 370	IBM
SYSTEM-2000	several	MRI Systems Inc.
	High-level application tools and database systems	
ADMINS/11	DEC 11	Admins Corporation
APL*PLUS	several	STSC, Inc.
EXPRESS	IBM 370	PRIME Management Decision Systems
INFO	PRIME	HENCO Inc.
INQUIRE	IBM 370	Infodata Systems, Inc.
MANTIS	several	Cincom
NPL	micros & minis	DeskTop Software Corporation
RAMIS II	IBM 370	Mathematica, Inc.
SPSS	several	SPSS, Inc.
TROLL	IBM 370	National Bureau of Economic Research Inc.
	Application development systems (Fourth-generation languages)	
FOCUS	IBM 370	Information Builders Inc.
NOMAD2	IBM 370	National CSS, Inc.
MAPPER	UNIVAC 1100	Sperry Univac

Nonprocedural Systems

Many of the software tools listed in the table may be characterized, at least partly, as using a *nonprocedural* language, in which the "nonprogrammer" user makes requests. This term simply means the user states *what* he wants but not *how* the computer is to process the request.

Users of certain of the nonprocedural systems have been reporting enormous increases in productivity, that is, they have shortened the time to develop new database applications by factors of 4:1 to 12:1. So successful are they in quickly bringing satisfaction to end users that the name "Fourth Generation Languages" is used by some writers to distinguish nonprocedural database systems from others. Martin offers the criteria that the "Fourth-Generation" label be restricted to systems that provide development times of less than one-tenth those required by programming new applications in COBOL.

While the idea of nonprocedural programming is emphasized throughout this book, nonprocedurality is not an end in itself but rather a means of simplifying for *people* certain kinds of requests to *computers.* All complex applications use procedural steps, but these are higher-level processes required by the end user, not detailed computer instructions.

For example, when you give instructions for your car to be lubricated, have its oil changed, and its wheels balanced, it is sufficient to state what you want without telling the mechanic how to proceed. However, if you also wish the car to have an official state inspection, have the brake lights repaired, be waxed and polished, have scratches retouched with spray paint, and be washed, you probably would state these instructions in a preferred sequence.

Our nonprocedural request to the reader is that you lay aside apprehensions you may have about computers, or preconceived ideas about programming, as well as the notion that you "are not mathematically inclined." Just follow all the examples in the text, and you will soon know a new language, after only ten chapters.

Acknowledgments

We extend our sincere thanks for the seemingly perpetual assistance of David Oakley, Robert Mills, Richard Meyer, Diane Robertson, Carol Manela, and Dennis Papara; for editorial assistance by Lorraine Sichel; and for critical review and advice by Larry Manire.

Thomas D. Truitt
Stuart B. Mindlin

CONTENTS

SECTION **ONE**

BASIC LANGUAGE FEATURES

CHAPTER **1**

INTRODUCTION

The wonders of electronic technology developed during the 1970s have placed powerful computers in the hands, in the briefcases, and on the desktops of persons in all fields of endeavor — people who have practical uses for a machine that can bring order to some of their records or reduce the drudgery of some of their functions. In the same period a software industry has flourished, searching in all directions for methods, sometimes called *languages*, for improving communications between people and computing machines.

At the beginning of the 1980s neither the industry nor the larger community of users of computer languages have agreed just which are better than others, much less which languages are best for average persons wishing to analyze data critical to their jobs. This book is an introduction to a language called NPL, which resembles a class of computer languages widely used in industry for analysis of management information with some of the largest computers. It illustrates how these same methods may be employed by persons with a microcomputer.

Anyone with a small computer can make effective use of the NPL language, if only to keep track of personal finances and a mailing list of friends and relations. While these two applications alone may not justify the costs, the point is that the ability to instruct a machine to perform useful tasks is readily available. The price for getting started is a personal effort somewhere between that required to learn to use a calculator and that needed to read this book.

To instruct your personal computer, you first learn a few commands. Then you practice, and your vocabulary will grow quite naturally. Most people have found the experience both fascinating and rewarding.

NONPROCEDURAL PROGRAMMING

In contrast to most computer languages, there is a special group of programming systems called *nonprocedural languages*, which have evolved over a period of fourteen years to have an important distinguishing quality: they require very few instructions on just *how* to achieve a particular result. Rather, they expect the user to give instructions on *what* data are to be used and *what* the output formats are to look like. It is assumed that the software will know the exact step-by-step process to be followed. Thus the user is relieved of the procedural steps. For example, Nonprocedural languages make it possible to give a set of instructions which might resemble the following hypothetical verbal request given to an assistant:

> Get that file over there, select only the records that look like this one and that are over one year old; display all of the critical data, together with the results of these calculations and comparisons, sorted by month and by these two variables, in these tabular and graphic formats, with appropriate statistical summaries.

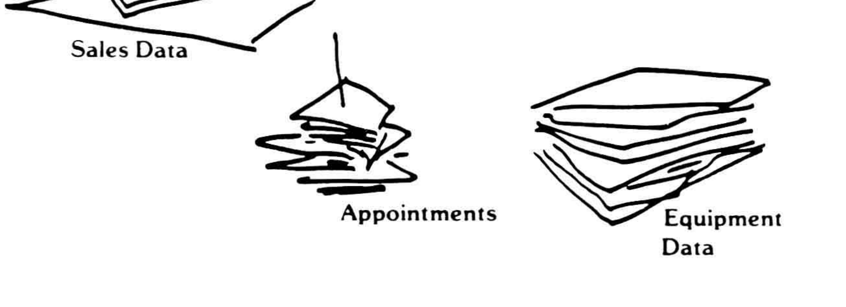
Procedural languages usually require a host of detailed sequential steps to fulfill such requests. Procedural languages offer many more options, but at a greater cost in time and effort by the programmer, in the majority of cases. Nonprocedural systems offer convenience and savings for a larger community of users at the price of modest limitations in the variety of display formats which might otherwise be available.

FROM "DATA" TO "INFORMATION"

NPL is a tool to help you convert your collections of miscellaneous *data* into meaningful *information*. No matter how comprehensive your data collecting may be, you will gain very little if you lack the means to sort, screen, collate, merge, classify, calculate, organize, and display data. What we need from these data lies in the interrelations, the averages, the trends, and the ratios among data items. It is this kind of information that allows us to plan, to forecast, to budget, and to make decisions. To achieve this goal, you need a rational *process* for data analysis.

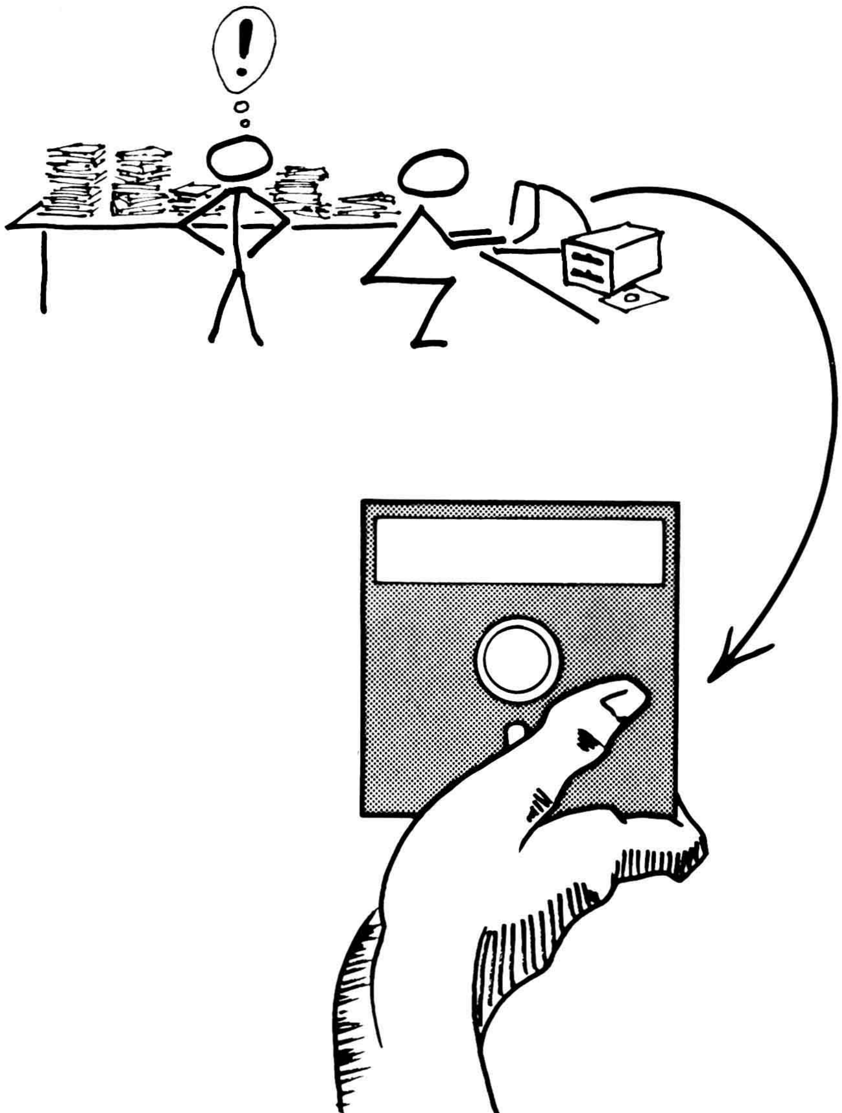
Here is a process you could follow:

Step 1: *Organize* the mounds of data records which clutter your life into piles that seem to have something in common.

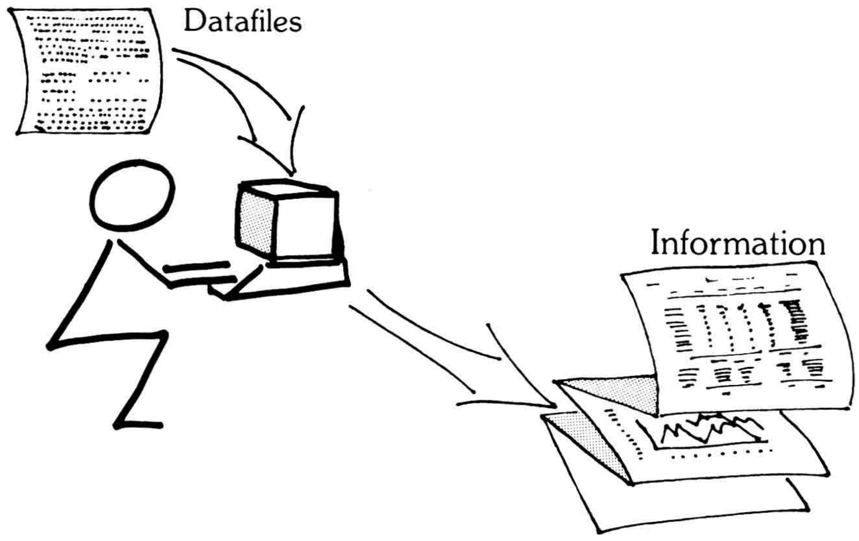


Step 2: *Describe* the data items (that is, the smallest units of information) which are common to each pile of records. For example, data item names associated with equipment data might be equipment name, quantity on-hand, and part number. Fieldnames associated with client data could include name, address, phone number, and social security number. The description of the common data items for a pile of records is stored in a special file called a master file.

Step 3: Once you have determined all of the fields associated with a collection of records, *capture* all the data items (e.g., names, addresses, part numbers) on a diskette. Do not copy, rewrite, or retype any item for the sake of neatness, or any other reason. Rewriting data items introduces errors. Capture the data now — make changes later. To capture data you use a special electronic form called a DATA-FORM.



Step 4: Sift, sort, screen, summarize, collate, classify, calculate, display, distribute, reorder, retype, and recap the data to your heart's content. A request that you make to NPL for information is called a *query*. NPL's response to your query is called a *report*.



A BROAD VIEW OF NPL

Before focusing on specifics, take a look at the NPL System Guide which appears in Appendix C and in the figures on the next few pages in a simpler form. The guide is intended as a ready reference card to help you remember the NPL keywords and their relation to the various modes. You should refer to the guide often while trying the sample problems in this book.

LEVELS AND MODES

The NPL SYSTEM GUIDE is a graphic way of showing you the relationships of the several operating states of NPL. It groups these states into modes and levels. At the lowest level is the Operating System, which is the software