*Advanced Pascal For Students*

·~

'·

*Assoc. Prof.* Dr. DOG~N IBRAHIM BSc MSc PhD MBCS MIEE CEng

*Computer Engineering Department, Near East University*

*Nicosia, Cyprus*

*1994*

1988

Near East University

;

(C)1994

All Rights Reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, including electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of the Author.

Printed & Bound in the

Turkish Republic Of Northern Cyprus, Nicosia.

**PREFACE**

This book is aimed for the second year University students studying Computer Programming, Computer Science, Computer Information Systems, or Computer Engineering. The topics in the book have successfully been thought to the second year Computer Engineering students at the Near East University. It is recommended that the book' should be completed in one semester.

The material in the book has been prepared with the SVS PASCAL compiler in mind, although the book could easily be used in teaching any PASCAL programming language. Some previous knowledge of another programming language would be useful, although not essential.

All the programs in the book have been tested and run on the UNISYS 6000/35 series computer of the Near East University, running the UNIX operating system.

Exercises are provided at the end of every chapter to help the students o practice their knowledge. The recommended method of study is such hat the students should have access to an online computer system, or preferably to a personal computer with the PASCAL compiler.

Chapter 2 is an introduction to PASCAL language and PASCAL variable types, constants, assignment statements and numbers are described with examples and exercises are given at the end of the chapter. Most of the concepts here are common to all high-level languages.

The structure of a PASCAL program is introduced in Chapter 3 and details of input and output are described with examples.

PASCAL control structures are presented in Chapter 4, again with many working examples to enable the students gain an insight to one of the most important topics of any programming language.

Chapter 5 presents functions in great detail. Both built-in and user defined functions are covered with examples on each topic. Like other chapters, exercises are provided at the end to enable students put their

owledge into practise.

Procedures are an important part of all high-level programming anguages and Chapter 6 describes the use of procedures in PASCAL programs.

3

The importance and the use of subscripted variables (arrays) are described in Chapter 7. Practical examples are given on both single and multi-dimensional arrays.

One of the powerful feautures of the PASCAL language is that it enables users to create their own data types. This is presented in Chapter 8 with working examples.

Strings and string handling are one of the most important feautures of high-level languages, enabling programmers to create user-friendly programs. The use of strings in PASCAL is presented in Chapter 9 with examples on the use of string processing functions.

Record structures enable a programmer to create complex data structures and the use of records is introduced in Chapter 10.

The use of files and file processing techniques are described in Chapter 11, again with working examples and exercises at the end of the chapter.

Array processing techniques, including the searching, stack operations, queue operations and sorting techniques are described in Chapter 12.

Perhaps one of the most powerful feutures of the PASCAL language is that it provides tools to programmers for the design of complex data structures in the form of linked lists. Several programming concepts and tools are provided in Chapter 13 to create complex data records, join them as linked lists and to manipulate the data easily in these lists.

I am grateful to my wife who checked the entire manuscript for errors and also put the manuscript into a form which can easily be understood.

**Dr. Doqan Ibrahim April 1994 - Nicosia.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Contents** | | | |  |
| PREFACE | |  |  | 3 |
| 1. | INTRODUCTION | | | 9 |
| 2. | DATA TYPES | |  | 11 |
|  | 2.1 | The Alphabet | | 11 |
|  | 2.2 | Variable Names | | 11 |
|  | 2.3 | Constants | | 12 |
|  | 2.4 | Variables | | 13 |
|  |  | 2.4.1 | Standard Variable Types | 14 |
|  |  | 2.4.2 | User Defined Data Types | 16 |
|  | 2.5 | Assignment Statements | | 17 |
|  | 2.6 | Arithmetic Expressions | | 18 |
|  |  | 2.6.1 | Hierarchy Of Operations | 19 |
|  |  | 2.6.2 | Numbers | 22 |
|  | 2.7 | Exercises | | 23 |
| 3. | STRUCTURE OF A PASCAL PROGRAM | | | 28 |
|  | 3.1 | Comments In PASCAL programs | | 29 |
|  | 3.2 | Displaying Data | | 30 |
|  | 3.3 | Output Field Width | | 34 |
|  |  | 3.3.1 | Displaying Integer Numbers | 34 |
|  |  | 3.3.2 | Displaying Real Numbers | 37 |
|  | 3.4 | Input In PASCAL | | 42 |
|  |  | 3.4.1 | Character Data | 47 |
|  | 3.5 | Exercises | | 50 |
| 4. | CONTROL STRUCTURES | | | 55 |
|  | 4.1 | WHILE Statement | | 58 |
|  | 4.2 | REPEAT Statement | | 64 |
|  | 4.3 | IF Statement | | 68 |
|  | 4.4 | FOR Statement | | 74 |
|  | 4.5 | CASE Statement | | 78 |
|  | 4.6 | GOTO Statement | | 83 |
|  | 4.7 | Reading A String Of Characters | | 85 |
|  | 4.8 | Exercises | | 87 |
| 5. | FUNCTIONS | |  | 92 |

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| *:* |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 11.1 | Text Files | | 179 |
|  | 11.2 | Creating A Text File | | 180 |
|  | 11.3. | Closing A File | | 180 |
|  | 11.4 | Opening An Existing File | | 182 |
|  | 11.5 | Reading Data From A File .. | | 183 |
|  | 11.6 | Exercises | | 195 |
| **12.** | **ARRAY PROCESSING** | | | **197** |
|  | 12.1 | Insertion in Arrays | | 198 |
|  | 12.2 | Deletion in Arrays | | 202 |
|  | 12.3 | Searching in Arrays | | ' 205 |
|  |  | 12.3.1 | Linear Search | 205 |
|  |  | 12.3.2 Binary Search | | 209 |
|  | 12.4 | Stack Operations | | 212 |
|  |  | 12.4.1 | PUSH Operations in Arrays | 212 |
|  |  | 12.4.2 POP Operations in Arrays | | 214 |
|  |  | 12.4.3 Postfix Expressions & Stacks | | 217 |
|  | 12.5 | Queue Operations | | 221 |
|  |  | 12.5.1 | Inserting Elements To A Queue 225 | |
|  |  | 12.5.2 Deleting From A Queue | | 227 |
|  | 12.6 | Sorting Arrays | | 231 |
|  |  | 12.6.1 | Selection Sort | 232 |
|  |  | 12.6.2 Bubble Sort | | 235 |
|  |  | 12.6.3 Insertion Sort | | 239 |
|  |  | 12.6.4 Quick Sort | | 242 |
|  | 12.7 | Efficiency Of Sort Algorithms | | 247 |
|  | 12.8 | Exercises | | 249 |
| **13.** | **POINTERS AND LINKED LISTS** | | | **252** |
|  | 13.1 | Pointers | | 253 |
|  |  | 13.1.1 | The NEW Statement | 255 |
|  | 13.2 | Processing Linked Lists | | 257 |
|  |  | 13.2.1 | Inserting a New Record | 258 |
|  |  | 13.2.2 -Removing a Record | | 260 |
|  |  | 13.2.3 Inserting Records Within | |  |
|  |  |  | a Linked List | 261 |
|  | 13.3 | Exercises | | 278 |
|  | **RECURSIVE PROGRAMMING** | | | **279** |
| **5.** | **FUNCTIONS REFERENCE** | | | **283** |
|  | Abs |  |  | 284 |
|  | Arctan |  |  | 285 |

|  |  |  |
| --- | --- | --- |
|  | Chr | 286 |
|  | Co neat | 287 |
|  | Copy | 288 |
|  | Cos | 289 |
|  | Delete | 290 |
|  | Eoln | 291 |
|  | Exit | 292 |
|  | Exp | 293 |
|  | Fill char | 294 |
|  | Halt | 295 |
|  | Insert | 296 |
|  | lsinf | 297 |
|  | lsnan | 298 |
|  | lsnum | 299 |
|  | Length | 300 |
|  | Ln | 301 |
|  | Odd | 302 |
|  | Pos | 303 |
|  | Pred | 304 |
|  | Round | 305 |
|  | Pwroften | 306 |
|  | Sin | 307 |
|  | Sqr | 308 |
|  | Sqrt | 309 |
|  | Su cc | 310 |
|  | Trunc | 310 |
| 16. | APPENDIX A | 312 |
|  | PASCAL RESERVED WORDS |  |
| 17. | APPENDIX B | 313 |
|  | PASCAL STORAGE ALLOCATION |  |
| 18. | APPENDIXC | 315 |
|  | USING THE SVS PASCAL COMPILER |  |
| 19. | APPENDIX D | 319 |
|  | ASCII CHARACTER CODES |  |
| 20. | APPENDIX E | 320 |
|  | BIBLIOGRAPHY |  |
| 21. | APPENDIX F | 322 |
|  | GLOSSARY OF COMPUTING TERMS |  |
| 22. | INDEX | 331 |

**1**

**INTRODUCTION**

PASCAL is one of the popular programming languages designed in the late 1960s. This programming language was first designed by Professor Niklaus Wirth of the Eidenissiche Technische Hocheschule of Zurich, Switzerland. PASCAL is not an acronym for anything and it is named after Blaise Pascal, the 17th century philosopher and mathematician.

For historical reasons, FORTRAN, COBOL and BASIC are more widely used high-level languages than PASCAL. FORTRAN, designed in the early 1950s, was the first widely used high-level computer programming language and this language is very useful for engineering applications. COBOL, developed in the late 1950s, was intended for business and data processing applications. BASIC is an easy to learn programming language which is available on nearly all computers, including the smallest types of home computers. Although the concepts of most programming languages are the same, PASCAL offers some useful concepts not found in other programming languages.

!though the early PASCAL was only available on the mainframe computers, it is now very commonly used on mini computers and personal computers (PCs).

PASCAL is one of the derivatives of the well known structured programming language known as ALGOL-60. PASCAL provides the user the ability to create his or her data types. This is a very powerful utility enabling the programmers to create very complex data types and data structures from simple data types.

PASCAL is a compiled language. PASCAL source code (or the PASCAL program) is written using a program editor. The source code is en compiled using a PASCAL compiler. The compiler generates an executable machine code which is then run on the host computer.

e PASCAL compiler described in this book is the Silicon Valley Software (SVS) compiler. This compiler is based upon the ANSI standard and it has a variety of nonstandard extensions to enable users

* carry out more complex programming tasks. SVS PASCAL compiler available on the Motorola 68020 and the Intel 386 based systems.

9

statements; statements;

The format of a PASCAL program is as follows:

**PROGRAM** name (input.output): declarations; declarations;

**BEGIN**

**END.**

A PASCAL program begins with the keyword **PROGRAM,** followed by a program name. Every line in a PASCAL program is terminated with ·a semicolon. There could be a number of declarations after the program name. The actual executable code starts after the keyword **BEGIN.**

The lines after BEGIN are valid executable PASCAL programming lines. Every PASCAL program is terminated with the keyword **END,** followed by a dot.

As an example, the following program (named SIMPLE), displays the message "A SIMPLE PROGRAM" on the screen:

**PROGRAM** SIMPLE(output); **BEGIN**

writeln("A SIMPLE PROGRAM");

**END.**

As we shall see in the next chapters, statement **writeln()** displays data on the screen and it is equivalent to the **PRINT** statement in BASIC.