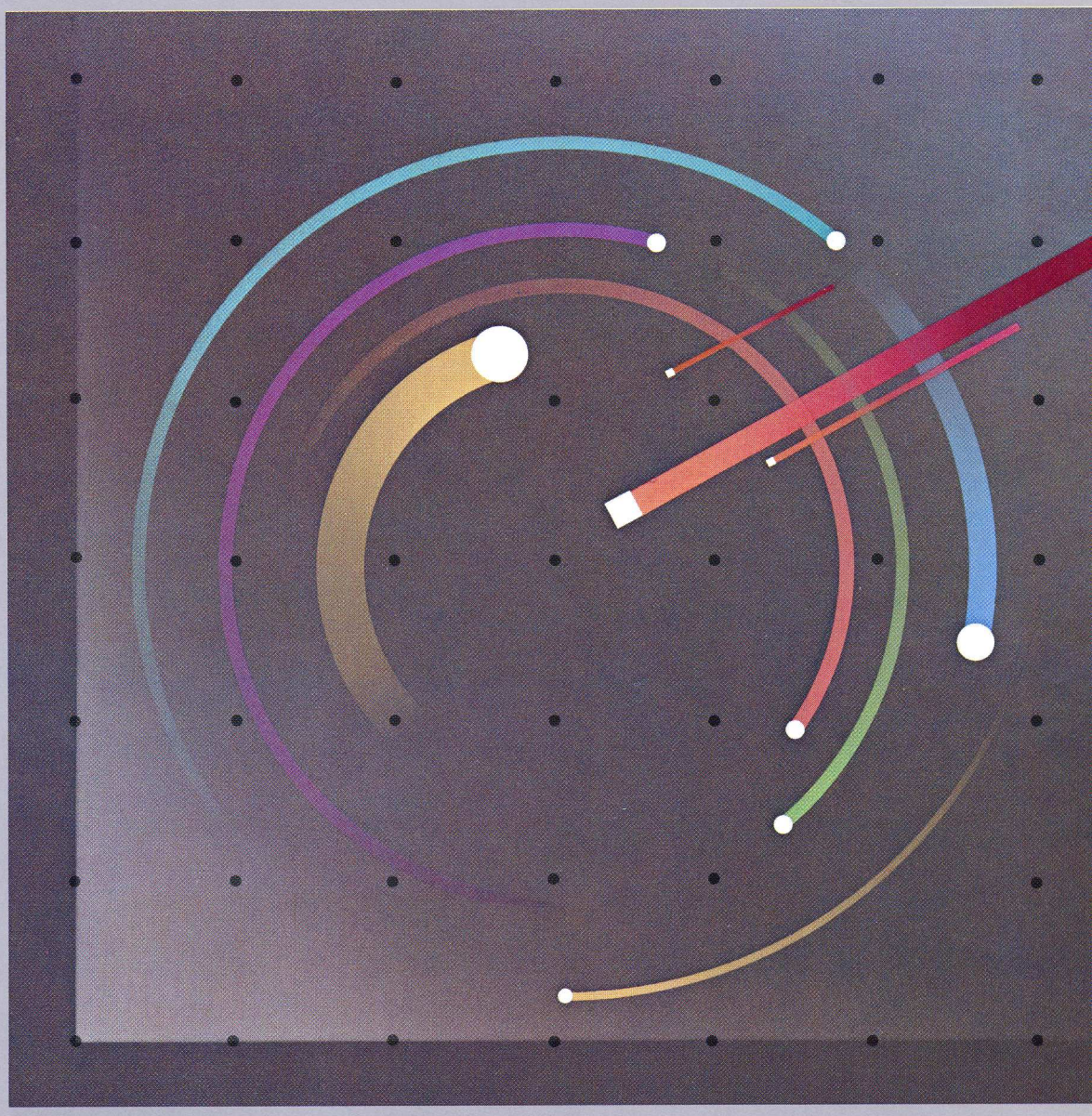
Pascal



Lisa's Pascal is a complete environment for developing Pascal programs. It includes a Pascal compiler, editor, linker, assembler, debugger, and a wide range of utilities.

A disciplined program structure, flexible data structures, and extensive data typing make Pascal an excellent vehicle for easily developing and maintaining Lisa™ applications and systems.

Pascal is a fast, effective way to develop applications for the Lisa system.

A powerful development language at your fingertips.
• Support for top-down, well-structured programs encourages programming that's reliable and easy to maintain.
• An assortment of variable types and structures are also available, including those defined by the user.

Pascal supports large program development.
• Units allow parts of a program to be compiled separately and linked to the main program later.
• Libraries provide access to collections of commonly used procedures and subroutines.
• Include files allow separately stored sections of source code to be included in a compilation.

Extensions aid application and systems development.
• Additions (such as procedure parameters, a "location of" operator, and the ability to link with assembly language) extend Pascal for use as a system programming language.
• IEEE Standard floating point computations provide unsurpassed accuracy for floating point calculations.
• Powerful compiler options include conditional compilation, compile-time variable declaration and assignment, include files, and control of range checking, listings, segmentation, and debugging information.

Product Highlights

A Complete, Powerful Development System
• Pascal contains all the development tools needed to create applications running in the workshop environment. This includes the Pascal compiler, linker, editor, assembler, debugger, and utilities. These same tools were used by Apple to develop application software for Lisa.

Workshop Manager
• Pascal and applications written with Pascal run in the Workshop, which is a complete development and execution environment similar to the Apple II and Apple /// Pascal environments.

Fast Development of Sophisticated Applications
• Pascal contains a host of features for efficient development of new applications programs. These features include a number of extensions to the ISO standard Pascal.
• Programs can be divided into units and segments, with separate compilations. This allows controlled development of different program sections and sharing of common libraries.
• Pascal includes a wide range of control structures, with varied conditional clauses. Pascal supports built-in memory management routines and a variety of built-in I/O capabilities.

Create Reliable, Easily Maintained Software
• Pascal has a number of features that encourage good programming practices. For example, data typing is strongly enforced. This insures that calls to procedures and functions contain the correct type and number of parameters, thus removing one of the major sources of errors in Pascal programs (particularly when routines are shared between programs).
• Nested procedure and function calls, disciplined program structure, and support for modular program development all contribute to software that is designed in a top-down, structured manner, which in turn, is reliable and easy to maintain.
• User-defined data types make it possible to specify English-like values; this helps make programs clearer. With subrange types, assignable values can be limited to just those that are specified by the programmer. This feature also helps make programs more dependable.

Extensions for Versatile Systems Development
• Pascal offers a number of enhancements for systems development. These include assembly language linkage, built-in functions for access to untyped memory locations, and procedure parameters.
• Pascal programs can access portions of the hardware interface in order to investigate mouse movement and keyboard events for stand-alone applications.
• Stand-alone Pascal applications can utilize the Lisa Quickdraw graphics routines. These routines control Lisa's elegant bit-mapped display images.

The assembler helps with speed-critical code and hardware interfacing.

- Since routines written in Lisa assembly language can be linked with Pascal programs, the main program can be written in Pascal and call assembly language routines for time-critical sections or for access to machine-dependent hardware features.
- Macros and conditional assembly are also provided.

Lisa's editor offers the standard Lisa user interface for fast editing and development.

- The editor uses the mouse, menus, and other Lisa elements to let you edit Pascal programs efficiently and easily.
- You can have multiple files—each in a separate folder—on the screen at one time. You can move between folders simply by selecting with the mouse.
- Source code can be transferred from one program to another, using the CUT & PASTE operation. Standard editing operations—such as FIND, REPLACE, INSERT, and UNDO—are also available.

A full set of utilities aid development.

- Among the utilities available are: performance analysis tools, segmentation management, system configuration, object file inspection and manipulation, and file transfer, compare, and search.

It's easy to transfer existing Pascal applications to Lisa.

- Pascal is based on ISO Pascal standard.
- It is very similar to Apple II and Apple /// Pascal—differences are well documented.
- Pascal will be compatible with future tools that will be made available for creating Lisa-style applications.

Specifications

I. Language Description

A. Program structure
Pascal programs have a strict structure for reliability and easy maintenance:
- Program heading.
- Label declaration.
- Constant definition.
- Type definition.
- Variable declaration.
- Procedure and function declaration.
- Executable statements.

B. Units
- Separately compiled object files, called units, may be created.
- They can then be linked with other units, or with the main program.
- Units may be nested.

C. Variable types
The following are supported:
- Integer.
- Long integer.
- Real (IEEE standard floating point).
- Boolean.
- Character.
- String.
- Enumerated.
- Subrange.
- Pointer.

Arrays, records, sets, and files containing the above types are also supported.

D. User-defined types
- Program test;

```
TYPE day = (Sun, Mon, Tues, Wed, Thurs, Fri, Sat);
VAR Deadline : Day;
  .
  .
IF Deadline = Fri then Panic;
```

E. Procedures and functions
- User-defined procedures and functions may be nested within both the main program and other procedures and functions.

F. Control structures
- Repetitive statements:
  WHILE...DO.
  REPEAT...UNTIL.
  FOR...DO.
- Conditional statements:
  IF...THEN...ELSE.
  CASE...OF...OTHERWISE.
- GOTO statement:
  GOTO label.
- This rich set of conditional and looping statements makes possible routines such as the following:

```
IF Day in [Wed..Sun].
THEN case Day of
    Wed:       writeln('Plenty of time.');
    Thurs:     writeln('One day left!');
    Fri:       writeln('Where is it?');
    otherwise writeln('Too late.')
END.
```

G. Input/Output
- I/O to block-structured, record-structured, and character-oriented devices is supported.
- Character-oriented I/O to text files and external devices:
  –READ (reads a series of values into a list of variables).
  –WRITE (writes out a series of values from a list of variables).
  –READLN (reads until the end of the line).
  –WRITELN (writes out an end-of-line when done).
  –EOLN (indicates whether an end-of-line has been reached).
- Block I/O: For fast bulk I/O, BLOCKREAD and BLOCKWRITE transfer single or multiple blocks at a time to and from disk files and external devices.
- General I/O: A number of functions are available for greater control over I/O:
  –RESET (OPENs existing file).
  –REWRITE (OPENs new file).
  –CLOSE.
  –EOF.
  –EOLN.
  –SEEK.
  –GET.
  –PUT.

H. Memory management
Commands available include:
- NEW (allocates memory used for dynamically allocated variables).
- MARK and RELEASE (reclaims memory used in the Pascal heap).
- MEMAVAIL (indicates the amount of memory available).

I. Mathematical functions
The following are available:
- Absolute value.
- Square.
- Square root.
- Sine.
- Cosine.
- Arctangent.
- Exponent (base *e*).
- Natural logarithm.

J. String Functions
The following are available:
- Length.
- Index.
- Concatenation.
- Substring.
- Delete substring.
- Insert substring.

Pascal Specifications

II. Execution Environment

Workshop Manager: a complete development and execution environment for independently developed applications. The workshop facilities are similar to the Apple II and Apple /// Pascal environments.

III. Compiler

- Code generation on/off.
- Debugging information.
- Conditional compilations.
- Compile time variable declaration, and assignment for conditional compilation control.
- Include files.
- Compile listings.
- Range checking on/off.
- Segmentation control.
- Stack expansion control.

IV. Assembler

- Macros.
- Conditional assembly.
- Linkage to Pascal programs.
- Full set of arithmetic operators.
- Listing controls.

Programs written in Pascal may call procedures and functions written in assembly language for speed-critical and highly machine-dependent activities.

V. Editor

The editor has the Lisa-style user interface. Its features include:

- Text files presented in folders on the screen.
- Operations selected from menus using the mouse.
- Multiple files displayed at once, each in its own folder.
- Text moved from one file to another simply by copying, or cutting and pasting.
- The mouse can also be used to:
  - open a folder.
  - start editing in any folder.
  - move a folder around the screen.
  - scroll up and down in a folder.
  - adjust selected text right or left.
  - select the display font.
  - set tab stops.
  - search for a string.
  - replace a string.
- Includes standard editing functions such as FIND, REPLACE, INSERT, and UNDO.

VI. Debugger

The Lisa debugger is a powerful low-level debugger offering these facilities:

- Display and set memory locations.
- Set and display register.
- Assemble and disassemble instructions.
- Set breakpoints, patchpoints, and traces at the machine level.
- Set up timing buckets for timing execution.
- Utility functions:
  - symbol and base conversion.
  - manipulating the hardware.

VII. Utility programs

- Debugging:
  - Pascal cross reference.
  - PATCH and DUMPHEX allow examination and modification of files using either ASCII or hexadecimal representation.
- File handling:
  - file comparison for both object and text files.
  - text search across multiple files.
  - scan multiple files by displaying them in quick succession.
  - file split and join.

VIII. System Configuration

- Pascal runs on any Lisa.
- Pascal Software
  - Compiler
  - Assembler
  - Linker
  - Editor
  - Debugger
  - Utilities
  - Workshop Manager
- Pascal Manuals
  - Reference Manual
  - MC68000 User's Manual
  - Workshop Manager Manual
  - System Software Manuals

Apple/U.S.
Apple Computer, Inc.
20525 Mariani Avenue
Cupertino, California 95014
(408) 996-1010
TLX 171-576

Apple/U.K.
Apple Computer (U.K.) Ltd.
Eastman Way
Hemel Hempstead
Herts HP2 7HQ
England
011-44-442-60244
TLX 851-825834

Apple/Europe
Apple Computer International
5/7 rue de Chartres
92200 Neuilly-sur-Seine
France
011-33-1-624-21-13
TLX 842-630296

Apple/Canada
Apple Canada
875 Don Mills Road
Don Mills
Ontario, Canada M3C 1V9
(416) 444-2531
800-268-7637
TLX 06-986561

A6F0006 100M-1/83 Printed in USA