

Major categories of software:

Operating system: A software package designed to control the basic input/output operations of the computer. The operating system handles tasks such as loading and running programs, storing data, and multiprocessing (which gives the appearance of running more than one program at a time). Examples: Windows XP, UNIX, LINUX, Macintosh OS 9.

Interpreter: A language translation program: translates one line of high level language at a time, executes the resulting machine language instructions immediately, then does next line, etc., if code doesn't contain errors (i.e., is executable).

Compiler: A language translation program used to transform high level code into machine-readable code.

Source code: this is high level, machine-independent, language, compiled into object code.

Object code is low-level, machine-dependent. There isn't a 1:1 comparison between source, object code. Compare these two time intervals: the "compile time" is the time required to translate source code into object code. The "run time" is the time the compiled program takes to execute.

Assembler: A program which converts symbolic language coded programs (using predefined symbols and mnemonics to represent instructions, addresses, variables, etc) into binary machine language (object) code: there is usually a 1:1 correspondence between each line of code and the binary numbers generated.

Loader: A program allowing transfer of a program from an external device (e.g. floppy disk, magnetic tape) into the computer's memory. This is a separate program for use on an assembler program's output. (Loader and assembler functions are both built into modern interpreter programs.)

Linker: this software allows combining of object programs (modules) and provides cross references between programs, and specifies location in memory where object program will be loaded. Used prior to loader. Also built into modern assemblers or compilers.

Monitor (or executive system): A primitive operating system on older computers. This software contains the necessary code to read essential keyboard commands, or branch to service or loading subroutines, and allows memory locations to be examined, changed, and verified. Monitors are usually resident in ROM (firmware).

Editor: A program permitting entry, editing, and storage of assembly language programs (or high level languages like Visual Basic). The program being edited is stored in a working area in memory so lines can be modified, deleted, etc. Programs are stored in text format (ASCII code). In Perl, there are specialized line editors available, or widely used word programs (like MS Word or NotePad) can be used.

Debugger: A program permitting checking, correcting of an object program residing in memory. Typical functions include insertion of breakpoints (program stops, displays variable values, next instruction to be executed, etc); restart after breakpoint, search for a specified value, etc. Simpler debuggers display and allow manipulation of hexadecimal-coded memory contents; fancier ones utilize symbolic codes for commands and addresses. Debuggers are features of most modern programming environments, like MS Visual Basic 6.0.

Diagnostic: Software that tests and detects errors or failures in hardware operation; often provided by the vendor of specific hardware components like disk drives.

Disassembler: A program that converts machine language into source programs (but it cannot reconstruct original symbolic names). Like a reverse transcriptase, it works backwards compared to an assembler.

Utility programs: These are standard routines like drivers for devices like a printer, disk drive, etc.

Library programs: These are software routines for math, graphics, string manipulation, etc., which can be linked as needed to your program, extending functionality of a language (usually from company marketing the original language). Libraries of Visual Basic and Perl programs are abundant.

Applications software: commercial programs tailored for specific tasks, e.g., word processor programs, DBMS, electronic spreadsheets, accounting packages.

Categories of applications software:

Integrated software: combines, in one package, modules for word processing, graphs, spreadsheets, database management (filing) routines., telecommunications (modem use). Example: MS Works, MS XP Office Suite, Corel Suite.

Financial software: bookkeeping, accounting (general ledger, accounts payable, accounts receivable, payroll, check-writing), investment tracking; special purpose: medical office account management. Example: Intuit Quicken

Database management systems (DBMS): files, reports, allows sorting, cataloging, search and retrieval of information; special purpose: scientific literature bibliography; patient record management. Example: FileMaker Pro

Number-crunching: spreadsheets with macros, graphic data analysis, statistics, curve-fitting, 3-D graphics. Special purpose: drug dosage calculations, data acquisition, "lab notebook" packages, scientific graphing (scatter, line, area, bar, pie charts, 3-D). Symbolic language processing. Example: Excel, Maple, Mathematica.

Graphics. design: drawing and painting programs,, click-art; pixel and object-oriented graphics; photo processing, CAD (computer-aided design), for pen plotting, audio-visual presentation graphics (slide, transparency making), animation tools. Special purpose: medical imaging. Example: Corel Draw

Desktop publishing (DTP): word processing plus page layout tools, laserwriter support, graphics insertion, color separation, font design, typesetting (e.g. TeX, Front Page)

Presentation software: slide shows, desktop presentations via projector, producing actual 35 mm slides (with an attached slidemaker) Example: MS PowerPoint

Communications: network management, file server software, telecommunications via modem, electronic mail (E-mail), connectivity. Examples: Eudora, MS Outlook, Outlook Express.

Word processors: prepare letters, documents with selection of fonts (including scientific, e.g., greek symbols, mathematical notation), page formats; spelling checker, punctuation, style editing, thesaurus, mail merge, outlining, OCR software. Special purpose: patient medical record entries. Example: Word, WordPerfect

System software: e.g. Mac OS 9 and X, Windows 98 and XP, UNIX, Linux, Iris.

Organizational tools: outlining, flowcharting, project management, expert systems, simulation tools, calendars. Special purpose: computer-aided diagnosis. Example: MacProject, MS Project.

Utilities: desk accessories (RAM-resident), disk copiers, disk management, encryption and security software, screen savers, anti-virus programs, backup, file recovery, disk management, file finders.

Multimedia: production software for combining sound tracks, video from tape, laser disk or live camera, CDROM; video animation, special video effects, titles, etc. (e.g. MacroMedia Director, Flash).

Clip Art: digitized images which can be "pasted" into other documents (using 'paint' or 'draw' graphics programs like SuperPaint). Available in assorted image formats including MacPaint, TIFF, PICT, PostScript, etc.; line drawings, digitized photographs in B&W, 1 bit, 8 bit gray scale, 24 or 32 bit color.

Desk Accessories: useful, small utility programs available within any application programs being executed; invoked on the Mac from the Apple pulldown menu on the Macintosh. Examples: Calculator, Rolodex.

Languages: C, C++, Visual BASIC, Prolog, LISP, Logo, Pascal, Forth, Modula 2, Fortran, Cobol, etc.

Programming tools: CASE (Computer-Aided Software Engineering), prototypers, debuggers.

Education: course builders, PILOT, simulators for various phenomena. Special purpose: medical education packages like HeartLab, Authoring software

Entertainment: action adventure, arcade, chess, bridge, flight simulators. Eg.: Doom, Half-Life.

Music: MIDI support, synthesizers, music composition, sheet music printing, MP3 encoders. Example: LimeWire.

INITs: programs which execute or are loaded initially when the computer's operating system software is first loaded. Examples: Startup Screen, Startup Sound, virus detection software, screensavers.

Fonts: character sets used in conjunction with word processors, graphics and all other applications programs using on-screen or printed text output. Different sets for LaserWriter, assorted sizes and formats. Examples: Times New Roman, Arial, Helvetica, Greek, Cyrillic.

Stacks: relational data files (text, graphics) accessed via HyperCard. (obsolete)

Tutorials. demos: help files, self-instruction 'tours' through applications programs.

Levels of computer languages:

Machine language: A computer 'understands' binary numbers only; but commands written out in binary are not easily understood or remembered; an easier to read equivalent is the decimal, hexadecimal or octal equivalent to the binary.

Assembler language: Uses symbolic equivalents for machine codes, symbolic addresses, variable names. Usually there's a 1:1 correspondence between symbols and machine instructions (each of which may take 1-4 bytes or more).

High level language: More concise; one instruction may equal many machine level or assembler instructions. There are numerous languages for general use. In science, there's FORTRAN, ALGOL; in business, it's COBOL; in general use are BASIC, C, PASCAL, ADA; compare structured vs. unstructured languages.

Problem-oriented language: For specialized applications, e.g., SPSS for statistics, STELLA for simulations; PILOT and LOGO for primary education; PLATO for secondary, college CAI; these languages are themselves written in high level language.

Comparison of language levels:

High level (BASIC): LETw=x+y-z

Assembler (Z-80): LD a,x
ADD a,y
SUB a,z
LD w,a

Object code: depends on where in memory the (assumed byte) values corresponding to w, x, y and z are located. Just suppose that x is at address A004, y is at A005, z at A006 and w at A007. Then

LD a,x = 3A04A0 (in hex) or	001110100000010010100000	(i n binary)
ADD a,y = C605A0	(this is a three byte instruction)	
SUB a,z = D606A0		
LD w,a = 3207A0		