System Software

A-Level Computer Science · Vocabulary in context

Word	Example
operating system /ˈɒpəreɪtɪŋ ˈsɪstəm/	Windows is an **operating system**. Windows
processor /ˈprəʊsesə/	The **processor** runs the program's instructions.
multitasking /ˈmʌltitæskɪŋ/	**Multitasking** lets you play music while you type.
memory management /ˈmeməri ˈmænɪdʒmənt/	**Memory management** gives each program its own part of RAM. RAM
memory protection /ˈmeməri prəˈtekʃn/	**Memory protection** stops one program changing another's data.
RAM /ræm/	Open programs are stored in **RAM**.
secondary storage /ˈsekəndəri ˈstɔːrɪdʒ/	A hard disk is **secondary storage**.
virtual memory /ˈvɜːtʃuːəl ˈmeməri/	**Virtual memory** uses the disk when RAM is full. RAM
paging /ˈpeɪdʒɪŋ/	**Paging** splits memory into blocks of the same size.
process management /ˈprəʊses ˈmænɪdʒmənt/	**Process management** decides which program runs next.
process /ˈprəʊses/	A program that is running is a **process**.
scheduling /ˈʃedjuːlɪŋ/	**Scheduling** gives each process a turn on the processor.

exercises.lol exercises.com.cn

Word	Example
time slice /taɪm slaɪs/	Each process runs for one **time slice**, then waits.
device driver /dɪˈvaɪs ˈdraɪvə/	A printer needs a **device driver**.
interrupts /ˈɪntərʌpts/	The keyboard sends **interrupts** when you type.
directory /daɪˈrektəri/	Put your files in a **directory** called Homework. Homework
access rights /ˈækses raɪts/	Students have **access rights** to read the file, but not change it.
authentication /ɔːˌθentɪˈkeɪʃn/	A password is a kind of **authentication**.
firewall /ˈfaɪəwɔːl/	A **firewall** blocks unsafe data.
hardware interrupt /ˈhɑːdweə ˈɪntərʌpt/	Pressing a key causes a **hardware interrupt**.
software interrupt /ˈsɒftweə ˈɪntərʌpt/	Dividing by zero causes a **software interrupt**.
interrupt handler /ˈɪntərʌpt ˈhændlə/	The **interrupt handler** runs when the interrupt arrives.
utility program /juːˈtɪlɪti ˈprəʊgræm/	A virus checker is a **utility program**.
antivirus /ˌæntɪˈvaɪrəs/	**Antivirus** software finds and removes viruses.
backup /ˈbækʌp/	Make a **backup** of your work every day.
compression /kəmˈpreʃn/	**Compression** makes the file smaller.

Word	Example
disk defragmenter /dɪsk ˌdiːˈfrægmentə/	A **disk defragmenter** puts the parts of each file back together.
signatures /ˈsɪgnɪtʃəz/	Antivirus software looks for the **signatures** of known viruses.
fragmented /frægˈmentɪd/	The file is **fragmented**: its parts are all over the disk.
incremental backups /ˌɪŋkrɪˈmentl ˈbækʌps/	**Incremental backups** save only the files that changed.
program library /ˈprəʊgræm ˈlaɪbrəri/	A **program library** holds code that many programs can use.
subroutines /ˈsʌbruːtiːnz/	Split the program into small **subroutines**.
library routines /ˈlaɪbrəri ruːˈtiːnz/	**Library routines** are ready-made and tested.
static library /ˈstætɪk ˈlaɪbrəri/	A **static library** is copied into the program.
executable /ɪgˈzekjʊtəbl/	The compiler makes an **executable** file.
dynamic library /daɪˈnæmɪk ˈlaɪbrəri/	A **dynamic library** is loaded when the program runs.
translator /trænˈsleɪtə/	A compiler is a kind of **translator**.
machine code /məˈʃiːn kəʊd/	The processor runs **machine code**: only 0s and 1s. 0 1
assembler /əˈsemblə/	The **assembler** turns assembly language into machine code.

exercises.lol exercises.com.cn

Word	Example
assembly language /əˈsemblɪ ˈlæŋgwɪdʒ/	**Assembly language** uses short words such as ADD and LDM. ADD LDM
compiler /kəmˈpaɪlə/	The **compiler** turns the whole program into machine code.
object code /ˈɒbdʒekt kəʊd/	The compiler gives **object code**.
interpreter /ɪnˈtɜːprɪtə/	An **interpreter** runs the program one line at a time.
bytecode /ˈbaɪtkəʊd/	Java code is first turned into **bytecode**. Java
virtual machine /ˈvɜːtʃuːəl məˈʃiːn/	The **virtual machine** runs the bytecode.
just-in-time compilation /dʒʌst ɪn taɪm ˌkɒmpɪˈleɪʃn/	**Just-in-time compilation** turns code into machine code as it runs.
portable /ˈpɔːtəbl/	The program is **portable**: it runs on Windows and on a Mac. Windows Mac
integrated development environment /ˈɪntɪgreɪtɪd dɪˈveləpmənt enˈvaɪrənmənt/	Python's IDLE is an **integrated development environment**. Python IDLE
debugger /ˈdiːbʌgə/	The **debugger** helps you find mistakes.
context-sensitive prompts /ˈkɒntekst ˈsensɪtɪv prɒmpts/	**Context-sensitive prompts** suggest what to type next.
auto-complete /ˈɔːtəʊ kəmˈpliːt/	**Auto-complete** finishes the name of the variable for you.
dynamic syntax checks /daɪˈnæmɪk ˈsɪntæks tʃeks/	**Dynamic syntax checks** show mistakes as you type.

Word	Example
prettyprint /ˈpretɪprɪnt/	**Prettyprint** colours and spaces the code so it is easy to read.
syntax highlighting /ˈsɪntæks ˈhaɪlaɪtɪŋ/	**Syntax highlighting** shows keywords in colour.
breakpoints /ˈbreɪkpɔɪnts/	Set **breakpoints** to stop the program at chosen lines.
single stepping /ˈsɪŋgl ˈstepɪŋ/	**Single stepping** runs one line at a time.
report window /rɪˈpɔːt ˈwɪndəʊ/	The **report window** shows the values of the variables.
version control /ˈvɜːʃn kənˈtrəʊl/	**Version control** saves every change to the code.
refactoring /rɪˈfæktərɪŋ/	**Refactoring** makes the code clearer without changing what it does.
unit test /ˈjuːnɪt test/	A **unit test** checks one small part of the program.

exercises.lol exercises.com.cn