A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

5.1.2 Program libraries and DLL files

Name: _____ Class: _____ Date: _____ **Total: 36 marks**

KEY WORDS program library 程序库 • library routines 库例程 • static library 静态库 • dynamic library 动态库 • executable 可执行文件

Objective

Learn to answer exam questions on **program libraries** 程序库 and **Dynamic Link Library (DLL)** files: what a library is, the benefits to a developer of building software from library files, and how a DLL differs from a library that is copied into the program. This is the last sheet of 5.1; language translators follow on sheet 5.2.

You must be able to:

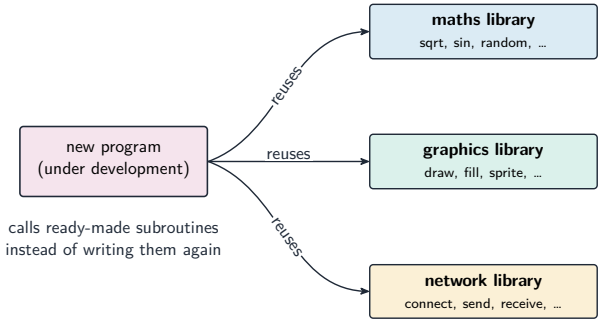
- define a program library, and give examples of what one contains
- explain the benefits to the developer of using **library routines** 库例程, and state the drawbacks
- explain the difference between a **static library** 静态库 and a **dynamic library** 动态库
- explain the benefits and drawbacks of DLL files

1 Worked examples

Study these first. Each one shows a **method** that a question later on this sheet needs.

■ What a program library is

- A program library is a collection of **pre-written, tested routines** (subroutines, classes, modules) that a program can use instead of the developer writing them.
- Examples: a maths library (square roots, trigonometry), a graphics library, a network library, an encryption library, a sorting library.
- The two-mark definition: *pre-written code, already tested, that a programmer can include in their own program.*



■ The benefits to the developer

The mark scheme wants the developer's **saving**: time, cost, expertise, reliability or maintenance.

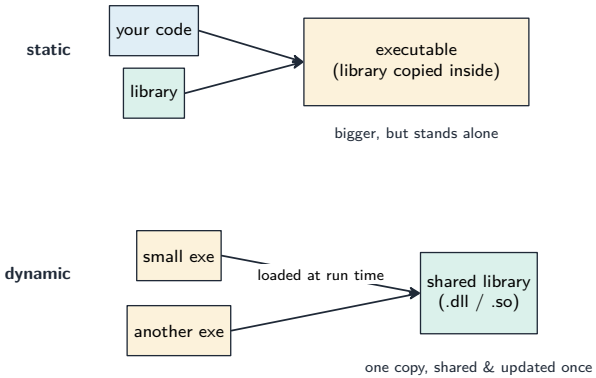
benefit	why
saves time and cost	the routine is already written; nothing to design, write or test
reliable	it has been tested and is used by many programs, so it is largely error-free
no expertise needed	the developer need not know how to write encryption, graphics or path-finding
consistent, easier to maintain	every program gets the same result; the common code lives in one place, so a fix is made once

Drawbacks: the routine may not do exactly what is needed and cannot be changed; the program depends on the library being available, correct and secure.

■ Static or dynamic?

- A library routine reaches the running program in one of two ways:

	static library	**dynamic library (DLL, `.so`)**
linked	at **compile** time: copied into the executable	at **run** time: loaded when called
the executable	larger, but stands alone	smaller, but needs the library file
sharing	each program has its own copy	one copy in memory, shared by programs
updating	recompile every program	replace the one file; every program gets the fix



■ **DLL files: benefits and drawbacks, for four marks**

Each benefit is a fact **plus its consequence** (two marks). *Explain two benefits of using DLL files:*

- the routine is loaded only when the program calls it, at run time, so the executable is **smaller** and memory is used only while the routine is needed
- one copy of the DLL is **shared** by every running program that uses it, so less memory is used and a bug is fixed **once**, without recompiling the programs

Drawbacks: the program will not run if the DLL is **missing**, moved or the wrong version; an updated DLL can change behaviour and break a program; a fake DLL put in its place runs with the program's rights.

2 Practice

Now use the methods above. The first questions need one idea each.

2.1 Define the term **program library**. [2]

2.2 Tick (✓) one box in each row to show whether the statement describes a static library or a dynamic library. [4]

Statement	static	dynamic
The routines are copied into the executable when the program is compiled.		
The library file is loaded into memory only when the program calls it.		
Several running programs share one copy of the library.		
Every program that uses the library must be recompiled when the library is updated.		

2.3 State **two** benefits to a programmer of using DLL files. [2]

2.4 State **one** drawback of using library routines. [1]

2.5 A programmer's program calls the library routine `SquareRoot`. State **two** reasons the programmer uses the routine instead of writing their own. [2]

2.6 State what a program needs, at run time, to use a routine in a DLL. [1]

3 Exam-style questions

3.1 A team is writing the control software for a delivery robot. The team uses a program library that includes a routine to find the shortest path between two points.

(a) **Explain two** benefits to the team of using the library routine. [4]

(b) **Describe one** drawback to the team of using the library routine. [2]

3.2 The team writes its own routines for controlling the robot's motors and decides to put them in a Dynamic Link Library.

(a) **Explain two** benefits to the team of creating the routines as a DLL. [4]

(b) A later update to the DLL causes the robot's older software to stop working. **Explain** why this can happen with a DLL. [2]

3.3 A software company sells a game. The game uses a graphics library.

Compare linking the graphics library statically with using it as a dynamic library, for the customers who install the game. Give **two** points for each. [4]

3.4 A programmer writing a banking app uses an encryption library.

(a) **Explain** why using the library is safer than the programmer writing their own encryption. [2]

(b) **Explain** one way in which relying on the library could make the app less safe. [2]

3.5 Three programmers each write a routine to check that a date is valid, and the three routines give different results.

(a) **Explain** how putting one date-checking routine in a program library would help the team. [3]

(b) **State one** other benefit to the team of building its own library of routines. [1]

Solutions

Each answer shows the steps, then the **result**.

2.1

- a collection of pre-written routines (subroutines, classes or modules)
- that have already been tested, and that a programmer can include and call in their own program

2.2

- copied into the executable at compile time: **static**
- loaded only when called: **dynamic**
- one shared copy: **dynamic**
- every program must be recompiled: **static**

Statement	static	dynamic
The routines are copied into the executable when the program is compiled.	✓	
The library file is loaded into memory only when the program calls it.		✓
Several running programs share one copy of the library.		✓
Every program that uses the library must be recompiled when the library is updated.	✓	

2.3 any **two** of:

- the executable file is smaller
- the DLL is loaded only when it is needed, so less memory is used
- one copy is shared by several programs
- the DLL can be updated without recompiling the programs that use it

2.4 any **one** of:

- the routine may not do exactly what is needed and cannot be changed
- the program depends on the library being available and correct
- the programmer must learn how the routine is called

2.5 any **two** of:

- the routine is already written, so it saves time
- it has already been tested, so it is reliable
- the programmer does not need to know the mathematics of how to calculate a square root

2.6

- the DLL file must be present (and the correct version) so it can be loaded into memory

3.1

(a) two benefits, each with its consequence:

- the routine is already written and tested, so the team saves the time it would take to write and test path-finding code
- the team does not need expertise in path-finding algorithms, so it can spend its time on the robot's own features
- (also accept) the routine is used by many programs, so it is reliable and the robot is less likely to have a fault in this part

(b) any **one**, described:

- the routine may not handle the robot's exact needs, for example moving obstacles, and the team cannot alter it, so it must work around the limits
- the software depends on the library: a bug or a security weakness in the routine becomes a bug in the robot

3.2

(a) any **two**, each explained:

- the routines are loaded only when the software calls them, so the executable is smaller and memory is used only while the motors are being controlled
- one copy of the DLL is shared by every program that controls the robot, so less memory is used
- the motor routines can be updated or fixed by replacing the DLL, without recompiling the programs that use it

(b) the DLL is loaded at run time, so the older software now runs with the new routines, not the ones it was tested with

- the update changed the behaviour or the interface of a routine, so the calls in the older software no longer work as expected

3.3 two points for each:

- **static:** the game is one larger file that runs on its own, with nothing else to install; the library cannot go missing or be the wrong version, but a graphics bug can only be fixed by shipping a new version of the game
- **dynamic:** the download is smaller and the DLL is shared with other games that use it; the library can be updated separately, but the game will not run if the DLL is missing or the wrong version

3.4

(a) encryption is very hard to write correctly, and a small mistake makes it insecure

- the library routine has been written by experts and tested by many users, so it is far less likely to contain a weakness

(b) the app depends on the library: a weakness or a bug in the library is a weakness in the app

- the programmer did not write the code and may not understand it, so a fault or an out-of-date routine can go unnoticed

3.5

(a) every programmer would call the same routine, so every part of the program checks dates in the same way

- the routine is written and tested once, so the check is correct everywhere, and a fault is fixed in one place
- the programmers save time: nobody writes the check a second time

(b) any **one** of: the library can be reused in the team's next program; a new programmer can use tested routines at once