Applications Generation (1.2.2)

Generic application: does not have a specific purpose. Tend to be the most installed and used products. Example: Word Processor.

Disadvantage: Large Storage Size
Advantages:

- Relatively Cheap, Thoroughly pre-tested, Lots of user support

Bespoke application: has a specific purpose. Can only be used for that specific purpose, but it is essential for that specific purpose. Only installed when you need it. Example: DBMS.

Advantages:

- The software will work exactly how you want it to
- The software will only have the features you need

Disadvantages:

- Long Development time
- Very expensive
- Little external user support

Don't use brand names in the exam.

Utility software: Software designed to keep your computer safe and keep it running efficiently

File Repair: Restores corrupted files to their original state

Backups: Making copies of data in case the original data is deleted. You can perform full backups or incremental backups.

Defragmentation: Reorganises fragments on a hard disk, putting fragments of files and free space together. Reduces the movement of the read/write head across the surface of the disk, which speeds up file access. Solid state drives have no moving parts. Defragmentation on them is unnecessary and reduces the drive's lifespan.

Open Source	Users	Creators
Benefits	Little or no restrictions on the use and modification of the code Can access the source code	Gets software out into wider community
Drawbacks	May not be fully tested	No income

Closed Source	Users	Creators
Benefits	Well supported, tested, and professionally built	Protected by the Copyright Design and patents Act Receive income
Drawbacks	Can't modify or distribute code Can't access source code	Constant demand for more features

Source code is easy for us to understand, read, maintain, and debug. However, machines only understand pure binary. Translation: The process of converting source code into machine code.

Assembler: Translates assembly language into machine code. One-to-one process.
Assembler: Advantages:

- Mnemonics are easier to remember
- Higher speed of execution than with compiled code

Assembler: Disadvantages:

- Program is more complicated to implement
- Difficult to remember the syntax

- For a specific processor

Compiler: Translates source code from high level languages into object code. The whole program is translated into machine code before it is run
Compiler: Advantages:

- Speed of execution is faster than with interpreted code
- Original source code is kept secret.

Compiler: Disadvantages:

- The program cannot run with syntax errors
- Code must be recompiled when the code is changed

Interpreter: Translates source code from high level languages into machine code line by line as the program is running.
Interpreter: Advantages:

- Program will run, even with syntax errors
- Code does not need to be recompiled when code is changed

Interpreter: Disadvantages:

- Translation software is required at run-time
- Source code is required

Lexical Analysis: The lexer scans the source code letter by letter. When it encounters a white space, operator symbol or special symbol, it decides that a word (lexeme) is complete. It checks if the lexeme is valid using a predefined set of rules. It is converted into a token. All the white space and comments will be removed. Input all tokens into a symbol table. The token stream and the symbol table are passed to the Syntax Analyser.

Syntax Analysis: Analyses the token stream, checking if it is in the correct syntax. Creates an error report (with the exact lines and locations of the errors). Builds an abstract syntax tree. When an identifier is added to the abstract syntax tree, the symbol table is checked to see if it exists.

Code Generation: where the abstract syntax tree is converted into object code.

Code Optimisation attempts to reduce the execution time and memory usage of the program as much as possible by:

- Removing unreachable code (program code that is never accessed because of programming constructs)
- Removing subroutines that are never called
- Removing variables and constants that are never referenced.

However, it can considerably increase compilation time for a program.

Advantages of using library routines:

- Quick and easy to use and hook into your own code
- Pre-tested and Pre-compiled

Disadvantages of using library routines:

- Adding functionality can be impossible
- Sometimes you don't know how efficient the code is (Black-boxing)

The linker: Links code and external library routines together correctly.
Static Linking: All code is in one large executable file.
Dynamic Linking: Libraries and code are in separate executable files, linked together. Compiled machine code is smaller. But, if the libraries change, the program may reach a syntax error.
The loader: The part of the OS that loads the executable files and libraries into memory.