Makefile

---

1. what is make and makefile ?

make is a build automation tool. It reads a file called Makefile and executes commands based on the rules defined in that file. It's primarily used for compiling and linking programs.

Makefile is a text file that contains a set of rules that make uses to determine which commands to execute.

2. what are target, dependencies and commands in Makefile ?

target: The name of the file or action you want to create or perform. It can be an executable, an object file, or a label for a set of commands.

dependencies: A list of files that the target depends on. If any dependency is newer than the target, the commands will be executed.

commands: The shell commands to execute to create or update the target. They must be preceded by a tab character.

3. write a makefile for generating binary which have main.c add.c sub.c mul.c div.c files.

4. what all are the auto-variables ? How are they useful?

Auto-variables are special variables that make automatically sets during the execution of a rule. They provide information about the target, dependencies, and other aspects of the current rule. They are very useful for writing generic rules.

- **$@:** The filename of the target.
- **$<:** The filename of the first dependency.
- **$^:** The filenames of all dependencies, with spaces between them.
- **$%:** The target member name, when the target is an archive member.
- **$*:** The stem of the target name (without the suffix).

5. write a makefile illustrating the example of auto variables.

6. How to execute a makefile from another makefile ? Write a Makefile command.

7. What is phony target in makefile ?
A phony target is a target that doesn't represent a file. It's used to execute commands, even if a file with the same name exists. It's declared using .PHONY

8. What is the difference between makefile and Makefile?
make looks for files named Makefile or makefile in that order. It is best practice to use Makefile for better portability

9. Write a password security system in Makefile ?

10. Write a Makefile illustrating the example of Wildcard ?

11. what does mean by
make distclean make install
make clean make all

- **make clean:** Removes object files and other intermediate files.
- **make all:** Builds the entire project (usually the default target).
- **make install:** Installs the built program and its related files to the system.
- **make distclean:** Removes everything created by make, including configuration files and archives.
-

12. what are the macros in Makefile ? Write some examples for this.
macros (also often called variables) are used to store text strings that can be expanded later.
Macros are expanded using $(variable) or ${variable}. $(variable) is the more common and recommended syntax.

- **= (Recursively Expanded):** This is the basic assignment. The value is expanded whenever the macro is used. This can lead to unexpected behavior if the value contains other macros that are later redefined.
- **:= (Simply Expanded):** The value is expanded only once, when the macro is defined. This is generally preferred for most cases.
- **+= (Append):** Appends a value to an existing macro.

- **?= (Conditional Assignment):** Assigns a value only if the macro is not already defined.

13. explain use of include and override in Makefile.
include: Includes another Makefile. This is useful for sharing common rules and variables.
override: Overrides a variable defined elsewhere, even if it's already defined with a value.

14. what is the benefit of Makefile ?
- **Automation:** Automates the build process, reducing manual effort.
- **Dependency tracking:** Rebuilds only the necessary files, saving time.
- **Consistency:** Ensures that the build process is always the same.
- **Portability:** Can be used on different systems with minimal changes.

15. generate a binary of any C program Using makefile and shell script.