**RIPHAH INTERNATIONAL UNIVERSITY ISLAMABAD**



**Lab #06**

**Bachelor of Computer Science – 5th Semester**

**Subject: Operating system**

**Submitted to: MS. Kausar**

**Submitted by: Eraj Shakkor**

**Sap ID: #33131**

**Q1: Explain the process of compiling a C program in Linux. What command is used to compile the program?**

The process of compiling a C program in Linux involves several steps:

1. **Preprocessing**: The preprocessor reads the source code, expands macros, and includes header files.
2. **Assembly**: The assembler translates the assembly code into machine code.
3. **Linking**: The linker resolves external references and creates an executable file.
4. **Compilation**: The compiler translates the preprocessed code into assembly code.

The command used to compile a C program is **gcc** (GNU C Compiler). The basic syntax is:

**gcc eraj.c -o eraj**

"**eraj.c**" (only save files with .c extension only) be a file that contains the C code.

**Q2: What is the purpose of the -o option in the gcc command? Provide an example.**

The **-o** option specifies the output file name for the compiled program. If not specified, the default output file name is **a.out**.

Example:

**gcc eraj.c -o eraj**

**Q3: What is the difference between g++ and gcc? When would you use each?**

**gcc** is the GNU C Compiler, which compiles C code. **g++** is the GNU C++ Compiler, which compiles C++ code. The main difference is that **g++** also links against the C++ standard library, whereas **gcc** does not.

Use **gcc** for compiling C programs and **g++** for compiling C++ programs.

**Q4: How do you compile and run a C++ program from the terminal? Provide the necessary commands.**

To compile a C++ program:

g++ -o myprogram myprogram.cpp

To run the program:

./myprogram

This command compiles **myprogram.cpp** into an executable program named " **myprogram** " that you run by typing **./myprogram** at the command line. It does nothing more than print " **myprogram**...." on the screen.

**Q5: What are templates in C++ in Linux? Write a simple example of a function template.**

Templates in C++ allow for generic programming, where a function or class can work with different data types. Here's a simple example of a function template:

template <typename T>

T max(T a, T b) {

return (a > b) ? a : b;

}

This **max** function template can be used with different data types, such as **int**, **float**, or **double**.

**Q6: Discuss the significance of file extensions in C programming. Why should source files be saved with .c or .cpp extensions?**

File extensions help the compiler and other tools identify the type of file and how to process it. In C programming, source files should be saved with **.c** extensions for C code,  **.cpp** extensions for C++ code. This convention helps the compiler and other tools recognize the file type and compile it accordingly.

**Q7: What are the common errors that can occur when compiling C programs, and how can they be resolved?**

Common errors include:

* Syntax errors: Fix by correcting the syntax.
* Undefined references: Fix by including the necessary header files or libraries.
* Compiler warnings: Fix by addressing the warnings, such as unused variables or functions.

**Q8: Explain how you can manage permissions for an executable file in Linux. What command is used for this purpose?**

To manage permissions for an executable file, use the chmod command. For example, to make an executable file readable, writable, and executable by the owner, and readable and executable by the group and others:

**chmod 755 myprogram**

In Linux, managing permissions for an executable file is done using the chmod command, which stands for "change mode." Each file in Linux has permissions associated with three categories:

1. **User (u)** – The owner of the file.
2. **Group (g)** – The group to which the file belongs.
3. **Others (o)** – Everyone else.

Permissions for each category are represented by three types:

* **Read (r)** – Allows viewing the file’s contents.
* **Write (w)** – Allows modifying the file.
* **Execute (x)** – Allows running the file as a program.

**Q9: What is a tarball, and what advantages does it offer for distributing software on Linux? Discuss the limitations of using tarballs for software installation and management.**

A tarball is a compressed archive file that contains a collection of files and directories. Tarballs offer advantages such as:

* Easy distribution and installation of software
* Compression reduces file size
* Preserves file permissions and timestamps

However, tarballs have limitations, such as:

* No dependency management
* No automatic uninstallation
* No easy way to update or uninstall software

**Q10: Explain the purpose of the RPM package format and how it addresses the shortcomings of tarballs.**

RPM (Red Hat Package Manager) is a package format that addresses the limitations of tarballs. RPM provides:

* Dependency management
* Automatic uninstallation
* Easy updates and patches
* Digital signatures for package verification

RPM packages are widely used in Linux distributions, such as Red Hat, Fedora, and CentOS, for easy software installation and management.