Operating Systems Lab

Fall 2024

Lab Task 06:

Understanding RPM/YUM Package Management

Tooba Baqai

46489

BSCS-5

Lab Instructor:

Kausar Nasreen Khattak

Email:

kausar.nasreen@riphah.edu.pk

Lab Task

Note: Include screenshots, required to illustrate your explanation for all Questions.

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

- Run the command sudo apt install to install a package.
- Execute sudo apt install gcc to install the GCC compiler.
- Execute sudo apt install g++ to install the G++ compiler.
- Run sudo apt update to update the package list.
- Create a new file by typing nano file.cpp.
- In the GNU nano editor, write your C program.
- Press Ctrl + X to exit the editor.
- Press Y to confirm saving the file, then press Enter.
- Run chmod 777 file.cpp to grant all permissions to the file.
- Compile the C program with gcc file.cpp -o lab to create an executable named "lab".
- Finally, run ./lab to execute the program and see the output.

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

The -o option in the gcc command allows us to specify the name of the output file. Without this option, the compiler creates a default output file named a.out. By using the -o option, we can customize the output file's name. For example, executing gcc file.c -o myprogram will compile the file.c file and create an executable named myprogram instead of the default a.out.

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

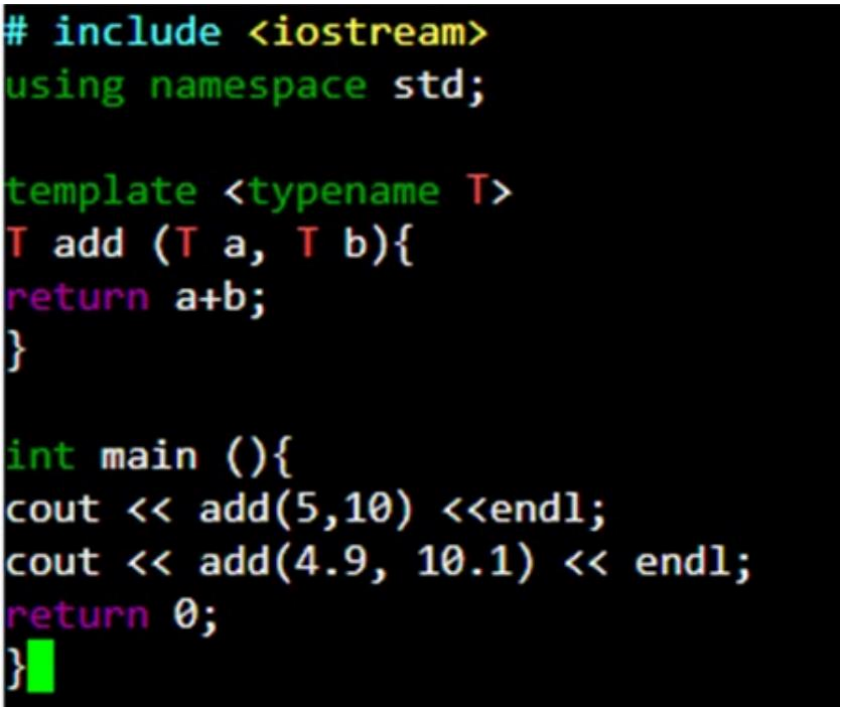
The key difference between g++ and gcc is that gcc is primarily used for compiling C programs, while g++ is designed specifically for compiling C++ programs. Although gcc can also compile C++ code, it doesn't automatically link with C++ libraries during the compilation process. To compile C++ programs with proper linking to C++ libraries, it's recommended to use g++. In summary, use gcc for C programs and g++ for C++ programs to ensure correct compilation and linking.

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

- Run chmod 777 file.cpp to grant full permissions to the file.cpp file.
- Compile the C++ program using g++ file.cpp -o lab to create an executable named "lab".
- Execute ./lab to run the program and view the output.

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

Templates in C++ allow us to create functions or classes that can work with any data type. This eliminates the need to rewrite the same code for different types, such as int, float, and others. In simple terms, templates make our code more flexible and reusable by enabling type-independent programming.



```
# include <iostream>
using namespace std;

template <typename T>
T add (T a, T b){
return a+b;
}

int main (){
cout << add(5,10) <<endl;
cout << add(4.9, 10.1) << endl;
return 0;
}
```

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

In C and C++ programming, file extensions like .c for C and .cpp for C++ are important because they inform the compiler about the type of code being handled. These extensions help the compiler distinguish between C and C++ source files, ensuring the correct compilation and linking processes..

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

- **Syntax Error:**
  - **Cause:** Missing semicolons, unmatched brackets**.**
  - **Solution:** Recheck the code thoroughly and correct any syntax mistakes, such as adding missing semicolons or fixing unmatched brackets.
- **Missing Libraries:**
  - **Cause:** Using functions like cin and cout without including the necessary library (e.g., iostream).
  - **Solution:** Ensure that all required libraries are included at the beginning of the program (e.g., #include <iostream>).
- **Linker Error:**
  - **Cause:** A function is declared but not defined**.**

- **Solution:** Verify that all declared functions have corresponding definitions and are properly linked**.**

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

In Linux, we can manage file permissions using the chmod command, which controls who can read, write, and execute a file. Permissions are set for three categories of users: the owner, the group, and others. By using chmod, you can specify which of these users can perform actions like reading, writing, or executing the file. For example, chmod 755 grants the owner full permissions (read, write, execute) while allowing the group and others to only read and execute the file.

Q9: What is 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 an archive file created using the tar command, which was one of the first file formats introduced for Linux. To extract the contents of a tarball, we use the untar command or tar -xvf. The advantages of tarballs include compressing multiple files and directories into a single file, reducing their size for faster downloads, and making it easier to distribute files. However, there are some limitations, such as tarballs not indicating the version of the software they contain, requiring users to manually find, install, and update any additional libraries or tools. Additionally, tarballs don't provide information about file locations or software dependencies.

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

When we install an RPM (Red Hat Package Manager) package, it automatically resolves and installs the required libraries or tools, making dependency management easier. RPM also simplifies software uninstallation, addressing one of the key shortcomings of tarballs, which require manual handling. Additionally, RPM provides valuable information, such as the software version, a list of files included in the package, and details on where those files will be or are installed. In contrast, tarballs lack this level of detail and automation.