**222473**

**DIKE FAVOUR ONYEDIKACHUKWU**

**CSC 235 (OBJECT ORIENTED PROGRAMMING)**

**GITHUB LINK:** [**https://github.com/Dikedevs/222473-C-Project-1**](https://github.com/Dikedevs/222473-C-Project-1)

1. Write briefly on Unix operating system especially Linux flavor.

**OVERVIEW OF THE UNIX OPERATING SYSTEM**

The UNIX operating system was designed to allow a number of programmers access the computer simultaneously and share its resources.

The operating system coordinates the use of the computer's resources, allowing one person, for example, to run a spell check program while another creates a document, lets another edit a document while another creates graphics, and lets another user format a document -- all at the same time, with each user oblivious to the activities of the others.

The operating system controls all of the commands from all of the keyboards and all of the data being generated, and permits each user to believe he or she is the only person working on the computer.

This real-time sharing of resources makes UNIX one of the most powerful operating systems ever.

Although UNIX was developed by programmers for programmers, it provides an environment so powerful and flexible that it is found in businesses, sciences, academia, and industry. Many telecommunications switches and transmission systems also are controlled by administration and maintenance systems based on UNIX.

While initially designed for medium-sized minicomputers, the operating system was soon moved to larger, more powerful mainframe computers. As personal computers grew in popularity, versions of UNIX found their way into these boxes, and a number of companies produce UNIX-based machines for the scientific and programming communities.

**LINUX**

Linux can freely be qualified as an Internet phenomenon. Created as a UNIX-based operating system under the GNU license two decades ago, Linux has set high standards in the release of open-source technologies that master the Internet nowadays. Due to its ground-breaking concept and support by a large community of developers, Linux underwent a truly explosive growth to turn into one of the most widely adopted UNIX-like OSs in the world. The fact that it is used as both a desktop operating system and a server platform make it accessible to millions of users online.

## Linux, Unix and GNU

By combining a lot of key characteristics such as fastness, stability, security and adaptability, Linux has become the most preferred alternative to commercial Unix and Microsoft operating systems. Its proven quality of performance has been given high credits by users and explains why most of the World Wide Web today is built upon the Linux software platform.

Based on the Unix core, originally developed back in the 1960s, Linux retains its modular concept, which makes it very stable and easily customizable. When GNU was initially started as a project in 1985, it aimed to provide a complete Unix-compatible software system. However, until the 1990s, work on the kernel and the drivers were in the initial stage. This motivated Linus Torvalds to create his own kernel in 1991 using the libraries and utilities from the GNU project, laying the foundations of the GNU/Linux project and creating the core of one of the most powerful operating systems.

## Linux advantages

One of the most valued advantages of Linux over the other platforms lies with the high security levels it ensures. Every Linux user is happy to work in a virus-free environment and use the regular virus-prevention time needed when working with other operating systems for other more important tasks.

Thanks to its open-source distribution, Linux is being constantly developed and updated by the constantly expanding community of programmers supporting it. Despite its dynamic nature, it is totally complete in terms of functionality and interface. All those ongoing development efforts are made with the sole purpose of keeping the platform flexible and ever adaptable to the changeable 'climate' of the WWW.

2. Write a short note on software functional requirements.

One of the classifications of software system requirements is **functional requirements**. Broadly speaking, functional requirements answer the question, “What should the software system do?” In technical terms, functional requirements are defined as follows:

* Statement of services and functionalities that the system should provide.
* Definition of the behavior of the system in response to particular situations.
* Definition of the reaction of the system to particular inputs.
* In relevant cases, states explicitly what the system should not do.

Example

The above diagram describes the functional requirements of a monitor. The functionalities of the monitor, such as turning on/off, brightness control, hue control, sleep, displaying the input, and saturation control, fall under functional requirements. These are services that the monitor system provides to the user.

Requirements such as performance, security, interface, and portability fall under **non-functional requirements**, which are constraints on the services/functions provided by the monitor system.

As clean as these definitions of functional and non-functional requirements may sound, in reality, the distinction between two different requirements may not be as clear-cut.

For example, the security requirement seems to be a non-functional requirement. However, when it is developed in detail, this requirement may generate other requirements that could be functional. For instance, to provide security, you add a user authentication system, which turns out to be a functional requirement. Therefore, requirements may not be independent of each other.

3. Why is Unix often preferred at some points?

Every programmer needs to ask themselves several questions about their development environment which, when correctly answered, can make the job much easier in the long run. One of the questions that need to be considered is which operating system to use for programming. While certainly not the only choice, Linux is a fantastic platform for programmers. Programmers prefer Linux for its versatility, security, power, and speed.

For example, to build their own servers. Linux can do many tasks similar or in specific cases better than Windows or Mac OS X. (For example, good management of memory consumption in Linux Web Servers, security networks, etc.) This is also one of the reasons for why it is preferred more. Customization and Unix compatible environment is also the main advantage of Linux. Intermediate and advanced programmers prefer comfortable and hack-able system and these features are all available in Linux. Several other important features of Linux, that is preferred by most of the programmers, are built-in package manager, Strong Unix paths for files (not backspaces), strict case sensitive for files and directories, availability of multiple destroy without restrictions of copyrights, big and awesome community around the world, continuous feedback and many more. Most importantly, Linux is Free and Open Source which is obviously one of its most amazing features liked by everyone. Similar many other cool reasons make Linux to be preferred instead of Windows and Mac OS.

4. Why is UNIX referred to as a scientist OS?

We use the operating system in our daily life. Every mobile and computer run by the operating system. There are various types of operating systems such as Windows, macOS, Linux, Ubuntu, UNIX, Android, iOS, and blackberry OS. Unix is often referred to as a scientist OS due to some of the following reasons;

**Memory usage:**

UNIX use less memory while running sophisticated programs. UNIX OS can handle virtual memory nicely. The virtual memory expands as more programs come into the main memory. Most of the tasks in UNIX is done by using fewer resources.

**Less code to execute:**

In GUI (graphical user interface), we sometimes need many mouse clicks to perform some specific task but in the case of UNIX we can simply write one command in CLI (command line interface) and that task is done.

**Safe and secure:**

UNIX provides a safe and secure platform in which multiple users can interact with the servers online without any security issues. The interaction with the UNIX servers is fast and without any bugs. UNIX uses UID and GID for controlling permissions for users and files are accessed by users through these permissions

**User interactions:**

There are many online servers, electric machines where user interaction is not involved. UNIX is an ideal choice for machines and devices where no or fewer user interactions are involved. UNIX can manage the automatic working of systems very well.

**Supports multiple users:**

In UNIX every user needs a username password to use the OS. Every file is protected from unauthorized use. Multiple users can log in to the system and use the OS as they need. You can take the example of RDP (Remote Desktop Protocol) server and VPS (Virtual Private Server). In RDP and VPS multiple users can log in to the system and every login gives private files access to the user. Every user has a user access control system by which they can access the files securely. All the users can open many apps at the same time and there are very few chances that your system may crash. The owner of the system can set the permission level to all the users and then any user can access the files as set by the owner.

**Multitasking:**

You can open many programs in the UNIX OS and all the programs work in parallel using multiprocessor technology.

**Modular:**

The utilities for UNIX are made in modular form. If you are a programmer then you can make small programs in modular form and then unite the modules and all the modules will work consistently throughout the system.

**Human readable source code:**

As most of the code is written in C language and is available as open-source so any user can see and understands the code. The source code is written in the English language. I know English words are different from source code but if you understand the basics of programming and also understand English then you can easily follow through with the code and change it.

**Hiring experts:**

As UNIX is more than 50 years old so there are many expert developers available that can help you to sort out your problem in the OS. The demand for UNIX developers is also high because most cloud-based applications run on UNIX. Online servers and mobile OS like Android is also developed in UNIX.

Some other features of UNIX include:-

* UNIX is free
* The file system is hierarchical by which accessing and retrieving files become easy
* The performance of UNIX is better than Windows NT
* Stable database access
* Better handling of internet and intranet in servers
* Internet-client and file server are better managed using Java in UNIX

5. What type of Programming Language is C?

C is an imperative procedural language supporting structured programming, lexical variable scope, and recursion, with a static type system. It was designed to be compiled to provide low-level access to memory and language constructs that map efficiently to machine instructions, all with minimal runtime support.

6. Give the detailed structure of a complete C programming language.

Preprocessor Commands

Functions

Variables

Statements & Expressions

Comments

The structure of a C program means the specific structure to start the programming in the C language. Without a proper structure, it becomes difficult to analyse the problem and the solution. It also gives us a reference to write more complex programs.

C language combines the power of a **low-level language** and a **high-level language**. The low-level languages are used for system programming, while the high-level languages are used for application programming. It is because such languages are flexible and easy to use. Hence, C language is a widely used computer language.

It supports various operators, constructors, data structures, and loop constructs. The features of C programming make it possible to use the language for system programming, development of interpreters, compilers, operating systems, graphics, general utilities, etc. C is also used to write other applications, such as **databases, compilers, word processors,** and **spreadsheets**. a Program for Beginners

The essential features of a C program are as follows:

**Pointers:** it allows reference to a memory location by the name assigned to it in a program.

**Memory allocation:** At the time of definition, memory is assigned to a variable name, allowing dynamic allocation of the memory. It means that the program itself can request the operating system to release memory for use at the execution time.

**Recursion:** When a function calls itself, it is known as recursion.

**Bit-manipulation:** It refers to the manipulation of data in its lowest form. It is also known as bits. The computer stores the information in binary format (0 and 1).

Let's start with the importance of specifying the structure of a C program.

7. How can I create a C programming file on the OS?

## STEP 1. Write and save the program

To write the source code of your first C program you need to open the Notepad++ text editor. The quickest way to do that in Windows 10 is to hit your **Win** key, type Notepad++ in the search window, and hit **Enter**.

Once you get the text editor running, copy the following piece of C source code:

*/\* hello.c*

*A first program in C \*/*

#include <stdio.h>

int main()

{

printf("Hello Your name\n");

printf("Welcome to C Program\n");

**return** 0;

}

and paste it into the editor.

Yes, this is your first C program! Now you can save the file somewhere, choosing C source file in the Save as type drop-down menu, and naming it hello.c. You must carefully note the location of the saved file, it should be something like C:/Users/b1234567/C Program/hello.c, where b1234567 is your student number.

Once the first program has been written, we want to find out what it actually does. We need to make the computer to execute it.

To do that, we need to translate out program into the form computers understand and execute. In the next step we will use the GCC compiler to get an executable file.

## STEP 2. Open Cygwin Terminal

We will be using the GCC compiler as a command line application, so we need to access the command line (also known as terminal).

You can do that in a similar way you did with Notepad++: hit you **Win** key and type Cygwin. The first option you get is likely to be the Cygwin64 Terminal --- this is what you need.

## STEP 3. Navigate to your program with Cygwin Terminal

Now you need to find your program. Remember you had to take note of its location? Let's assume its location is C:/Users/b1234567/C Program/hello.c. To compile and execute your program you need to get to the directory C:/Users/b1234567/C Program/ by changing your current directory with the cd (stands for change directory) command.

cd C:/Users/b1234567/C Program/

You can check where you are with the pwd (print working directory) command:

pwd

If everything is all right, you'll get this:

/cygdrive/c/Users/b1234567/C Program/

You can browse the contents of the directory to make sure your source code file is here with the ls (list files) command:

ls

The result should be similar to this:

hello.c

## STEP 4. Compile the program to get the executable file

To compile your program and get an executable, you need to run the GCC compiler supplying the source file name and the output executable file name. You can do so with the following command:

gcc hello.c -o hello.exe

The command should silently exit and create the executable file hello.exe containing the compiled program ready for execution.

## STEP 5. Run the executable

To run the executable, simply type its name adding characters ./ in front of it, like this:

./hello.exe

This should result in the following lines appearing in the terminal:

Hello Your name

Welcome to C Program