Software engineering Module 2 – Introduction to programming with C lang

1. Overview of C programming

Theory Exercise:

History:

The C programing language was developed in early 1970s by Dennis Ritchie, it has become one of the most influential programming languages in history. Ritchie's primary goal was to create a language suitable for developing the UNIX operating system. By 1973, UNIX was rewritten in C marking one of the first major operating systems t be implemented in a high-level language. This showed C's capability to combines low-level machine control with the flexibility of a higher-level language.

Evolution

C was standardize in 1989 by the American National Standards Institute (ANSI) which ensured C's compatibility across different platforms and compliers. Because of this standardization C started to gain popularity globally. In the following years C went through many updates which introduced features like inline functions, variable length arrays, and better support for multithreading and concurrency.

Importance

C's importance is because of its versatility and influence. It is the foundation of many modern programming languages like c++, c#, objective-c and Java. Also its close relation with hardware makes it ideal for developing operating systems, embedded systems and performance-critical applications. Also its simplicity and efficiency makes it a strong option for learning programming fundamentals. Its syntax encourage for a deeper understanding of memory management, data structures and algorithm design making a strong base for new developers.

Why is It Still Used Today

Even of the introduction of new programming languages, C remains as a major part in the tech world. It isn't replaceable in systems programming where precise hardware control and performance are required. Applications such as operating systems, real-time systems and embedded devices often rely on C due to its minimal runtime and direct access to hardware. Also, programs written is C can be compiled and run on almost any machine this paired with vast availability of libraries and tools also ensures that it continues to be relevant for software development tasks.

Lab Exercise: Applications where C programming is extensively used are: -

- Linux kernel – Linux is an operating system and its kernel enables management of hardware resources.
- MySQL – It is the most popular database management system.
- VLC – It is a popular open-source media player

2. Setting Up Environment

Theory Exercise:

Steps to install a C compiler and set-up an Integrated Development Environment (IDE) are: -

- Search ‘MinGW’ on google and download the MinGW setup application
- Open the exe file and complete the set-up
- Open MinGW installer and select the ‘*mingw32-base*’ and ‘*mingw32-binutils*’ files to download.
- Find the path of MinGW folder in the C drive
- Open that folder and then open the bin folder
- Copy the path on the bin folder
- Now open ‘*Edit system environment variables’* application
- Click on Environment variables
- Now select the Path variable and click on edit
- Now click on New and then paste the path you copied before
- Open VS code and install C/C++ extension (by Microsoft) and code runner extension by (Jun Han)
- The set-up is done

3. Basic Structure of a C Program

Theory Exercise: Basic structure of a C program looks like: -

```
#include <stdio.h>

Main()
```

```
{
        // comment
        /*
        Multiline comment
        */
        Block of code (variables, functions, loops, etc)
}
```

Here,

# - is the preprocessor

Include – Is a keyword that tells to add a library

<> - Inside we write the name of the library which we want to include

Stdio.h – It is a library which provides the standard input and output functions

main() – This is function which starts the execution of the code

{} – Inside this we write the block of code which will execute in the function

// - This used to write single line comments

*/

*/ - This is used to write multi-line comments

Data types – These are the types of data which we can use in C lang. There are two types of data types:-

- Primitive data types – these are the data types which are pre-built into C. Example: - integer, float, long, double, char.
- Non-Primitive data types – These are the data types which are defined by the user. Example: - string, struct.

Variables – means to vary. These help us to store data. There are few rules for naming variables, which are: -

- It should start with alphabets
- It shouldn’t contain spaces
- It can have numbers in it, but not in the starting
- It shouldn’t contain special characters

Examples of valid variable names: x, y, num1, first_name, etc.

4. Operators in C

Theory Exercise: Types of operators in C are: -

- Arithmetic operator – These are used do basic arithmetic functions like addition, subtraction, division, and multiplication.
- Relational operator – These are used to find the relation between two variables. For example: <, >, <=, >=, ==.
- Assignment operators – These are used to assign values to variables. For example; =, *=, +=, -=, /=.
- Increment/ Decrement operators – These are used to increase or decrease the value of variable by 1. For example: --a, a--, ++a, a++.z
- Bitwise operators – These **NOT DONE**
- Conditional operators – These are used to run different part of code basic on conditions. For example: - if, if … else, if …. else if …. Else, nested if.

5. Control Flow Statements in C

Theory Exercise: Decision making statement are used to perform different actions based on conditions. Decision making statements in C are: -

- If – It is used when only a single condition is to be evaluated. For example: If a student has scored more the 40% he passes.
- If … else – It is used when a single condition is evaluated on it being true or false. For Example: If a student scored more than 40% he passes else he fails.
- If … else if … else – it is used when multiple condition are to be evaluated. For example: Deciding student's grade (A+, A, B+, B, C or fail) on basis of their score.
- Nested if – It is used when you need to evaluate a condition after evaluating some other conditions. For example:
- Switch case – It is an alternative to if…else if…else ladder but it can be used only with integer and character data types. For example: Any menu driven program.

6. Looping in C

Theory Exercise:

While loops – These are used when the conditions change dynamically or when you don't know the number of iterations required.

For loops – These are used when the number of iterations are known and when you are using a counter.

Do-while loops – These are used when the loop will execute atleast once, like for menu-driven programs.

7. Loop Control Statements

Theory Exercise:

Break – It is used to end the loop if a particular condition is true

Continue – It is used to skip the iteration when a condition is true

Goto – It is used to create loops without using loops

8. Functions in C

Theory Exercise: Functions are a block of code. Explanation of functions declaration, definition and calling with example: -

Declaration: It is declared before the main function and after the header files (libraries)

Ex:-

```
// functions_return_type <function_name>(data_type(of the input variables));

void func();
```

Definition: The function is defined after the main function at the end of the code.

Ex:-

```
// functions_return_type <function_name>(data_type(of the input variables) <variable_name>)

{

        //block of code

}
```

Calling: The function in called within the main function.

Ex:-

```
main(){
        func();

}
```

9. Arrays in C

Theory Exercise: Arrays are collection of same kind of data, it is useful as we don't need use multiple variables to store the data. For example – To store scores of multiple subjects instead of using different variable for each subject we can create an array of the scores and access score of a particular subject with help of the index of the array.

One dimensional array store data in a list like format (like to store marks of a single student) whereas in multi-dimensional array the data can be store in form of tables (like to store data of multiple students).

10. Pointers in C

Theory Exercise: Pointers are variables which hold the memory address of another variable. This help us to access variables by reference instead of value and also makes file handling possible.

Declare -

Int *<variable_name>;

Initialize -

<pointer_variable> = &<variable_name> // the name variable who's address is to be stored.

11. Strings in C

Theory Exercise:

strlen():- It returns the length of string as a integer. It is can be used to iterate over the string's index.

strcpy():- It copies one string into another. It takes two arguments, first one is the empty string where the data is to be copied and second one from which the data is to be taken.

strcat():- It join two strings into one string. It takes two arguments, the resulting string will be stored in the variable passed as the first argument.

strcmp():- It compares the value of two strings. It takes two arguments and compare them and returns value 0 if the strings are equal or 1 if the strings aren't equal.

strchr():- It is

12. Structures in C

Theory Exercise: Structures are collection of different kind of data.

Example to showcase how to declare and initialize a structure and access its members: -

Declaration:

```
struct students {
int sid;
char sname[20];
int roll_no;
float percent;
};
```

Initialize:

```
struct students s;
printf(“\nEnter student ID : “);
scanf(“%d”&s.sid);
printf(“\nEnter student’s name : “);
scanf(“%s”, &s.sname);
printf(“\nEnter student’s roll number : “);
scanf(“%d”, &s.roll_no);
printf(“\nEnter student’s marks : “);
scanf(“%f”, &s.percent);
```

Accessing:

```
printf(“\students name : %s”, s.sname);
```

13. File Handling in C

Theory Exercise: File handling is important as while running a program the variables which store data only store that data until the program is running, to store that permanently we need to store that data in a file. How to perform different file functions: -

Opening: - to open a file we first need to declare a pointer of type FILE which holds the address of the file and then we can open the file using fopen() function.

Syntax – pointer_variable = fopen(<file_name>, <file_mode>)

Closing: - To close a file, we use the fclose() function.

Syntax – fclose(<file_name>)

Reading: - To read from a file we use fgets() function which takes a variable to store data in, size of the string to be read and file pointer as it's arguments.

Writing: - To write into a file we fputs() function which takes the variable from which the data is to be written into the file and the file pointer as arguments.