1.#include <stdio.h>

Struct Student {

Char regNumber[20];

Char firstName[50];

Char lastName[50];

Float programming;

Float networking;

Float accounting;

Float computation;

Float operatingSystem;

};

Int main() {

Struct Student student;

// Input student details

Printf(“Enter registration number: “);

Scanf(“%s”, student.regNumber);

Printf(“Enter first name: “);

Scanf(“%s”, student.firstName);

Printf(“Enter last name: “);

Scanf(“%s”, student.lastName);

Printf(“Enter programming marks: “);

Scanf(“%f”, &student.programming);

Printf(“Enter networking marks: “);

Scanf(“%f”, &student.networking);

Printf(“Enter accounting marks: “);

Scanf(“%f”, &student.accounting);

Printf(“Enter computation marks: “);

Scanf(“%f”, &student.computation);

Printf(“Enter operating system marks: “);

Scanf(“%f”, &student.operatingSystem);

// Display student details

Printf(“\nStudent Details:\n”);

Printf(“Registration Number: %s\n”, student.regNumber);

Printf(“First Name: %s\n”, student.firstName);

Printf(“Last Name: %s\n”, student.lastName);

Printf(“Programming Marks: %.2f\n”, student.programming);

Printf(“Networking Marks: %.2f\n”, student.networking);

Printf(“Accounting Marks: %.2f\n”, student.accounting);

Printf(“Computation Marks: %.2f\n”, student.computation);

Printf(“Operating System Marks: %.2f\n”, student.operatingSystem);

Return 0

2.Define the following terms as used in C programming

* Compiler: A compiler is a software tool that translates high-level programming language code (source code) written by a programmer into machine code or an intermediate code that can be executed directly by a computer's CPU
* Source Code: Source code refers to the human-readable, written instructions that a programmer creates using a high-level programming language. It is the original code that is written before it is transformed into machine code or an executable program.
* Object Code: Object code is the output produced by a compiler after it translates the source code. It is a binary representation of the program that is not human-readable. This code is specific to the target computer architecture and needs further processing to create an executable file.
* Linkers: Linkers are tools that take one or more object files generated by a compiler and combine them into a single executable program. They resolve references between different object files, ensuring that functions and variables declared in one file can be used in another, and they produce the final executable file that can be run on the computer.

3.Using an example e.g a program to add two numbers explain the compilation process in C

Preprocessing: The preprocessor handles directives like #include and performs text substitution. It includes the contents of the standard input/output library (<stdio.h> in this case).

Compilation: The compiler translates the preprocessed source code into assembly code or an intermediate representation.

Assembly: The assembler converts the assembly code into machine code or object code specific to the target architecture.

Linking: The linker takes the object code from the compilation and links it with the necessary library functions (like those from stdio.h). It resolves symbols and produces the final executable file.

4.Explain the difference between a compiler and an interpretor

* Translation Process:Compiler: Translates the entire source code into machine code or an intermediate code before execution.Interpreter: Translates the source code line by line and executes it immediately without creating a separate compiled file.
* Execution Speed:Compiler: Generally produces faster-executing code because the entire program is translated before execution.Interpreter: Typically slower than compiled code as it translates and executes code line by line.
* Generated Code:Compiler: Generates an executable file or object code, which is independent of the source code and can be executed repeatedly.Interpreter: Does not generate a separate executable file; it interprets the source code each time it runs.
* Debugging:Compiler: Can make debugging more challenging since errors are often reported after the entire code is translated.Interpreter: Allows for easier debugging as errors are reported immediately after encountering the problematic code.
* Memory Usage:Compiler: Generally requires more memory as it creates a compiled version of the entire program.Interpreter: Tends to use less memory since it processes the code line by line.
* Portability:Compiler: The compiled code is specific to the target machine architecture, potentially requiring recompilation for different platforms.Interpreter: Generally more portable since it can interpret source code directly, without the need for a platform-specific compiled version.

5.List all the main categories of operator available in C programming

* Arithmetic Operators:+ (Addition)- (Subtraction)\* (Multiplication)/ (Division)% (Modulus)
* Relational Operators:== (Equal to)!= (Not equal to)< (Less than)> (Greater than)<= (Less than or equal to)>= (Greater than or equal to)
* Logical Operators:&& (Logical AND)|| (Logical OR)! (Logical NOT)
* Bitwise Operators:& (Bitwise AND)| (Bitwise OR)^ (Bitwise XOR)~ (Bitwise NOT)<< (Left shift)>> (Right shift)
* Assignment Operators:= (Assignment)+= (Add and assign)-= (Subtract and assign)\*= (Multiply and assign)/= (Divide and assign)%= (Modulus and assign)&= (Bitwise AND and assign)|= (Bitwise OR and assign)^= (Bitwise XOR and assign)<<= (Left shift and assign)>>= (Right shift and assign)
* Increment and Decrement Operators:++ (Increment)—(Decrement)
* Conditional Operator (Ternary Operator):? : (Conditional expression)
* sizeof Operator:sizeof (Returns the size, in bytes, of a data type or object)