An***array***is defined as the collection of similar type of data items stored at contiguous memory locations. Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc.***The break statement*** terminates the execution of the nearest enclosing do , for , switch , or while statement in which it appears. Control passes to the statement that follows the terminated statement.***To make an infinite loop***, just use true as your condition. true is always true, so the loop will repeat forever. Warning: Please make sure you have a check that exits your loop, otherwise it will never end.In C programming***, scanf()*** is one of the commonly used function to take input from the user. The scanf() function reads formatted input from thestandard input such as keyboards.***A null pointer*** is a pointer which points nothing. Some uses of the null pointer are: a) To initialize a pointer variable when that pointer variable isn't assigned any valid memory address yet. b) To pass a null pointer to a function argument when we don't want to pass any valid memory address.The ***difference between Actual Parameters and Formal Parameters*** is that Actual Parameters are the values that are passed to the function when it is invoked while Formal Parameters are the variables defined by the function that receives values when the function is called.***C functions*** are used to avoid rewriting same logic/code again and again in a program. There is no limit in calling C functions to make use of same functionality wherever required. We can call functions any number of times in a program and from any place in a program.***Storage Classes*** are used to describe the features of a variable/function. These features basically include the scope, visibility and life-time which help us to trace the existence of a particular variable during the runtime of a program.C language uses 4 storage classes, namely:***auto***: This is the default storage class for all the variables declared inside a function or a block. Hence, the keyword auto is rarely used while writing programs in C language. Auto variables can be only accessed within the block/function they have been declared and not outside them (which defines their scope). Of course, these can be accessed within nested blocks within the parent block/function in which the auto variable was declared***. extern***: Extern storage class simply tells us that the variable is defined elsewhere and not within the same block where it is used. Basically, the value is assigned to it in a different block and this can be overwritten/changed in a different block as well. So an extern variable is nothing but a global variable initialized with a legal value where it is declared in order to be used elsewhere. It can be accessed within any function/block. Also, a normal global variable can be made extern as well by placing the 'extern' keyword before its declaration/definition in any function/block. ***static***: This storage class is used to declare static variables which are popularly used while writing programs in C language. Static variables have the property of preserving their value even after they are out of their scope! Hence, static variables preserve the value of their last use in their scope. So we can say that they are initialized only once and exist till the termination of the program. Thus, no new memory is allocated because they are not re-declared.***register***: This storage class declares register variables that have the same functionality as that of the auto variables. The only difference is that the compiler tries to store these variables in the register of the microprocessor if a free registration is available. This makes the use of register variables to be much faster than that of the variables stored in the memory during the runtime of the program***. Type casting*** refers to changing an variable of one data type into another. The compiler will automatically change one type of data into another if it makes sense. For instance, if you assign an integer value to a floating-point variable, the compiler will convert the int to a float.***logical operators***In this program, operators (&&, || and !) are used to perform logical operations on the given expressions.&& operator – "if clause" becomes true only when both conditions (m>n and m! =0) is true. Else, it becomes false.|| Operator – "if clause" becomes true when any one of the condition (o>p || p!=20) is true. It becomes false when none of the condition is true.! Operator – It is used to reverses the state of the operand.If the conditions (m>n && m!=0) is true, true (1) is returned. This value is inverted by "!" operator.So, "! (m>n and m! =0)" returns false (0).***Static memory allocation*** is an allocation technique which allocates a fixed amount of memory during compile time and the operating system internally uses a data structure known as Stack to manage this***.An Exit Control*** Loop checks the condition for exit and if given condition for exit evaluate to true, control will exit from the loop body else control will enter again into the loop.Such type of loop controls exit of the loop that's why it is called exit control loop.Exit Control loopDo…While loop:Do …while loop is an exit control loop.It is a variation of while loop.When we want to execute or perform some task at least for once, we can use do…while loop structure.In a while loop if the expression becomes false at very first try then loop will never be executed.Since, do…while is of type exit control loop it checks the condition at last so, first time the loop will be execute unconditionally.If the condition for the exit becomes true then loop will be terminate otherwise it will be executed for the next time***.C program to reverse a number***#include<stdio.h>int main() { int n, reverse = 0, remainder; printf("Enter an integer: "); scanf("%d", &n); while (n != 0) { remainder = n % 10; reverse = reverse * 10 + remainder; n /= 10; } printf("Reversed number = %d", reverse); return 0;}.***A "For" Loop*** is used to repeat a specific sblock of code a known number of times. For example, if we want to check the grade of every student in the class, we loop from 1 to that number. When the number of times is not known before hand, we use a "While" loop.***Unknown number of times:***"Ask the User to Guess a pre-determined number between 1 and 100". You have no way of knowing how many guesses it will take."Randomly look in an array for a given value." You have no way of knowing how many tries it will take to find the actual value.***Known number of times:***Compute the average grade of the class. While you (the programmer) might not know how many grades exist in the class, the computer will know. Usually this is accomplished by using the "length" function on an array.Print the odd numbers from 1 to 1001.Search a list (array) of numbers for the biggest grade. Again, the computer "knows" how many grades there are, so a for loop is appropriate***.Opening a file***The fopen() function is used to create a file or open an existing file:fp = fopen(const char filename,const char mode);There are many modes for opening a file:r - open a file in read mode w - opens or create a text file in write mode a - opens a file in append mode r+ - opens a file in both read and write mode a+ - opens a file in both read and write mode w+ - opens a file in both read and write mode.Now you might be thinking, "This just prints text to the screen. How is this file IO?"The answer isn't obvious at first, and needs some understanding about the UNIX system. In a UNIX system, everything is treated as a file, meaning you can read from and write to it.This means that your printer can be abstracted as a file since all you do…***Call by value in C***In call by value method, the value of the actual parameters is copied into the formal parameters. In other words, we can say that the value of the variable is used in the function call in the call by value method.In call by value method, we can not modify the value of the actual parameter by the formal parameter.In call by value, different memory is allocated for actual and formal parameters since the value of the actual parameter is copied into the formal parameter.***The actual parameter*** is the argument which is used in the function call whereas formal parameter is the argument which is used in the function definition.***:*** ***A union*** is a special data type available in C that allows to store different data types in the same memory location. You can define a union with many members, but only one member can contain a value at any given time. Unions provide an efficient way of using the same memory location for multiple-purpose.***Defining a Union***To define a union, you must use the union statement in the same way as you did while defining a structure. The union statement defines a new data type with more than one member for your program. The format of the union statement is as follows –union [union tag] { member definition; member definition; member definition;} [one or more union variables]; The union tag is optional and each member definition is a normal varia…***If else statement*** Syntax of if else statement:If condition returns true then the statements inside the body of "if" are executed and the statements inside body of "else" are skipped.If condition returns false then the statements inside the body of "if" are skipped and the statements in "else" are executed.if(condition) { // Statements inside body of if}else { //Statements inside body of else}***Example of if else statement***In this program user is asked to enter the age and based on the input, the if..else statement checks whether the entered age is greater than or equal to 18. If this condition meet then display message "You are eligible for voting", however if the condition doesn't meet ***Nested If..else statement*** When an if else statement is present inside the body of another "if" or "else" then this is called nested if else.Syntax of Nested if else statement:if(condition) { //Nested if else inside the body of "if" if(condition2) { //Statements inside the body of nested "if else { //Statements inside the body of nested "else }else { //Statements inside the body of "else"}***Example of nested if..else***#include <stdio.h>int main(){ int var1, var2; printf("Input the value of var1:"); scanf("%d", &var1); printf("Input the value of var2:"); scanf("%d",&var2); if (var1 != var2) {printf("var1 is not equal to var2\n");//Nested if elseif (var1 > var2){printf("var1 is g…An ***if statement*** consists of a Boolean expression followed by one or more statements.SyntaxThe syntax of an 'if' statement in C programming language is –if(boolean_expression) { /* statement(s) will execute if the boolean expression is true */}If the Boolean expression evaluates to true, then the block of code inside the 'if' statement will be executed. If the Boolean expression evaluates to false, then the first set of code after the end of the 'if' statement (after the closing curly brace) will be executed.C programming language assumes any non-zero and non-null values as true and if it is either zero or null, then it is assumed as false value.***C if statement***ExampleLive Demo #include <stdio.h>int main () { /* local variabl…***What is a structure?*** A structure is a key word that create user defined data type in C/C++. A structure creates a data type that can be used to group items of possibly different types into a single type. ***How to create a structure?*** 'struct' keyword is used to create a structure. Following is an example. struct address{ char name[50]; char street[100]; char city[50];char state[20]; int pin;};***How to access structure elements?*** Structure members are accessed using dot (.) operator. C#include<stdio.h>struct Point{ int x, y};int main(){ struct Point p1 = {0, 1}; // Accessing members of point p1 p1.x = 20; printf ("x = %d, y = %d", p1.x, p1.y); return 0} ***IDE***C/C++ integrated development environments, or C/C++ IDEs, are software platforms that provide programmers and developers a comprehensive set of tools for software development in a single product, specifically in the C and/or C++ programming languages: ***Elements of c***Tokens.Comments.Keywords.Identifiers.Constants.String literals.Punctuation and special characters. **Tokens** are the smallest elements of a program, which are meaningful to the compiler. The following are the types of tokens: Keywords, Identifiers, Constant, Strings, Operators, etc. **Keywords** are predefined, reserved words used in

programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier.**Identifier** refers to name given to entities such as variables, functions, structures etc. Identifiers must be unique. They are created to give a unique name to an entity to identify it during the execution of the program. In C language, a number or character or string of characters is called a **constant.** And it can be any data type. Constants are also called as literals. There are two types of constants − Primary constants − Integer, float, and character are called as Primary constants. A **variable** is nothing but a name given to a storage area that our programs can manipulate. Each variable in C has a specific type, which determines the size and layout of the variable's memory; the range of values that can be stored within that memory; and the set of operations that can be applied to the variable. Main **datatypes**. The C language provides the four basic arithmetic type specifiers char, int, float and double, and the modifiers signed, unsigned, short, and long. The following table lists the permissible combinations in specifying a large set of storage size-specific declarations. A **variable declaration** is useful when you are using multiple files and you define your variable in one of the files which will be available at the time of linking of the program. You will use the keyword extern to declare a variable at any place. **An operator** is a symbol that tells the compiler to perform specific mathematical or logical functions. C language is rich in built-in operators and provides the following types of operators −Arithmetic OperatorsRelational OperatorsLogical OperatorsBitwise OperatorsAssignment OperatorsMisc Operators. An **arithmetic operator** performs mathematical operations such as addition, subtraction, multiplication, division etc on numerical values (constants and variables).: **A relational operator** checks the relationship between two operands.There are 3 **logical operators** in C language. They are, logical AND (&&), logical OR (||) and logical NOT (!).The result of **an assignment expression** is an lvalue. All assignment operators have the same precedence and have right-to-left associativity.: The **conditional operator** is also known as a ternary operator. The conditional statements are the decision-making statements which depends upon the output of the expression. It is represented by two symbols, i.e., '?'***. I/O*** refers to the input - output functions in C language.High level I/O Functions.getc ( ) /fgetc() read a character from a file.putw ( )write a number into a filegetw ( )read number from a filefputs ( )write a string into a file.in computer science, ***control flow*** is the order in which individual statements, instructions or function calls of an imperative program are executed or evaluated. The emphasis on explicit control flow distinguishes an imperative programming language from a declarative programming language.An ***array*** is defined as the collection of similar type of data items stored at contiguous memory locations. Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc.: In C programming, a ***string*** is a sequence of characters terminated with a null character \0 . For example: char c[] = "c string"; When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end by default.: **A one-dimensional array** (or single dimension array) is a type of linear array. Accessing its elements involves a single subscript which can either represent a row or column index.**A two-dimensional array** in C can be thought of as a matrix with rows and columns. The general syntax used to declare a two-dimensional array is: A two-dimensional array is an array of several one-dimensional arrays. Following is an array with five rows, each row has three columns: int my_array[5][3]; In C/C++, we can define **multidimensional arrays** in simple words as an array of arrays. Data in multidimensional arrays are stored in tabular form (in row-major order). The total number of elements that can be stored in a multidimensional array can be calculated by multiplying the size of all dimensions.***Modularization*** is a method to organize large programs in smaller parts, i.e. the modules. Every module has a well defined interface toward client modules that specifies how "services" provided by this module are made available.C programming language allows coders to define functions to perform special tasks. As functions are defined by users, they are called **user-defined functions.** user-defined functions have contained the block of statements which are written by the user to perform a task: A **Multi Function Program.** A function is a self-contained block of code that performs a particular task. Once a function has been designed and packed, it can be treated as a 'black box' that takes some data from the main program and returns a value. The inner details of operation are invisible to the rest of the program**: To pass an entire array** to a function, only the name of the array is passed as an argument. result = calculateSum(num); However, notice the use of [] in the function definition. This informs the compiler that you are passing a one-dimensional array to the function.: It is called inside a program whenever it is required to **call a function**. It is only called by its name in the main() function of a program. We can pass the parameters to a function calling in the main() function.: In programming terminology, a ***bit field*** is a data structure that allows the programmer to allocate memory to structures and unions in bits in order to utilize computer memory in an efficient manner. **The syntax of declaring a pointer** is to place a * in front of the name. A pointer is associated with a type (such as int and double) too. Naming Convention of Pointers: Include a " p " or " ptr " as prefix or suffix, e.g., iPtr , numberPtr , pNumber , pStudent . You need **to initialize a pointer** by assigning it a valid address. This is normally done via the address-of operator (&). The address-of operator (&) operates on a variable, and returns the address of the variable. For example, if number is an int variable, &number returns the address of the variable number.**: A character string** is often specified by enclosing the characters in single or double quotes. For example, WASHINGTON would be a name, but 'WASHINGTON' and “WASHINGTON” would be character strings. **To create a file** in a 'C' program following syntax is used, FILE *fp; fp = fopen ("file_name", "mode"); In the above syntax, the file is a data structure which is defined in the standard library. fopen is a standard function which is used to open a file.**Steps for Processing a File**Declare a file pointer variable.Open a file using fopen() function.Process the file using the suitable function.Close the file using fclose() function.**The three basic programming constructs** sequence is the order in which instructions occur and are processed.selection determines which path a program takes when it is running.iteration is the repeated execution of a section of code when a program is running.**Actual Parameter** : The variable or expression corresponding to a formal parameter that appears in the function or method call in the calling environment.**Formal parameters** are always variables, while actual parameters do not have to be variables. Parameter Written in Function Call is Called “Actual Parameter”. One can use numbers, expressions, or even function calls as actual parameters. In above, para1 is called the Formal Parameter num1 is called the Actual Parameter.**The call by reference** method of passing arguments to a function copies the address of an argument into the formal parameter. Inside the function, the address is used to access the actual argument used in the call. It means the changes made to the parameter affect the passed argument.**ssss** method of passing arguments to a function copies the actual value of an argument into the formal parameter of the function. In this case, changes made to the parameter inside the function have no effect on the argument. By default, C programming uses call by value to pass arguments.**Register variables** tell the compiler to store the variable in CPU register instead of memory. Frequently used variables are kept in registers and they have faster accessibility. We can never get the addresses of these variables. “register” keyword is used to declare the register variables.**return statement** ends the execution of a function, and returns control to the calling function. Execution resumes in the calling function at the point immediately following the call. A return statement can return a value to the calling function.**Dynamic memory allocation in C**The concept of dynamic memory allocation in c language enables the C programmer to allocate memory at runtime. Dynamic memory allocation in c language is possible by 4 functions of stdlib.h header file.malloc()calloc()realloc()free()Before learning above functions, let's understand the difference between static memory allocation and dynamic memory allocation.**malloc()**allocates single block of requested memory.**calloc()**allocates multiple block of requested memory.**realloc()**reallocates the memory occupied by malloc() or calloc() functions.**free()** frees the dynamically allocated memory.**sJump statements in** C/C++ are a type of Control Statements in C/C++ used to interrupt the normal flow of the program. It makes the program jump to another section of the program unconditionally when encountered. It can also be used to terminate any loop.