K.S.R. COLLEGE OF ENGINEERING (Autonomous)					R 2018
SEMESTER – II					
18CS211	**C PROGRAMMING**	L	T	P	C
		3	0	0	3

Prerequisite:*Fundamental knowledge in Problem Solving Techniques.*

Objectives:

- *To study the basics of C primitives, operators and expressions.*
- *To understand the different primitives and user defined data types.*
- *To learn the structural programming concepts*

UNIT - I	**FUNDAMENTALS OF C PROGRAMMING**	**[9]**

History of C: Middle level language – Structured language – Programmer's language – Compilers Vs. Interpreters – Library and Linking – Expressions: Basic Data Types – Variables – C scopes –Type qualifiers –Storage class specifiers – Variable initialization – Constants – Operators – Expressions.

UNIT - II	**STATEMENTS, ARRAY AND STRING**	**[9]**

Statements: Selection Statements – Iteration statements – Jump statements – Expression statements – Block statements. Array: Single-Dimension arrays –Two-Dimensional arrays – Multidimensional arrays – String: Declaring and Initializing String Variables – String Handling Functions and Operations.

UNIT - III	**FUNCTIONS ANDPOINTERS**	**[9]**

Function: General form of function – Understanding the scope of a function – Function arguments – Recursion. Pointers: Pointer variables – Pointer Operators – Pointer expressions – Pointers and Arrays – Indexing pointer – Multiple indirections – Initializing pointers – Pointers to functions.

UNIT - IV	**STRUCTURES, UNIONS AND CONSOLE I/O**	**[9]**

Accessing Structure Members – Structure Assignments – Arrays of Structures – Passing Structures to Functions – Structure pointers – Arrays and Structures within structures. Unions – Console I/O: Reading and Writing Characters – Reading and Writing Strings – Formatted Console I/O: printf() and scanf().

UNIT - V	**FILES AND PREPROCESSORS**	**[9]**

Files: Streams and Files – File System Basics – fread() and fwrite()– fseek() and Random-Access I/O – fprintf() and fscanf() – Command line argument. Preprocessor**:** #define, #error, #include, Conditional Compilation Directives, #undef.

Total = 45 Periods

Course Outcomes: On Completion of this course, the student will be able to

- *Understand the syntax and semantics of the C language.*
- *Demonstrate programs using control statements,arrays and strings.*
- *Use the knowledge of functions and pointers to develop solutions.*
- *Demonstrate efficient programs using structures, unions and console I/O.*
- *Apply the concepts of files and preprocessors to real world problems.*

Text Books :

1	Herbert Schildt, "C - The Complete Reference", Tata McGraw-Hill, 2013.
2	Ashok N.Kamathane, "Computer Programming", Pearson Education, 2014.

References :

1	PradipDey and ManasGhosh, "Fundamentals of Computing and Programming in C", 1st Edition, Oxford University Press, 2013.
2	Anita Goel and Ajay Mittal, "Computer Fundamentals and Programming in C", Dorling Kindersley (India) Pvt. Ltd., Pearson Education in South Asia, 2011.
3	Yashavant P. Kanetkar, "Let Us C", BPB Publications, 2011.
4	E.Balagurusamy, "Programming in ANSI C", Tata McGraw-Hill, 2012.
5	Nptel.ac.in/courses/106104128/

UNIT - I	FUNDAMENTALS OF C PROGRAMMING

2 Marks Questions and Answers

1. What is meant by Installation and Assembling?

Installation –It is the process of loading the software package into the computer. Assembling – It is the process of mounting different computer peripherals into one, to make the computer to function properly.

2. What are language translators?

The language translators are the programs which come under system software category. They are Compilers, Interpreters and Assembler.

3. What are a Compiler, Assembler and Interpreter?

Compiler: It is a program which is used to convert the high level language program into machine language.

Assembler: It is a program which is used to convert the assembly level language program into machine language.

Interpreter: It is a program; it takes one statement of a high level language program, translates it into machine language instruction and then immediately executes the resulting machine language instruction.

4.What is a linker?

A linker is a program that combines object modules to form an executable program. Many programming languages allow you to write different pieces of code, called modules, separately. This simplifies the programming task because you can break a large program into small, more manageable pieces.

Modules has to be put together. This is the job of the linker.

In addition to combining modules, a linker also replaces symbolic addresses with real addresses. Therefore, you may need to link a program even if it contains only one module.

5. What is a loader?

In computing, a loader is the part of an operating system that is responsible for one of the essential stages in the process of starting a program, loading programs, that is, starting up programs by reading the contents of executable files (executables- files containing program text) into memory, then carrying out other required preparatory tasks, after which the program code is finally allowed to run and is started when the operating system passes control to the loaded program code.

6. What is Booting?

In computing, booting (also known as "booting up") is a bootstrapping process that starts operating systems when the user turns on a computer system. A boot sequence is the initial set of operations that the computer performs when power is switched on. The boot loader typically loads the main operating system for the computer.

7. Differentiate machine language and high level language

S No	Machine language	High level Language
1	Represented in numbers.	Human readable form.
2	Directly executed by the Central Processing Unit.	Should be translated into machine code by compiler / interpreter.
3	Example: ADD A, B where A and B are operands and ADD is an opcode.	Example: C, C++

8. Difference between Compiler and Interpreter.

S No	Compiler	Interpreter
1	Executes source code into target or assembly code.	Executes source code directly or to an intermediate form.
2	Compilers convert once the source program.	Interpreter converts every time the program runs.
3	Languages for compiler conversion: C, C++.	Languages for interpreter conversion: MATLAB, Python.

9. What are the steps involved in booting?

First, the Power On Self Tests(POST) is conducted. These tests verify that the system is operating correctly and will display an error message and/or output a series of beeps known as beep codes depending on the BIOS manufacturer.

Second, is initialization in which the BIOS look for the video card built in BIOS program and runs it. The BIOS then looks for other devices' ROMs to see if any of them have BIOSes and they are executed as well.

Third, is to initiate the boot process. The BIOS looks for boot information that is contained in file called the master boot record (MBR) at the first sector on the disk. If it is searching a floppy disk, it looks at the same address on the floppy disk for a volume boot sector. Once an acceptable boot record is found the operating system is loaded which takes over control of the computer.

10. What are the different data types available in ‘C’?

There are four basic data types available in ‘C’.

1. int

2. float
3. char
4. double

11. What are Keywords?

Keywords are certain reserved words that have standard and pre-defined meaning in 'C'. These keywords can be used only for their intended purpose.

12. What is an Operator and Operand?

An operator is a symbol that specifies an operation to be performed on operands.

Example: *, +, -, / are called arithmetic operators.

The data items that operators act upon are called operands.

Example: a+b; In this statement a and b are called operands.

13. What is Ternary operators or Conditional operators?

Ternary operators is a conditional operator with symbols ? and :

Syntax: variable = exp1 ? exp2 : exp3

If the exp1 is true variable takes value of exp2. If the exp2 is false, variable takes the value of exp3.

14. What are the Bitwise operators available in 'C'?

&	-	Bitwise AND
|	-	Bitwise OR
~	-	One's Complement
>>	-	Right shift
<<	-	Left shift
^	-	Bitwise XOR are called bit field operators

Example: k=~j; where ~ take one's complement of j and the result is stored in k.

15. What are the logical operators available in 'C'?

The logical operators available in 'C' are

&&	-	Logical AND
||	-	Logical OR
!	-	Logical NOT

16. What is the difference between Logical AND and Bitwise AND?

Logical AND (&&)

Only used in conjunction with two expressions, to test more than one condition. If both the conditions are true the returns 1. If false then return 0.

AND (&)

Only used in Bitwise manipulation. It is a unary operator.

17. What is the difference between '=' and '==' operator?

Where = is an assignment operator and == is a relational operator.

Example:

while (i=5) is an infinite loop because it is a non zero value and while (i==5) is true only when i=5.

18. What is type casting?

Type casting is the process of converting the value of an expression to a particular data type.

Example:

int x,y;

c = (float) x/y; where a and y are defined as integers. Then the result of x/y is converted into float.

19. What is conversion specification?

The conversion specifications are used to accept or display the data using the INPUT/OUTPUT statements.

20. What is the difference between 'a' and "a"?

'a' is a character constant and "a" is a string.

21. How many bytes are occupied by the int, char, float, long int and double?

int	-	2 Bytes
char	-	1 Byte
float	-	4 Bytes
long int	-	4 Bytes
double	-	8 Bytes

22. What are the types of I/O statements available in 'C'?

There are two types of I/O statements available in 'C'.

Formatted I/O Statements
Unformatted I/O Statements

23. What is the difference between ++a and a++?

++a means do the increment before the operation (pre increment) a++ means do the increment after the operation (post increment)

Example:

```
a=5;
x=a++;          /* assign x=5*/
y=a;            /*now y assigns y=6*/
x=++a;          /*assigns x=7*/
```

24. What is a String?

String is an array of characters.

25. What is a global variable?

The global variable is a variable that is declared outside of all the functions. The global variable is stored in memory, the default value is zero. Scope of this variable is available in all the functions. Life as long as the program's execution doesn't come to an end.

26. What are the Escape Sequences present in 'C'

\n	-	New Line
\b	-	Backspace
\t	-	Form feed
\'	-	Single quote
\\	-	Backspace
\t	-	Tab
\r	-	Carriage return
\a	-	Alert
\"	-	Double quotes

27. Why header files are included in 'C' programming?

This section is used to include the function definitions used in the program.

Each header file has 'h' extension and include using '# include' directive at the beginning of a program.

28. List out some of the rules used for 'C' programming.

All statements should be written in lower case letters. Upper case letters are only for symbolic constants.

Blank spaces may be inserted between the words. This improves the readability of statements.

It is a free-form language; we can write statements anywhere between '{' and '}'.

Opening and closing braces should be balanced.

29. Define delimiters in 'C'.

Symbols	Delimiters	Use
:	Colon	Useful for label
;	Semicolon Parenthesis	Terminates Statement
( ) [ ]	Square Bracket Curly Brace	Used in expression and functions
{ }	Hash	Used for array declaration Scope of
#	Comma	statement Preprocessor directive
,		Variable Separator

30. What do you mean by variables in 'C'?

- A variable is a data name used for strong a data value.
- Can be assigned different values at different times during program execution.
- Can be chosen by programmer in a meaningful way so as to reflect its function in the program.
- Some examples are: Sum percent_1 class total

31. List the difference between float and double data type.

S No	Float	Double Float / Double
1	Occupies 4 bytes in memory	Occupies 8 bytes in memory
2	Range : 3.4 e-38 to 3.8e+38	Range : 1.7 e-308 to 1.7e+308
3	Format Specifier: % f	Format Specifier: % lf
4	Example : float a;	Example : double y; There exists long double having a range of 3.4 e -4932 to 3.4 e +4932 and occupies 10 bytes in memory. Example: long double k;

32. Differentiate break and continue statement

S No	break	continue
1	Exits from current block / loop	Loop takes next iteration
2	Control passes to next statement	Control passes to beginning of loop
3	Terminates the program	Never terminates the program

33. List the types of operators.

S No	Operators Types	Symbolic Representation
1	Arithmetic operators	= , - , * , / and %
2	Relational operators	> , < , == , >=, <= and !=
3	Logical operators	&& , || and !
4	Increment and Decrement operators	++ and –
5	Assignment operators	= , + = , - = , * = , / = , ^ = , ; = , & =
6	Bitwise operators	& , | , ^ , >> , << , and ~
7	Comma operator	,
8	Conditional operator	? :

33. . What is a Modulo Operator?

'%' is modulo operator. It gives the remainder of an integer divisio

UNIT - II	STATEMENTS, ARRAY AND STRING

2 Marks Questions and Answers

1. What is the difference between if and while statement?

if	while
(i) It is a conditional statement	(i) It is a loop control statement
(ii) If the condition is true, it executes some statements.	(ii) Executes the statements within the while block if the condition is true.
(iii) If the condition is false then it stops the execution the statements.	(iii) If the condition is false the control is transferred to the next statement of the loop.

2. What is the difference between while loop and do…while loop?

In the while loop the condition is first executed. If the condition is true then it executes the body of the loop. When the condition is false it comes of the loop. In the do…while loop first the statement is executed and then the condition is checked. The do…while loop will execute at least one time even though the condition is false at the very first time.

2. What are the types of Arrays?

1.One-Dimensional Array
2. Two-Dimensional Array
3. Multi-Dimensional Array

3. What is the use of '\0' character?

When declaring character arrays (strings), '\0' (NULL) character is automatically added at end. The '\0' character acts as an end of character array.

4. Define Strings.

Strings:

The group of characters, digit and symnbols enclosed within quotes is called as Stirng

(or) character Arrays. Strings are always terminated with '\0' (NULL) character. The compiler automatically adds '\0' at the end of the strings.

Example:

char name[]={'C','O','L','L','E','G','E','E','\0'};

The character of a string are stored in contiguous memory locations as follows:

C	O	L	L	E	G	E	\0

1000 1001 1002 1003 1004 1005 1006 1007

5. How to declare a one dimensional array?

```
data_type array_name[array_size];
```

For example:
```
int age[5];
```

6. How to initialize a one dimensional array?

Arrays can be initialized at declaration time in this source code as:

```
int age[5]={2,4,34,3,4};
```

7. How to sort an array?

Arrays often need to be sorted in either ascending or descending order. There are many well known methods for doing this; the *quick sort* algorithm is among the most efficient. This section briefly describes one of the easiest sorting methods called the *selection sort*.

The basic idea of selection sort is:

For each index position I in turn:

1. Find the smallest data value in the array from positions I to (Length - 1), where "Length" is the number of data values stored.
2. Exchange the smallest value with the value at position I.

8. Defining 2D Arrays

- type array_name [size1] [size2]
- int sales[4] [3] in above case
- As with single dimension arrays, each dimension of an array is indexed from 0 to 1-maximum size
- The first index indicates row and the second index indicates column

9. Initializing 2D Arrays

- static int table[2,3] = {1,3,5,7,9,2}
- static int table[2,3] = {{1,3,5}, {7,9,2}}

10. What is multi dimensional array?

- C allows arrays of more than two dimensions.
- Exact limit is determined by the compiler
- Size is more imp than the no of dimensions
- Survey of rainfall of last 3 years of all 12 months of five cities: int survey [3][12][5]
- View multidimensional arrays as repeated 2D arrays

11. What is String Array?

In C language Strings are defined as an array of characters or a pointer to a portion of memory containing ASCII characters. A string in C is a sequence of zero or more characters followed by a NULL '\0' character.

12. How to Reading and Writing Strings?

One possible way to read in a string is by using *scanf*. However, the problem with this, is that if you were to enter a string which contains one or more spaces, scanf would finish reading when it reaches a space, or if return is pressed. As a result, the string would get cut off. So we could use the *gets* function

A *gets* takes just one argument - a char pointer, or the name of a char array, but don't forget to declare the array / pointer variable first! What's more, is that it automatically prints out a newline character, making the output a little neater.

A *puts* function is similar to *gets* function in the way that it takes one argument - a char pointer. This also automatically adds a newline character after printing out the string. Sometimes this can be a disadvantage, so *printf* could be used instead.

13. How to String Variable Declarations and Assignments?

String variables can be declared just like other arrays:
char phrase[14];

String arrays can be initialised or partially initialised at the same time as being declared, using a list of values enclosed in "{}" braces (the same is true of arrays of other data types). For example, the statement

```
char phrase[14] = {'E','n','t','e','r',' ','a','g','e',':',' ','\0'};
```

14. Write the limitations of getchar( ) and sacnf( ) functions for reading strings.(JAN 2009)

getchar( )

To read a single character from stdin, then getchar() is the appropriate.

scanf( )

scanf( ) allows to read more than just a single character at a time.

15. What is the difference between scanf() and gets() function?

In scanf() when there is a blank was typed, the scanf() assumes that it is an end. gets() assumes the enter key as end. That is gets() gets a new line (\n) terminated string of characters from the keyboard and replaces the '\n' with '\0'.

16. What will be the output of the following C code?

```
#include <stdio.h>
int main()
{
char *str = "hello, world";
 char *str1 = "hello, world";
 if (strcmp(str, str1))
    printf("equal");
 else
    printf("unequal");
 }
```

Ouput: unequal

17. Mention the various String Manipulation Functions in C.

1 strcpy(s1, s2); Copies string s2 into string s1.

2 strcat(s1, s2); Concatenates string s2 onto the end of string s1.

3 strlen(s1); Returns the length of string s1.

4 strcmp(s1, s2); Returns 0 if s1 and s2 are the same; less than 0 if s1s2.

5 strchr(s1, ch); Returns a pointer to the first occurrence of character ch in string s1.

6 strstr(s1, s2); Returns a p

18. Define Strings. Strings:

The group of characters, digit and symbols enclosed within quotes is called as Stirng (or) character Arrays. Strings are always terminated with '\0' (NULL) character. The compiler automatically adds '\0' at the end of the strings.

Example: char name[]={'C','O','L','L','E','G','E','E','\0'};

19. Define character.

Sequences of characters are enclosed within single quotes.

20. How to represent string and Character.

Char is represented by %c. and string is represented by %s.

21. List the characteristics of Arrays.

All elements of an array share the same name, and they are distinguished form one another with help of an element number. Any particular element of an array can be modified separately without disturbing other elements.

UNIT - III	FUNCTIONS AND POINTERS

2 Marks Questions and Answers

1. What is meant by Recursive function?

If a function calls itself again and again, then that function is called Recursive function.

2. What is a Pointer? How a variable is declared to the pointer?

Pointer is a variable which holds the address of another variable.

Pointer Declaration:

datatype *variable-name;

Example:

```
int *x, c=5;
x=&a;
```

3. What are the uses of Pointers?

Pointers are used to return more than one value to the function
Pointers are more efficient in handling the data in arrays
Pointers reduce the length and complexity of the program
They increase the execution speed
The pointers saves data storage space in memory

4. What is the output of the program?

```
main() junk(int i, int j)
{ {
int i=5;j=2; i=i*j;
junk(i,j); j=i*j;
printf("\n %d %d",i,j); }
}
```

Output: 1.2, 2

5. What are * and & operators means?

'*' operator means 'value at the address'
'&' operator means 'address of'

6. What is meant by Preprocessor?

Preprocessor is the program, that process our source program before the compilation.

7. How can you return more than one value from a function?

A Function returns only one value. By using pointer we can return more than one value.

8. Is it possible to place a return statement anywhere in 'C' program?

Yes. The return statement can occur anywhere.

9. How do you define enumerated data type?

```
enum mar_status
{ single,married,widow };
enum mar_status person1,person2;
person1=married;
```

Here the person1 is assigned to value zero.

10. What is dynamic memory allocation?

Allocating the memory at run time is called as dynamic memory allocation.

11. What are the various dynamic memory allocation functions?

malloc() - Used to allocate blocks of memory in required size of bytes.
free() - Used to release previously allocated memory space.
calloc() - Used to allocate memory space for an array of elements.
realloac() - Used to modify the size of the previously allocated memory space.

12. What is the difference between an array and pointer?

Difference between arrays and pointers are as follows.

Array Pointer

1.Array allocates space automatically.
2.It cannot be resized.
3.It cannot be reassigned.
4.Size of(array name) gives the number of bytes occupied by the array.
1.Pointer is explicitly assigned to point to an allocated space.
2.It can be resized using realloc ().
3.Pointers can be reassigned.
4.Sezeof(pointer name) returns the number of bytes used to store the pointer variable.

13. What is the purpose of the function main()?

The function main () invokes other functions within it. It is the first function to Be called when the program starts execution.

Some salient points about main() are as follows:

1. It is the starting function .
2. It returns an int value to the environment that called the program.
3. Recursive call is allowed for main() also.
4. It is a user-defined function.
5. Program exection ends when the closing brace of the function main() is reached.
6. It has two arguments (a) argument count and (b)argument vector (reprensents strings passed.)
7. Any user-defined name can also be used as parameters for main() instead of argc and argv

14. What is dangling pointer?

In C, a pointer may be used to hold the address of dynamically allocated memory. After this memory is freed with the free() function, the pointer itself will still contain the address of the released block. This is referred to as a dangling pointer. Using the pointer in this state is a serious programming error. Pointer should be assigned NULL after freeing memory to avoid this bug.

15. Compare structures and unions.

Structure Union

Every member has its own memory. The keyword used is struct. All members occupy separate memory location, hence different interpretations of the same memory location are not possible. Consumes more space compared to union.All members use the same memory. The keyword used is union. Different interpretations for thesame memory location are possible. Conservation of memory is possible.

16. Is it better to use a macro or a function?

Macros are more efficient (and faster) than function, because their corresponding code is inserted directly at the point where the macro is called. There is no overhead involved in using a macro like there is in placing a call to a function. However, macros are generally small and cannot handle large, complex codingconstructs. In cases where large, complex constructs are to handled, functions are more suited, additionally; macros are expanded inline, which means that the code is replicated for each occurrence of a macro.

17. Define sscanf() and sprint() functions.

The sscanf():

This function allows to read character from a character Array and writes to another array. Similar to scanf(), but instead of reading from standard input, it reads from an array.

The sprintf():

This function writes the values of any data type to an array of characters.

18. What is the use of 'typedef''?

It is used to create a new data using the existing type.

Syntax: typedef data type name;

Example:

typedef int hours: hours hrs;/* Now, hours can be used as new datatype */

19. What is 'C' functions? Why they are used?

A function is a self-contained block (or) a sub-program of one or more statements that performs a special task when called. To perform a task repetitively then it is not necessary to re-write the particular block of the program again and again. The function defined can be used for any number of times to perform the task.

20. Differentiate library functions and User-defined functions.

Library Functions

a) Library functions are pre-defined set of functions that are defined in C libraries.

b) User can only use the function butcannot change (or) modify thisfunction.

User-defined Functions

a) The User-defined functions are the functions defined by the useraccording to his/her requirement.

b) User can use this type of function. User can also modify this function.

21. What are the steps in writing a function in a program.

a) Function Declaration (Prototype declaration): Every user-defined functions has to be declared before the main().

b) Function Callings:The user-defined functions can be called inside any functions like main(),user-defined function, etc.

c) Function Definition: The function definition block is used to define the user-defined functions withstatements.

22. Give the syntax for using user-defined functions in a program.

Syntax for using user-defined functions in a program

Syntax:

```
function declaration; function definition;
main() main()
{ {
====== ======
function calling; (or) function calling;
====== ======
} }
function definition;
```

23. Classify the functions based on arguments and return values.

Depending on the arguments and return values, functions are classified into four types.

a) Function without arguments and return values.
b) Function with arguments but without return values.
c) Function without arguments but with return values.
d) Function with arguments and return values.

24. Distinguish between Call by value Call by reference.

Call by value

a) In call by value, the value of actual agreements is passed to the formal arguments and the operation is done on formal arguments.
b) Formal arguments values are photocopies of actual arguments values.
c) Changes made in formal arguments valued do not affect the actual arguments values.

Call by reference.

a) In call by reference, the address of actual argurment values is passed toformal argument values. b) Formal arguments values are pointers to the actual argument values.
c) Since Address is passed, the changes made in the both arguments values are permanent.

25. What is a use of 'return' Keyword?

The 'return' Keyword is used only when a function returns a value.

26. **Define function**

A function is a set of instructions that are used to perform specified tasks which repeatedly occurs in the main program.

27. What is meant by user defined function?

The function defined by the users according to their requirements is called user defined functions.

2**7. Give the advantages of user defined functions**

a) The length of the source program can be reduced by dividing it into the smaller functions) By using functions it is very easy to locate and debug an error.

c) The user-defined function can be used in many locate and debug an error.

d) Functions avoid coding of repeated programming of the similar instructions.

28. How Functions works?

Whenever function is called control passes to the called function and working of the calling function temporarily stopped. When the execution of the called function is completed then a control returns back to the calling function and executes the next statement.

29. What are the points to be followed while declaring functions?

- The list of parameters must be separated by comma.
- The names of the parameters are optional but data types are must.
- If the function does not return any value, then the return type void is must.
- The data type of actual and formal parameters must match.

30. What are the three elements of user-defined functions?

Function definition: It is the process of specifying and establishing the user defined function by specifying all of its elements and characteristics.

Function declaration: Like the normal variables in a program, the function can also be declared before they defined and invoked.

Function call: The function can be called by simply specifying the name of the function, return values and parameters if presence.

31. What is meant by actual and formal parameters?

Actual Parameters: These are parameters transferred from the calling program (main program) to the called program (function).

Formal parameters: These are the parameters, transferred in to the calling function (main) from the called program (function).

UNIT - IV	STRUCTURES, UNIONS AND CONSOLE I/O

2 Marks Questions and Answers

1. Declare the Structure with an example?

```
struct name
{
char name[10];
int age;
float salary;
} e1, e2;
```

2. Declare the Union with an example?

```
union name
{
char name[10];
int age;
float salary;
} e1, e2;
```

3. What is a Structure?

Structure is a group name in which dissimilar data's are grouped together.

4. What is Union?

Union is a group name used to define dissimilar data types. The union occupies only the maximum byte of the data type. If you declare integer and character, then the union occupies only 2 bytes, whereas structure occupies only 3 bytes.

5. What is meant by Preprocessor?

Preprocessor is the program, that process our source program before the compilation.

6.How do you define enumerated data type?

```
enum mar_status
{ single,married,widow };
enum mar_status person1,person2;
person1=married;
```

Here the person1 is assigned to value zero.

7. What is meant by debugging?

Debugging is the process of locating and isolating the errors.

8. Specify any five syntax error messages.

Missing semicolon
Missing braces
Missing quotes
Improper comment characters
Undeclared variables

9. What are the pre-processor directives?

Macro Inclusion
Conditional Inclusion
File Inclusion

10. What is dynamic memory allocation?

Allocating the memory at run time is called as dynamic memory allocation.

11. What are the various dynamic memory allocation functions?

malloc() - Used to allocate blocks of memory in required size of bytes.
free() - Used to release previously allocated memory space.
calloc() - Used to allocate memory space for an array of elements.
realloac() - Used to modify the size of the previously allocated memory space.

12. Why does n++ execute than n=n+1?

The expression n++ requires a single machine instruction such as INR to carry out the increment operation whereas; n+1 requires more instructions to carry out this operation.

13. Compare arrays and structures.

Comparison of arrays and structures is as follows.

Arrays Structures

An array is a collection of data items of same data type. Arrays can only be declared. There is no keyword for arrays. An array name represents the address of the starting element. An array cannot have bit fields. A structure is a collection of data items of different data types. Structures can be declared and defined. The keyword for structures is struct. A structure name is known as tag. It is a shorthand notation of the declaration. A structure may contain bit fields.

14. Compare structures and unions.

Structure Union

Every member has its own memory. The keyword used is struct. All members occupy separate memory location, hence different interpretations of the same memory location are not possible. Consumes more space compared to union. All members use the same memory. The keyword used is union. Different interpretations for the same memory location are possible. Conservation of memory is possible.

15. Is it better to use a macro or a function?

Macros are more efficient (and faster) than function, because their corresponding code is inserted directly at the point where the macro is called. There is no overhead involved in using a macro like there is in placing a call to a function. However, macros are generally small and cannot handle large, complex coding constructs. In cases where large, complex constructs are to handled, functions are more suited, additionally; macros are expanded inline, which means that the code is replicated for each occurrence of a macro.

16. What is nested structure?

Structures can be used as structures within structures. It is also called as 'nesting of structures'.

Syntax:

```
struct structure_nm
{
        <data-type> element 1;
        <data-type> element 2;
        - - - - - - - - - - -
        - - - - - - - - - - -
        <data-type> element n;

        struct structure_nm
        {
                <data-type> element 1;
                <data-type> element 2;
                - - - - - - - - - - -
                - - - - - - - - - - -
                <data-type> element n;
        }inner_struct_var;
}outer_struct_var;
```

17. What is union?

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];
```

18. What is storage class?

A storage class defines the scope (visibility) and life time of variables and/or functions within a C Program.

There are following storage classes which can be used in a C Program

- auto
- register
- static
- extern

19. What is auto storage class?

auto is the default storage class for all local variables.

```
{
int Count;
auto int Month;
}
```

The example above defines two variables with the same storage class. auto can only be used within functions, i.e. local variables.

20. What is register storage class?

The **register** is used to define local variables that should be stored in a register instead of RAM. This means that the variable has a maximum size equal to the register size (usually one word) and cant have the unary '&' operator applied to it (as it does not have a memory location).

```
{
register int  Miles;
}
```

Register should only be used for variables that require quick access - such as counters. It should also be noted that defining 'register' goes not mean that the variable will be stored in a register. It means that it MIGHT be stored in a register - depending on hardware and implimentation restrictions.

21. What is static storage class?

The **static** is the default storage class for global variables. The two variables below (count and road) both have a static storage class.

```
   static int Count;
int Road;

{
   printf("%d\n", Road);     }
```

22. Differentiate between array and structure.

Arrays	Structure
An array is a collection of similar items.	A structure is a collection of milar data items
An array is derived data type	It is a user defined data type.
It behaves like a built in data	It must be declared and defined
An array can be increased or eased	Structure element can be added

UNIT - V	FILES AND PREPROCESSORS

2 Marks Questions and Answers

1.What is meant by files?

A file is a collection of related information that is permanently stored on the disk and allows us to access and alter the information whenever necessary.

2.Define bits and bytes?

Bits: bits known as binary digits, bits are the smallest value in a data file. Each bit value can only be a 0 or 1.

Bytes: it is provide the next step in the data file. Bytes are most commonly made up of eight bits and used to store a single character, such as a number, a letter or any other character found in a character set.

3.What are the types of files?

Sequential file, Indexed sequential file, Random access file.

4. What is the difference between printf (), sprintf (), fprintf ()?

Printf () issued to print the text or value of the variables in the screen.

Sprintf () is used to store the values in the character array or string.

Fprintf () is used to store the values of variable in the file.

5. **List some file functions**

- fopen()
- fclose()
- fgetc()
- fprintf()
- ftell()
- rewind()
- unlink()
- rename()

6. Define pre-processor in C.

The C Preprocessor is not part of the compiler, but is a separate step in the compilation process. In simplistic terms, a C Preprocessor is just a text substitution tool. We'll refer to the C Preprocessor as the CPP. Example: #define Substitutes a preprocessor macro #include Inserts a particular header from another file.

7. Define Macro in C.

A macro definition is independent of block structure, and is in effect from the #define directive that defines it until either a corresponding #undef directive or the end of the compilation unit is encountered. Its format is: #define identifier replacement.

Example: #define TABLE_SIZE 100 ;

int table1[TABLE_SIZE];

int table2[TABLE_SIZE];

8. What are conditional Inclusions in Preprocessor Directive?

Conditional inclusions (#ifdef, #ifndef, #if, #endif, #else and #elif) These directives allow including or discarding part of the code of a program if a certain condition is met. #ifdef allows a section of a program to be compiled only if the macro that is specified as the parameter has been defined, no matter which its value is. For example: 1 #ifdef TABLE_SIZE 2 int table[TABLE_SIZE]; 3 #endif

9. What you meant by Source file Inclusion in Preprocessor directive?

Source file inclusion (#include) This directive has also been used assiduously in other sections of this tutorial. When the preprocessor finds an #include directive it replaces it by the entire content of the specified file. There are two ways to specify a file to be included: 1 #include "file" 2 #include <file>

10. What is a preprocessor, what are the advantages of preprocessor?

Ans: A preprocessor processes the source code program before it passes through the compiler.

1- a preprocessor involves the readability of program

2- It facilitates easier modification

3- It helps in writing portable programs

4- It enables easier debugging

5- It enables testing a part of program

6- It helps in developing generalized program

11. What are the facilities provided by preprocessor?

Ans: 1-file inclusion

2-substitution facility

3-conditional compilation

12. What are the two forms of #include directive?

Ans: 1.#include‖filename‖

2.#include

the first form is used to search the directory that contains the source file.

If the search fails in the home directory it searches the implementation defined locations.

In the second form ,the preprocessor searches the file only in the implementation defined locations.

13. How would you use the functions randomize() and random()?

Ans: Randomize() initiates random number generation with a random value.

Random() generates random number between 0 and n-1;

14. What do the functions atoi(), itoa() and gcvt() do?

Ans: atoi() is a macro that converts integer to character.

itoa() It converts an integer to string

gcvt() It converts a floating point number to string

15. How would you use the functions fseek(), freed(), fwrite() and ftell()?

Ans: fseek(f,1,i) Move the pointer for file f a distance 1 byte from location

i. fread(s,i1,i2,f) Enter i2 dataitems,each of size i1 bytes,from file f to string s.

fwrite(s,i1,i2,f) send i2 data items,each of size i1 bytes from string s to file f.

ftell(f) Return the current pointer position within file f.

The data type returned for functions fread,fseek and fwrite is int and ftell is long int.

16. What is the difference between the functions memmove() and memcpy()?

Ans: The arguments of memmove() can overlap in memory. The arguments of memcpy() cannot.

17. What is a file?

Ans: A file is a region of storage in hard disks or in auxiliary storage devices.It contains bytes of information .It is not a data type.

18.what are the types of file?

Ans: Files are of two types 1-high level files (stream oriented files) :These files are accessed using library functions 2-low level files(system oriented files) :These files are accessed using system calls

19.what is a stream?

Ans: A stream is a source of data or destination of data that may be associated with a disk or other I/O device. The source stream provides data to a program and it is known as input stream. The destination streamr eceives the output from the program and is known as output stream.

20. What is meant by file opening?

Ans: The action of connecting a program to a file is called opening of a file. This requires creating an I/O stream before reading or writing the data.

21. What is FILE?

Ans: FILE is a predefined data type. It is defined in stdio.h file.

22. What is a file pointer?

Ans: The pointer to a FILE data type is called as a stream pointer or a file pointer. A file pointer points to the block of information of the stream that had just been opened.

23. How is fopen()used ?

Ans: The function fopen() returns a file pointer. Hence a file pointer is declared and it is assigned as FILE *fp; fp= fopen(filename,mode); filename is a string representing the name of the file and the mode represents: ―r‖ for read operation ―w‖ for write operation ―a‖ for append operation ―r+‖,‖w+‖,‖a+‖ for update operation

24.How is a file closed ?

Ans: A file is closed using fclose() function Eg. fclose(fp); Where fp is a file pointer.

25. What is a random access file?

Ans: A file can be accessed at random using fseek() function fseek(fp,position,origin); fp file pointer position number of bytes offset from origin origin 0,1 or 2 denote the beginning ,current position or end of file respectively.

26. What is the purpose of ftell ?

Ans: The function ftell() is used to get the current file represented by the file pointer. ftell(fp); returns a long integer value representing the current file position of the file pointed by the file pointer fp.If an error occurs ,-1 is returned.

27. What is the purpose of rewind() ?

Ans: The function rewind is used to bring the file pointer to the beginning of the file. Rewind(fp); Where fp is a file pointer.Also we can get the same effect by feek(fp,0,0);