



**G.S. Mandal's
Maharashtra Institute of Technology, Aurangabad
Department of Computer Science and Engineering**

LAB MANUAL

CSE206: Advanced C (2019-20 Part 1)

Maharashtra Institute of Technology, Aurangabad
NH-211, MIT Campus, Satara Village Road, Aurangabad- 431 010 (M.S.); India.
Phone: (0240) 2375222; Fax: (0240) 2376618, E-mail: principalmitt@mit.asia
Website: www.btech.mit.asia

Department of Computer Science and Engineering

Vision

To develop the department as a center of excellence in the field of computer science and engineering by imparting knowledge & training to the students for meeting growing needs of the industry & society.

Mission

Providing quality education through a well-designed curriculum in tune with the challenging needs of software industry by providing state of the art facilities and to impart knowledge in the thrust areas of computer science and engineering.

Department of Computer Science and Engineering

Program Educational Objectives

PEO1: To prepare the students to achieve success in Computing Domain to create individual careers, innovations or to work as a key contributor to the private or Government sector and society.

PEO2: To develop the ability among the students to understand Computing and mathematical fundamentals and apply the principles of Computer Science for analyzing, designing and testing software for solving problems.

PEO3: To empower the students with ability to quickly reflect the changes in the new technologies in the area of computer software, hardware, networking and database management.

PEO4: To promote the students with awareness for lifelong learning, introduce them to professional practice, ethics and code of professionalism to remain continuous in their profession and leaders in technological society.

Program Specific Objectives

PSO1: Identify appropriate data structures and algorithms for a given contextual problem and develop programs to design and implement applications.

PSO3: Design and manage the large databases and develop their own databases to solve real world problems and to design, build, manage networks and apply wireless techniques in mobile based applications.

PSO3: Design a variety of computer-based components and systems using computer hardware, system software, systems integration process and use standard testing tools for assuring the software quality.

Department of Computer Science and Engineering

Program Outcomes

PO1: Apply knowledge of mathematics, science, and engineering fundamentals to solve problems in Computer science and Engineering.

PO2: Identify, formulate and analyze complex problems.

PO3: Design system components or processes to meet the desired needs within realistic constraints for the public health and safety, cultural, societal and environmental considerations.

PO4: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data for valid conclusions.

PO5: Select and apply modern engineering tools to solve the complex engineering problem.

PO6: Apply knowledge to assess contemporary issues.

PO7: Understand the impact of engineering solutions in a global, economic, environmental, and societal context.

PO8: Apply ethical principles and commit to professional ethics and responsibilities.

PO9: Work effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

PO10: Communicate effectively in both verbal and written form.

PO11: Demonstrate knowledge and apply engineering and management principles to manage projects and in multi-disciplinary environment.

PO12: To engage in life-long learning to adopt to the technological changes.

Department of Computer Science and Engineering

Course: CSE 206- Advanced C CSE 224- Computer Laboratory I

Course Outcomes:

After Completing the course students will be able to

CO1: Analyze the given situation and use appropriate user defined data types in C to implement programming solution (BL3)

CO2: Apply dynamic memory allocation to solve given programming problem (BL3)

CO3: Apply bitwise operators to solve programming problem (BL3)

CO4: Compare the use of storage class specifiers in C (BL3)

CO5: Apply C file handling operations to perform operations on text and binary file (BL3)

CO6: Analyze the use of topics learned in the subject to provide solution for any real-world problem statement of your choice in group of two(BL4)

Mapping

Experiment No.	Blooms Level	Mapping To CO	Mapping To PO
1	3	1	1,2,3
2	3	1	1,2,3
3	3	2	1,2,3
4	3	3	1,2,3
5	3	5	1,2,3
6	3	5	1,2,3
7	3	5	1,2,3
8	3	4	1,2,3
9	3	1	1,2,3
10	4	6	1,2,3,9,10,11



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2018-19)

LAB: write lab no

SUBJECT: subject code-subject name

Index

Contents		Page No.
Vision Mission		i
Program Educational Objectives		ii
Program Specific Objectives		ii
Program Outcomes		iii
Course Outcomes		iv
Mapping		iv
Exp No.	Title of Experiment	page nos.
1	Lab Work#0: Review of Basic Constructs	1-3
2	Lab Work#1: Structures and Unions	4-6
3	Lab Work#2: Dynamic Memory Allocation	7-9
4	Lab Work#3: Bitwise Operators	10-13
5	Lab Work#4: File Handling (Text Files)	14-17
6	Lab Work#5: File Handling operations on text files	18-19
7	Lab Work#6: File Handling (Binary Files)	20-24
8	Lab Work#7: Storage Class Specifiers	25-31
9	Lab Work#8: Hardware Interaction	32
10	Lab Work#9: Mini Project	33
Appendix A	Problems Statements	34-43



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#0

NAME:
ROLL NO:
BATCH:

AIM: Applying Basic C Programming Constructs- functions, conditional statements and iteration statements

OBJECTIVE: The objective of this lab work is:

1. Students will be able to analyze given Problem Statement#0 and apply relevant C programming constructs to solve the problem in group of Two.
2. Students will revise the Basic C Programming Constructs, - functions, conditional statements and iteration statements, and apply them in programming

OUTCOMES: After completing the Lab Work#0 students will be able to:

1. Apply the Basic C Programming Constructs, - functions, conditional statements and iteration statements, in solving programming problem mentioned in Problem Statement#0
2. Analyze Problem Statement#1 and choose which programming elements to apply.
3. Function effectively as member of the team for performing lab work#0

PRE-REQUISITE:

1. Basic C Programming Constructs, - functions, conditional statements and iteration

INPUT-OUTPUT:

1. The input will be RPM and output should be whether the watch unit is suitable to launch in market or must be discarded.

THEORY:

C is a general-purpose programming language that is extremely popular, simple and flexible. It is machine-independent, structured programming language which is used extensively in various applications.

Where is C used? Key Applications

1. 'C' language is widely used in embedded systems.
2. It is used for developing system applications.
3. It is widely used for developing desktop applications.
4. Most of the applications by Adobe are developed using 'C' programming language.
5. It is used for developing browsers and their extensions. Google's Chromium is built using 'C' programming language.
6. It is used to develop databases. MySQL is the most popular database software which is built using 'C'.
7. It is used in developing an operating system. Operating systems such as Apple's OS X, Microsoft's Windows, and Symbian are developed using 'C' language. It is used for developing desktop as well as mobile phone's operating system.
8. It is used for compiler production.
9. It is widely used in IOT applications.

C if Statement

The syntax of the if statement in C programming is:

1. `if (test expression)`
2. `{`



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

```
3. // statements to be executed if the test expression is true
4. }
```

How if statement works?

The **if** statement evaluates the test expression inside the parenthesis ().

If the test expression is evaluated to true, statements inside the body of **if** are executed.

If the test expression is evaluated to false, statements inside the body of **if** are not executed.

C for Loop

C programming has three types of loops:

for loop

while loop

do...while loop

We will learn about for loop in this tutorial. In the next tutorial, we will learn about while and do...while loop.

for Loop

The syntax of the for loop is:

```
1. for (initializationStatement; testExpression; updateStatement)
2. {
3. // statements inside the body of loop
4. }
```

How for loop works?

The initialization statement is executed only once.

Then, the test expression is evaluated. If the test expression is evaluated to false, the for loop is terminated.

However, if the test expression is evaluated to true, statements inside the body of for loop are executed, and the update expression is updated.

Again the test expression is evaluated.

This process goes on until the test expression is false. When the test expression is false, the loop terminates.

SOURCE CODE

//source code here

INPUT/OUTPUT:

//Execute the program for all possible combination of inputs and mention the output



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

ASSESSMENT QUESTIONS:

1. Can you implement a program to find if entered number is factorial number.

Eg: Input 120--- Yes (as it is factorial of 5)

Input 100--- No(as it is not factorial of any number)

2. Compare the situations where it will be suitable to use for loop vs while loop

** Courtesy:

<https://www.guru99.com/>

<https://www.programiz.com/>



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#1

NAME:

ROLL NO:

BATCH:

AIM: Analyze the appropriate use of non-primitive data types- structure, union and arrays in C.

OBJECTIVE: The objective of this lab work is:

3. Students will be able to analyze given Problem Statement#1 and apply relevant non primitive data types appropriately- structure, union, array.

OUTCOMES: After completing the Lab Work#1 students will be able to:

1. Evaluate appropriate usage of – structure, union, arrays
2. Apply structure, union and array to solve the given problem.
3. Apply nested structures, nested union and array of structures.

PRE-REQUISITE:

2. Basic C Programming Constructs, - functions, conditional statements and iteration
3. Syntax of structure, union, array

INPUT-OUTPUT:

2. The input will be option from user as given in the banking application specified in Problem Statement#1.
3. The output will be displaying the account details after applying the selected operation and showing the menu again till user's choice.

THEORY:

C Structure

In C, there are cases where we need to store multiple attributes of an entity. It is not necessary that an entity has all the information of one type only. It can have different attributes of different data types. For example, an entity **Student** may have its name (string), roll number (int), marks (float). To store such type of information regarding an entity student, we have the following approaches:

- Construct individual arrays for storing names, roll numbers, and marks.
- Use a special data structure to store the collection of different data types.

Structure in c is a user-defined data type that enables us to store the collection of different data types. Each element of a structure is called a member. Structures ca; simulate the use of classes and templates as it can store various information

The **,struct** keyword is used to define the structure. Let's see the syntax to define the structure in c.

```
struct structure_name
{
    data_type member1;
    data_type member2;
    .
    .
```



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

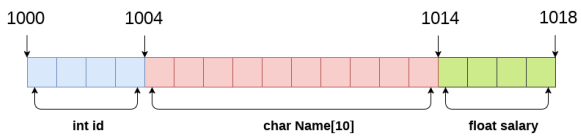
CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

data_type memberN;

};



```
struct Employee  
{  
    int id;  
    char Name[10];  
    float salary;  
} emp;
```

sizeof (emp) = 4 + 10 + 4 = 18 bytes

where;
sizeof (int) = 4 byte
sizeof (char) = 1 byte
sizeof (float) = 4 byte

□ → 1 byte

Here, **struct** is the keyword; **employee** is the name of the structure; **id**, **name**, and **salary** are the members or fields of the structure.

Union

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.

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 variable definition, such as `int i;` or `float f;` or any other valid variable definition. At the end of the union's definition, before the final semicolon, you can specify one or more union variables but it is optional. Here is the way you would define a union type named `Data` having three members `i`, `f`, and `str` –

```
union Data {  
    int i;  
    float f;  
    char str[20];  
} data;
```

Now, a variable of **Data** type can store an integer, a floating-point number, or a string of characters. It means a single variable, i.e., same memory location, can be used to store multiple types of data. You can use any built-in or user defined data types inside a union based on your requirement.

The memory occupied by a union will be large enough to hold the largest member of the union. For example, in the above example, `Data` type will occupy 20 bytes of memory space because this is the maximum space which can be occupied by a character string.

Accessing Union Members

To access any member of a union, we use the **member access operator** (`.`). The member access operator is coded as a period



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

between the union variable name and the union member that we wish to access.
the values of **i** and **f** members of union got corrupted because the final value assigned to the variable has occupied the memory location and this is the reason that the value of **str** member is getting printed very well.

SOURCE CODE

//source code here

INPUT/OUTPUT:

//Execute the program for all possible combination of inputs and mention the output

ASSESSMENT QUESTIONS:

1. Consider the following C declaration

```
struct {  
    short s[5];  
    union {  
        float y;  
        long z;  
    }u;  
} t;
```

Assume that objects of the type short, float and long occupy 2 bytes, 4 bytes and 8 bytes, respectively. The memory requirement for variable t, ignoring alignment considerations, is

2. Compare Structure and Union on at-least 3 parameters(in your own words)

** Courtesy:

<https://www.javatpoint.com>

<https://www.tutorialspoint.com/>



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#2

NAME:
ROLL NO:
BATCH:

AIM: Apply Dynamic Memory Allocation for given problem

OBJECTIVE: The objective of this lab work is:

4. Students will be able to analyze given Problem Statement#2 and apply Dynamic Memory Allocation to solve the problem

OUTCOMES: After completing the Lab Work#2 students will be able to:

4. Evaluate appropriate usage of – Dynamic Memory Allocation
5. Apply Dynamic Memory Allocation

PRE-REQUISITE:

4. Basic C Programming Constructs, - functions, conditional statements and iteration
5. Syntax of structure
6. Dynamic Memory Allocation operators in C

INPUT-OUTPUT:

4. The input will be option from user as given in the banking application specified in Problem Statement#2.
5. The output will be displaying the account details after applying the selected operation and showing the menu again till user's choice.

THEORY:

DIFFERENCE BETWEEN STATIC MEMORY ALLOCATION AND DYNAMIC MEMORY ALLOCATION IN C:

Static memory allocation	Dynamic memory allocation
In static memory allocation, memory is allocated while writing the C program. Actually, user requested memory will be allocated at compile time.	In dynamic memory allocation, memory is allocated while executing the program. That means at run time.
Memory size can't be modified while execution. Example: array	Memory size can be modified while execution. Example: Linked list

Memory Allocation Process

Global variables, **static** variables and program instructions get their memory in **permanent** storage area whereas **local** variables are stored in a memory area called **Stack**.

The memory space between these two region is known as **Heap** area. This region is used for dynamic memory allocation during execution of the program. The size of heap keep changing.



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD**

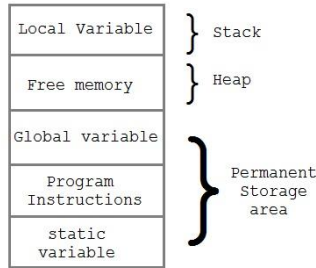
LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I



DYNAMIC MEMORY ALLOCATION IN C:

The process of allocating memory during program execution is called dynamic memory allocation.

DYNAMIC MEMORY ALLOCATION FUNCTIONS IN C:

C language offers 4 dynamic memory allocation functions. They are,

1. malloc()
2. calloc()
3. realloc()
4. free()

Function	Syntax
malloc ()	malloc (number *sizeof(int));
calloc ()	calloc (number, sizeof(int));
realloc ()	realloc (pointer_name, number * sizeof(int));
free ()	free (pointer_name);

Function	Description
malloc()	allocates requested size of bytes and returns a void pointer pointing to the first byte of the allocated space
calloc()	allocates space for an array of elements, initialize them to zero and then returns a void pointer to the memory
free	releases previously allocated memory
realloc	modify the size of previously allocated space



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Difference between malloc() and calloc()

calloc()	malloc()
calloc() initializes the allocated memory with 0 value.	malloc() initializes the allocated memory with garbage values.
Number of arguments is 2	Number of argument is 1
Syntax : (cast_type *)calloc(blocks , size_of_block);	Syntax : (cast_type *)malloc(Size_in_bytes);

Source Code

INPUT/OUTPUT:

//Execute the program for all possible combination of inputs and mention the output

ASSESSMENT QUESTIONS:

1. Write logic (in form of diagram) to find if there are any loops in a singly linked list?
2. There are singly linked lists, double linked lists can you propose a tertiary linked list?? And where to use this(any application)

** Courtesy:

<https://www.javatpoint.com>

<https://www.tutorialspoint.com/>



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#3

NAME:
ROLL NO:
BATCH:

AIM: Apply Bitwise Operators to solve given programming problems

OBJECTIVE: The objective of this lab work is:

- Students will be able to analyze given Problems in the Problem Statement#3 and apply Bitwise operators to solve and write programs.

OUTCOMES: After completing the Lab Work#3 students will be able to:

- Evaluate appropriate usage of – Bitwise Operators to solve programming problems
- Apply C bitwise operators in programming

PRE-REQUISITE:

- Basic C Programming Constructs, - conditional statements
- Bitwise operators in c
- Binary representation of numbers

INPUT-OUTPUT:

- There are 10 general problems given in problem statement#3.
- The output will be as desired in respective problems.

THEORY:

Bit Wise Operators In C:

- These operators are used to perform bit operations. Decimal values are converted into binary values which are the sequence of bits and bit wise operators work on these bits.
- Bit wise operators in C language are & (bitwise AND), | (bitwise OR), ~ (bitwise NOT), ^ (XOR), << (left shift) and >> (right shift).

TRUTH TABLE FOR BIT WISE OPERATION & BIT WISE OPERATORS:

x	y	x y	x&y	x^y
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

Below Are The Bit-Wise Operators And Their Name In C Language.

- & – Bitwise AND
- | – Bitwise OR
- ~ – Bitwise NOT



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

4. \wedge – XOR
5. \ll – Left Shift
6. \gg – Right Shift

Integer Representation

Integers are whole numbers or fixed-point numbers with the radix point fixed after the least-significant bit. They are contrast to real numbers or floating-point numbers, where the position of the radix point varies. It is important to take note that integers and floating-point numbers are treated differently in computers. They have different representation and are processed differently (e.g., floating-point numbers are processed in a so-called floating-point processor). Floating-point numbers will be discussed later.

Computers use a fixed number of bits to represent an integer. The commonly-used bit-lengths for integers are 8-bit, 16-bit, 32-bit or 64-bit. Besides bit-lengths, there are two representation schemes for integers:

1. Unsigned Integers: can represent zero and positive integers.
2. Signed Integers: can represent zero, positive and negative integers. Three representation schemes had been proposed for signed integers:
 - a. Sign-Magnitude representation
 - b. 1's Complement representation
 - c. 2's Complement representation

You, as the programmer, need to decide on the bit-length and representation scheme for your integers, depending on your application's requirements. Suppose that you need a counter for counting a small quantity from 0 up to 200, you might choose the 8-bit unsigned integer scheme as there is no negative numbers involved.

Sign-Magnitude Representation

- The most-significant bit (msb) is the *sign bit*, with value of 0 representing positive integer and 1 representing negative integer.
- The remaining $n-1$ bits represents the magnitude (absolute value) of the integer. The absolute value of the integer is interpreted as "the magnitude of the $(n-1)$ -bit binary pattern".

Example 1: Suppose that $n=8$ and the binary representation is 0 100 0001B.

Sign bit is 0 \Rightarrow positive

Absolute value is 100 0001B = 65D

Hence, the integer is +65D

The drawbacks of sign-magnitude representation are:

1. There are two representations (0000 0000B and 1000 0000B) for the number zero, which could lead to inefficiency and confusion.



CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

2. Positive and negative integers need to be processed separately.

1's Complement Representation

- Again, the most significant bit (msb) is the *sign bit*, with value of 0 representing positive integers and 1 representing negative integers.
- The remaining $n-1$ bits represents the magnitude of the integer, as follows:
 - for positive integers, the absolute value of the integer is equal to "the magnitude of the $(n-1)$ -bit binary pattern".
 - for negative integers, the absolute value of the integer is equal to "the magnitude of the *complement (inverse)* of the $(n-1)$ -bit binary pattern" (hence called 1's complement).

Example 1: Suppose that $n=8$ and the binary representation 0 100 0001B.

Sign bit is 0 \Rightarrow positive

Absolute value is 100 0001B = 65D

Hence, the integer is +65D

Again, the drawbacks are:

1. There are two representations (0000 0000B and 1111 1111B) for zero.
2. The positive integers and negative integers need to be processed separately.

2's Complement Representation

- Again, the most significant bit (msb) is the *sign bit*, with value of 0 representing positive integers and 1 representing negative integers.
- The remaining $n-1$ bits represents the magnitude of the integer, as follows:
 - for positive integers, the absolute value of the integer is equal to "the magnitude of the $(n-1)$ -bit binary pattern".
 - for negative integers, the absolute value of the integer is equal to "the magnitude of the *complement of the $(n-1)$ -bit binary pattern plus one*" (hence called 2's complement).

Example 1: Suppose that $n=8$ and the binary representation 0 100 0001B.

Sign bit is 0 \Rightarrow positive

Absolute value is 100 0001B = 65D

Hence, the integer is +65D

Computers use 2's complement in representing signed integers. This is because:

1. There is only one representation for the number zero in 2's complement, instead of two representations in sign-magnitude and 1's complement.
2. Positive and negative integers can be treated together in addition and subtraction. Subtraction can be carried out using the "addition logic".



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Source Code with output

1. To check if Nth Bit is set/ reset.
2. To check if entered number is even or odd.
3. To count number of zero's in entered numbers' binary equivalent
4. To count number of one's in entered numbers' binary equivalent
5. To rotate bits of a number.
6. Swap two numbers
7. Find Two's Complement of entered number
8. Convert Uppercase to Lower Case
9. C Program to round Floor of integer to next Lower Power of 2
10. C Program to Reverse all the Bits of a 32-bit Integer using Bitwise

ASSESSMENT QUESTIONS:

1. Write program to convert a decimal number to Hexa-Decimal. (you can use arrays)

** Courtesy:

<https://www3.ntu.edu.sg>

<https://fresh2refresh.com>



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#4

NAME:
ROLL NO:
BATCH:

AIM: Apply C Text File handling to solve given problem

OBJECTIVE: The objective of this lab work is:

6. Students will be able to analyze given Problems in the Problem Statement#4 and apply C file handling functions to come up with a programming solution

OUTCOMES: After completing the Lab Work#4 students will be able to:

8. Evaluate appropriate usage of – Text File handling functions in C
9. Apply C Text File handling
10. Understand usage of file opening modes

PRE-REQUISITE:

10. Basic C Programming Constructs
11. Basics of C File Handling
12. File opening modes
13. Text file handling functions

INPUT-OUTPUT:

8. As per menu options in problems statement#4
9. The output will be as desired in respective problems.

THEORY:

WHAT IS FILE?

File is a collection of bytes that is stored on secondary storage devices like disk. There are two kinds of files in a system. They are,

1. Text files (ASCII)
 2. Binary files
- Text files contain ASCII codes of digits, alphabetic and symbols.
 - Binary file contains collection of bytes (0's and 1's). Binary files are compiled version of text files.

A **file** represents a sequence of bytes on the disk where a group of related data is stored. File is created for permanent storage of data. It is a ready made structure.

In C language, we use a structure **pointer of file type** to declare a file.

C provides a number of functions that helps to perform basic file operations. Following are the functions,

Function	description
----------	-------------



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

fopen()	create a new file or open a existing file
fclose()	closes a file
getc()	reads a character from a file
putc()	writes a character to a file
fscanf()	reads a set of data from a file
fprintf()	writes a set of data to a file
getw()	reads a integer from a file
putw()	writes a integer to a file
fseek()	set the position to desire point
ftell()	gives current position in the file
rewind()	set the position to the begining point

Opening a File or Creating a File

The `fopen ()` function is used to create a new file or to open an existing file.

General Syntax:

```
*fp = FILE *fopen (const char *filename, const char *mode) ;
```

Here, `*fp` is the FILE pointer (`FILE *fp`), which will hold the reference to the opened(or created) file.

filename is the name of the file to be opened and **mode** specifies the purpose of opening the file. Mode can be of following types,

mode	description
r	opens a text file in reading mode
w	opens or create a text file in writing mode.



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

a	opens a text file in append mode
r+	opens a text file in both reading and writing mode
w+	opens a text file in both reading and writing mode
a+	opens a text file in both reading and writing mode
rb	opens a binary file in reading mode
wb	opens or create a binary file in writing mode
ab	opens a binary file in append mode
rb+	opens a binary file in both reading and writing mode
wb+	opens a binary file in both reading and writing mode
ab+	opens a binary file in both reading and writing mode

Closing a File

The `fclose()` function is used to close an already opened file.

General Syntax :

```
int fclose ( FILE *fp) ;
```

Here `fclose()` function closes the file and returns **zero** on success, or **EOF** if there is an error in closing the file. This **EOF** is a constant defined in the header file **stdio.h**.

Difference between Append and Write Mode

Write (w) mode and Append (a) mode, while opening a file are almost the same. Both are used to write in a file. In both the modes, new file is created if it doesn't exist already.

The only difference they have is, when you open a file in the write mode, the file is reset, resulting in deletion of any data already present in the file. While in append mode this will not happen. Append mode is used to append or add data to the existing data of file(if any). Hence, when you open a file in Append(a) mode, the cursor is positioned at the end of the present data in the file.



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

SOURCE CODE:

***** copy your source code here with proper indentation

INPUT-OUTPUT:

***** Paste the output

ASSESSMENT QUESTIONS:

1. Consider the following to write the file handling program. It is expected to do the following:
 - a) Read a file
 - b) Accept a word from the user
 - c) Check whether the word exists in the file. It should print appropriate message if found or not found.
 - d) Repeat the steps b and c till the user enters 'stop'.

Courtesy:

<https://www.javatpoint.com/>



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#5

NAME:
ROLL NO:
BATCH:

AIM: Apply C Text File handling to solve given problem

OBJECTIVE: The objective of this lab work is:

7. Students will be able to analyze given Problems in the Problem Statement#5 and apply C file handling functions to come up with a programming solution

OUTCOMES: After completing the Lab Work#4 students will be able to:

11. Evaluate appropriate usage of – Text File handling functions in C
12. Apply C Text File handling
13. Understand usage of file opening modes

PRE-REQUISITE:

14. Basic C Programming Constructs
15. Basics of C File Handling
16. File opening modes
17. Text file handling functions

INPUT-OUTPUT:

10. As per menu options in problems statement#4
11. The output will be as desired in respective problems.

THEORY:

Logic to copy contents from one file to another

Step by step descriptive logic to copy file contents from one file to another.

1. Input file path of source and destination file.
2. Open source file in r (read) and destination file in w (write) mode.
3. Read character from source file and write it to destination file using fputc().
4. Repeat step 3 till source file has reached end.
5. Close both source and destination file.

Logic to Count Number of Words in a given Text or Sentence

This is a C Program to Count the Number of Words in a given text or Sentence.

Problem Description

This program takes a string as input and count the number of words in the input string.

Problem Solution

1. Take a string as input.
2. Using for loop search for a empty space in between the words in the string.
3. Consecutively increment a variable. This variable gives the count of number of words.

Logic to find occurrence of a word in file

Step by step descriptive logic to find first occurrence of a word in file.

1. Input source file in which you need to search, store its reference to fptr.
2. Input word to search in file, store it in some variable say word.
3. Initialize two variables line = -1 and col = -1. They will store index of line and column of searched word in file.
4. Read a line from file, store it in str.
5. Increment line count by one.



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

6. Find first index of word in str and store it in col. If word is found in str then print line and col which is required index and jump to step 8.
7. Now, here's the tricky part. I have used strstr() C library function. It returns pointer to first occurrence of a given word in a string. To find first occurrence of word in str use `pos = strstr(str, word);` (where pos is pointer to character).
8. pos will point to first occurrence of word in str if exists, otherwise points to NULL. Check if (`pos != NULL`) then find column index using `col = pos - str`.
9. Repeat step 4-6 till end of file.
10. Close fptr file to release all resources.

Logic to count occurrences of a word in file

Step by step descriptive logic to count occurrences of a word in file.

1. Open source file in r (read) mode, store its reference to fptr.
2. Input word to search from user, store it in word.
3. Initialize a variable count = 0 to store total occurrences of word.
4. Read a line from file and store it in str.
5. Increment count by one, if an occurrence of word is found in str.
6. Repeat step 4-5 till end of file.
7. Finally close fptr to release all resources.

SOURCE CODE:

***** copy your source code here with proper indentation

INPUT-OUTPUT:


***** Paste the output

ASSESSMENT QUESTIONS:

1. Try to find out problems of using files as storage, limitations of file handling. What is the better alternative?

Courtesy:

<https://codeforwin.org>

	G.S. MANDAL'S	
	MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD	
	LAB WORK INSTRUCTION SHEET	
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING		
CLASS: S.Y. B. TECH PART: 1 (2019-20)	LAB:	SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#6

NAME:

ROLL NO:

BATCH:

AIM: Apply C Binary File handling functions to solve given problem

OBJECTIVE: The objective of this lab work is:

8. Students will be able to analyze given Problems in the Problem Statement#6 and apply C file handling functions to come up with a programming solution

OUTCOMES: After completing the Lab Work#6 students will be able to:

14. Evaluate appropriate usage of – Binary File handling functions in C- fread(), fwrite()
15. Apply C Binary File handling
16. Use file handling for record keeping and updating as per problem statement

PRE-REQUISITE:

18. Basic C Programming Constructs
19. C file handling functions
20. File opening modes

INPUT-OUTPUT:

12. As per menu options in problems statement#6
13. The output will be as desired in respective problems.

THEORY:

Binary files

Binary files are mostly the **.bin** files in your computer. Instead of storing data in plain text, they store it in the binary form (0's and 1's). They can hold a higher amount of data, are not readable easily, and provides better security than text files.

Reading and writing to a binary file

Functions fread() and fwrite() are used for reading from and writing to a file on the disk respectively in case of binary files.

Writing to a binary file

1. To write into a binary file, you need to use the fwrite() function. The functions take four arguments:
 - address of data to be written in the disk
 - size of data to be written in the disk
 - number of such type of data
 - pointer to the file where you want to write.


1. fwrite(addressData, sizeData, numbersData, pointerToFile);

Example 3: Write to a binary file using fwrite()

```
#include <stdio.h>
#include <stdlib.h>
```

```
struct threeNum
{
    int n1, n2, n3;
};
```

```
int main()
{
```

	G.S. MANDAL'S	
	MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD	
	LAB WORK INSTRUCTION SHEET	
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING		
CLASS: S.Y. B. TECH PART: 1 (2019-20)	LAB:	SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

<pre> int n; struct threeNum num; FILE *fptr; if ((fptr = fopen("C:\\program.bin", "wb")) == NULL){ printf("Error! opening file"); // Program exits if the file pointer returns NULL. exit(1); } for(n = 1; n < 5; ++n) { num.n1 = n; num.n2 = 5*n; num.n3 = 5*n + 1; fwrite(&num, sizeof(struct threeNum), 1, fptr); } fclose(fptr); return 0; } </pre> <ul style="list-style-type: none"> • In this program, we create a new file program.bin in the C drive. • We declare a structure threeNum with three numbers - n1, n2 and n3, and define it in the main function as num. • Now, inside the for loop, we store the value into the file using fwrite(). • The first parameter takes the address of num and the second parameter takes the size of the structure threeNum. • Since we're only inserting one instance of num, the third parameter is 1. And, the last parameter *fptr points to the file we're storing the data. • Finally, we close the file. <p>Reading from a binary file Function fread() also take 4 arguments similar to the fwrite() function as above.</p> <ol style="list-style-type: none"> 1. fread(addressData, sizeData, numbersData, pointerToFile); <p>Example 4: Read from a binary file using fread()</p> <pre> #include <stdio.h> #include <stdlib.h> struct threeNum { int n1, n2, n3; }; int main() { int n; </pre>



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

```
struct threeNum num;
FILE *fptr;

if ((fptr = fopen("C:\\program.bin", "rb")) == NULL){
    printf("Error! opening file");

    // Program exits if the file pointer returns NULL.
    exit(1);
}

for(n = 1; n < 5; ++n)
{
    fread(&num, sizeof(struct threeNum), 1, fptr);
    printf("n1: %d\tn2: %d\tn3: %d", num.n1, num.n2, num.n3);
}
fclose(fptr);

return 0;
}
```

- In this program, you read the same file program.bin and loop through the records one by one.
- In simple terms, you read one threeNum record of threeNum size from the file pointed by *fptr into the structure num.
- You'll get the same records you inserted in Example 3.

Getting data using fseek()

If you have many records inside a file and need to access a record at a specific position, you need to loop through all the records before it to get the record.

This will waste a lot of memory and operation time. An easier way to get to the required data can be achieved using fseek().

As the name suggests, fseek() seeks the cursor to the given record in the file.

Syntax of fseek()

1. fseek(FILE * stream, long int offset, int whence);

The first parameter stream is the pointer to the file. The second parameter is the position of the record to be found, and the third parameter specifies the location where the offset starts.

Whence	Meaning
SEEK_SET	Starts the offset from the beginning of the file.
SEEK_END	Starts the offset from the end of the file.
SEEK_CUR	Starts the offset from the current location of the cursor in the file.

Different whence in fseek()



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

```
Example 5: fseek()
#include <stdio.h>
#include <stdlib.h>

struct threeNum
{
    int n1, n2, n3;
};

int main()
{
    int n;
    struct threeNum num;
    FILE *fptr;

    if ((fptr = fopen("C:\\program.bin", "rb")) == NULL){
        printf("Error! opening file");

        // Program exits if the file pointer returns NULL.
        exit(1);
    }

    // Moves the cursor to the end of the file
    fseek(fptr, -sizeof(struct threeNum), SEEK_END);

    for(n = 1; n < 5; ++n)
    {
        fread(&num, sizeof(struct threeNum), 1, fptr);
        printf("n1: %d\t n2: %d\t n3: %d\n", num.n1, num.n2, num.n3);
        fseek(fptr, -2*sizeof(struct threeNum), SEEK_CUR);
    }
    fclose(fptr);

    return 0;
}
```

- This program will start reading the records from the file program.bin in the reverse order (last to first) and prints it.

SOURCE CODE:

***** copy your source code here with proper indentation



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

INPUT-OUTPUT:


***** Paste the output

ASSESSMENT QUESTIONS:

2. Compare Text and Binary Files with their applications

Courtesy:

<https://www.programiz.com>

	G.S. MANDAL'S	
	MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD	
	LAB WORK INSTRUCTION SHEET	
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING		
CLASS: S.Y. B. TECH PART: 1 (2019-20)	LAB:	SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#7

NAME:
ROLL NO:
BATCH:

AIM: Apply storage class specifiers and analyze the given program excerpts using different storage class specifiers and predict their response

OBJECTIVE: The objective of this lab work is:

1. Students will be able to apply storage class specifier – static and extern
2. Students will be able to analyze the usage of these storage class specifiers and their effect.
3. Students will understand – lifetime, scope concept.

OUTCOMES: After completing the Lab Work#7 students will be able to:

1. Make use of static and extern in the program
2. Understand the use of storage class specifiers
3. Understand lifetime and scope of a variable and its effects

PRE-REQUISITE:

1. Basic C Programming Constructs
2. Basic knowledge of storage class specifiers in C

THEORY:

A storage class defines the scope, visibility and lifetime of variables. Refer For More [C Storage Classes](#)

auto storage class
static storage class
register storage class
extern storage class
storage class

All variables declared inside a function without any storage classes keyword is considered as auto storage class by default.

In auto storage class a memory is allocated automatically to the variables when the function is called and deallocates automatically when the function exits.

Variables under auto storage classes are local to the block or function. So, it is also called as local variables.

The keyword **auto** is rarely used because all uninitialized variables are said to have auto storage classes.

static storage class

A static variable is a variable that tells the compiler to retain the value until the program terminates.

They are created once when the function is called, even though the function gets repeated it retains the same value and exists until the program terminates.

The default value of static variable is zero(0).


The keyword for a variable to declared under static storage class is **static**

register storage class

Register variables are similar to auto variables. The only difference is register variables are stored in CPU register instead of memory.

Register variables are faster than normal variables. Mostly programmers uses register to store frequently used variables.

Programmers can store only few variables in the CPU register because size of register is less.

	G.S. MANDAL'S	
	MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD	
	LAB WORK INSTRUCTION SHEET	
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING		
CLASS: S.Y. B. TECH PART: 1 (2019-20)	LAB:	SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

The keyword for a variable to declared under register storage class is **register**.
extern storage class
extern variable is a programmer's shorthand to represent external variable.
extern variables are also known as global variables because extern variables are declared above the main function.
So, the variables can be accessed by any function.
We can also access extern variables of one file to another file. But make sure both files are in same folder. The keyword for a variable to declared under extern storage class is extern.

1. auto

This storage class denotes that an identifier has automatic storage duration. This means once the scope in which the identifier was defined ends, the object denoted by the identifier is no longer valid.
Since all objects, not living in global scope or being declared static, have automatic storage duration by default when defined, this keyword is mostly of historical interest and should not be used:

```
int foo(void)
{
    /* An integer with automatic storage duration. */
    auto int i = 3;

    /* Same */
    int j = 5;

    return 0;
} /* The values of i and j are no longer able to be used. */
```

2. register

Hints to the compiler that access to an object should be as fast as possible. The register storage class is more appropriate for variables that are defined inside a block and are accessed with high frequency. For example,

```
/* prints the sum of the first 5 integers*/
/* code assumed to be part of a function body*/
{
    register int k, sum;
    for(k = 1, sum = 0; k < 6; sum += k, k++);
    printf("\t%d\n",sum);
}
```

In C11, The _Alignof operator is also allowed to be used with register arrays.

3. extern

Used to declare an object or function that is defined elsewhere (and that has external linkage). In general, it is used to declare an object or function to be used in a module that is not the one in which the corresponding object or function is defined:

```
/* file1.c */
int foo = 2; /* Has external linkage since it is declared at file scope. */
/* file2.c */
#include <stdio.h>
int main(void)
{
    /* `extern` keyword refers to external definition of `foo`. */
    extern int foo;
```




G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

```
printf("%d\n", foo);  
return 0;  
}
```

Things get slightly more interesting with the introduction of the inline keyword in C99:

Hints to the compiler that the function bar might be inlined and suppresses the generation of an external symbol, unless stated otherwise.

/* Should usually be placed in a header file such that all users see the definition */

```
inline void bar(int drink)
```

```
{  
    printf("You ordered drink no.%d\n", drink);  
}
```

To be found in just one .c file. Creates an external function definition of bar for use by other files. The compiler is allowed to choose between the inline version and the external definition when bar is called. Without this line, bar would only be an inline function, and other files would not be able to call it.

```
extern void bar(int);
```

4. static

The static storage class serves different purposes, depending on the location of the declaration in the file:

To confine the identifier to that translation unit only (scope=file).

/* No other translation unit can use this variable. */

```
static int i;
```

/* Same; static is attached to the function type of f, not the return type int. */

```
static int f(int n);
```

To save data for use with the next call of a function (scope=block):

```
void foo() {  
    static int a = 0; /* has static storage duration and its lifetime is the  
        * entire execution of the program; initialized to 0 on  
        * first function call */  
    int b = 0; /* b has block scope and has automatic storage duration and  
        * only "exists" within function */
```

```
    a += 10;
```

```
    b += 10;
```

```
    printf("static int a = %d, int b = %d\n", a, b);
```

```
}
```

```
int main(void) {  
    int i;  
    for (i = 0; i < 5; i++) {  
        foo();  
    }  
}
```

```
return 0;
```



CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

```
}  
/*  
This code prints:
```

```
static int a = 10, int b = 10  
static int a = 20, int b = 10  
static int a = 30, int b = 10  
static int a = 40, int b = 10  
static int a = 50, int b = 10  
*/
```

Storage classes in C

Storage Specifier	Storage	Initial value	Scope	Life
auto	stack	Garbage	Within block	End of block
extern	Data segment	Zero	global Multiple files	Till end of program
static	Data segment	Zero	Within block	Till end of program
register	CPU Register	Garbage	Within block	End of block

SOURCE CODE and INPUT-OUTPUT:

***** copy your source code here with proper indentation

1. Demonstrate use of static storage specifier
2. Demonstrate use of extern storage specifier
3. Convert one of your lab works to multiple files by using extern



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Try them out:

S.no	Question	Answer:
1	What will be the output of following ? <pre>void main() { static num = 4; printf("%d ",--num); if(num) main(); }</pre>	
2	Which storage class is used for faster execution ?	
3	Storage class of a variable determines ?	
4	What is the output of following ? <pre>main() { extern int a; printf("\n%d",a); }</pre>	
5	What is the output of following? <pre>main() { extern int fun(float); int a; a=fun(3.14); printf("%d",a); } int fun(aa) float aa; { return((int)aa); }</pre>	
6	What will be the value of i and j in the following code ? <pre>main() { auto int i,j=6; printf("%d%d",i,j); }</pre>	
7	What different values of " i " following code prints ? <pre>int i; main() { printf("%d",i); increment(); }</pre>	



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH PART: 1 (2019-20)	LAB:	SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I
---	-------------	---

		<pre> decrement(); } increment() { i++; printf("%d",i); } decrement() { i--; printf("%d",i); } </pre>	
8		<pre> extern int s; int t; static int u; main() { } </pre> <p>Which of s, t and u are available to a function present in another file?</p>	
9		<pre> #include<stdio.h> void main() { static num = 4; printf("%d ",--num); if(num) main(); } </pre>	
10		<pre> #include<stdio.h> int main(){ register int a=10; int *p; p=&a; printf("%u",p); return 0; } </pre>	
11		<pre> #include<stdio.h> </pre>	



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

```
void fn()
{
    static int i=5;
    printf("%d\t",++i);
}
int main()
{
    fn();
    fn();
    return 0;
}
```

12

```
#include <stdio.h>
static int i=10;
int main(){
    i=5;
    for(i=0;i<5;i++){
        static int a=10;
        printf("%d\t",a++);
    }
    return 0;
}
```

ASSESSMENT QUESTIONS:

1. What are the problems associated with using extern variables?

Courtesy:

<https://www.2braces.com/>



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

LAB WORK#8

NAME:
ROLL NO:
BATCH:

AIM: To demonstrate the hardware communication through C programs.

OBJECTIVE: The objective of this lab work is:

1. To demonstrate hardware communication through C program
2. Communication of C program and Keyboard

OUTCOMES: After completing the Lab Work#8 students will be able to:

1. Apply simple hardware communication logic to communicate to hardware device- keyboard
2. Understand hardware communication programming through C.

PRE-REQUISITE:

1. Basic syntax of C.
2. Hardware communication with C programs

INPUT-OUTPUT:

1. Input -As discussed in lab
2. Output- As discussed in lab

THEORY:

Visit the link for details

<http://www.dtic.mil/dtic/tr/fulltext/u2/a207317.pdf>

SOURCE CODE:

INPUT-OUTPUT:

***** Paste the screenshot of your output window (invert colors to black text on white background)

ASSESSMENT QUESTIONS:

Can you illustrate usage of C in embedded systems – along with a small example program(optional)



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Lab Work#9

TITLE OF PROJECT

NAME OF TEAM MEMBERS:

ROLL NOS:

AIM: To create a mini project on the selected topic, which will apply all the learning from the subject Advanced C (CSE 224). They will present the work

OBJECTIVE: The objective of this lab work is:

1. To enable students, apply the topics and programming learned in the subject CSE 224- Advanced C in to a mini project on the topic of their choice approved by the subject in-charge.

OUTCOMES: After completing the Lab Work#9 students will be able to:

1. Get an idea of application of topics done in the subject CSE 224 Advanced C in a broader perspective.
2. Evaluate which programming elements are to be used to prepare a programming solution for their selected topic
3. Students will documentation norms.
4. Will look out for some problem and create a programming solution in C programming Language.
5. Enable them to take up projects in future and work in teams.

PRE-REQUISITE:

1. Course of CSE 224 -Advanced C

INPUT-OUTPUT:

3. Input -As per their problem statement
4. Output- As per their problem statement

SOURCE CODE:

***** copy your source code here with proper indentation

INPUT-OUTPUT:

***** Paste output

DIAGRAMATIC REPRESENTATION:

Paste the photo of diagrammatic representation prepared



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Problem Statement#0

FasterTrack is a company which manufactures analog wrist watches. There is a speed with which its internal gears need to move to keep watch times and speed up-to-date. FasterTrack recently launched its new model of sports watch- "Brodis". Usually, the life span of fasterTrack Watch is more than 10 years, i.e. It should work for atleast 10 years. Unfortunately, a serious bug was detected while manufacturing of 'Brodis' model was about to complete, due to which the **RPM increases or decreases every year** in the following manner:



- multiply the number 323 to sum of the squares of the digits of the RPM,
- shift the digits of the RPM to the right by 1 position in a cyclic way, and finally
- extract the last two digits of the new number obtained and add to the result obtained in 'a'.
- The result (c) is the new RPM.

This issue is that if the RPM increases by a certain limit, then the watch disk crashes, which is also dangerous to the person who is using it. **The maximum limit of the RPM of the watch is 8 times of the RPM**, i.e. if RPM of watch is 1000, then the maximum RPM can be $1000 * 8 = 8000$.

This issue needs to be resolved before they release the 'Brodis' model of FasterTrack to the general public. You are given a contract to find out (based on the RPM) which watch will last for more than 10 years, and which will not.

Assume that the RPM is between 3524 and 8524 only, both inclusive.

Hint:

Task: In this program, the value of RPM is taken from the user using scanf() statement. You are required to write code that does the following:

1. Find out the sum of the squares of the digits of the RPM
2. Multiply the number obtained in (1) by 323
3. Do a cyclic right shift of digits of the RPM i.e. if Number is 1234, after cyclic right shift, the number will be 4123. Thereafter, take the last two digits of the number obtained just now, and add it to the number obtained in point 2. Thus, obtaining a new RPM value.
4. Do these steps till the number of years are 10 or the watch has reached the maximum RPM



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Example 1: Assume an **RPM of 5524**. So, the **maximum limit of RPM is $5524 * 8 = 44192$**

RPM	Years	SSD	A SSD * 323	B Cyclic Right Shift	A + last 2 digits of B	Result (Next RPM)
5524	1	$4^2 + 2^2 + 5^2 + 5^2$ $= 16 + 4 + 25 + 25$	$70 * 323$ $= 22610$	4552	$22610 + 52$	22662
22662	2	$2^2 + 6^2 + 6^2 + 2^2 + 2^2$ $= 4 + 36 + 36 + 4 + 4$	$84 * 323$ $= 27132$	22266	$27132 + 66$	27198
27198	3	$8^2 + 9^2 + 1^2 + 7^2 + 2^2$ $= 64 + 81 + 1 + 49 + 4$	$199 * 323$ $= 64277$	82719	$64277 + 19$	64296

Stop computing further as the maximum limit of RPM is 44192, and the RPM after 3 years is 64296

****Courtesy: IITBombayX: CS101.1x Introduction to Computer Programming**



G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Problem Statement#1

"Gringotts, the wizard bank' recently opened its branch in Hogwarts school of witchcraft and wizardry.



They provide following functionality in the bank:

1. Open Account
2. Deposit Money
3. Withdraw Money
4. Change Account Details

Harry Potter is newly enrolled student in the school and is very famous due to all the history. To keep an eye on him the house master Professor McGonagall decided to give Harry the task of creating a C program for above bank operations. McGonagall gave following specifications:

- There are total 1000 Students in the school.





**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

- Each bank account must have following details:
 - Account number (Generated automatically and Unique)
 - Account holders – first and last name
 - Nominee's- first and last name
 - Address- room no, hostel name, floor
 - Initial Amount deposit
 - Also, there is a field where they key keep any one of the following values:
 - Lock type- any magical security
 - Password
 - List of wands, maximum 5

You are Ron Weasley, you and Hermione decided to help Harry in this task



Problem Statement#2

You created the program for "Gringotts, the wizard bank" and the program was running fine for the school, but Prof. Snape came to you with a problem.

The number of students in the school is not fixed (1000 as stated earlier), now it can increase or decrease every year. So, he told you to modify the program as per the need. As the school has limited memory so you were asked to keep care that memory is utilized in optimum way, you decided to apply Dynamic Memory Allocation which





**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

is recently taught in the class.

For a sample you simplified the account structure with only following details:

1. Account number
2. Balance

And Operation options as:

1. Create Account
2. Display Details
3. Withdraw
4. Deposit



Recreate the program and impress Prof. Snape.



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Problem Statement#3



www.clipartof.com · 43224

Your friend Albert Einstein has recently joined college after a long break. It's mid-semester and he is facing lots of problems to cope-up with the syllabus. In your Advanced C lectures the topic of bitwise operators is going on and Albert is facing lots of difficulties in writing programs using bitwise operators. Teacher gave him extra programs to practice at home and submit next day.

You decided to help him in programming and following list of programs was forwarded by Einstein, try solving these problems strictly by using Bitwise Operators:

1. To check if Nth Bit is set/ reset.
2. To check if entered number is even or odd.
3. To count number of zero's in entered numbers' binary equivalent
4. To count number of one's in entered numbers' binary equivalent
5. To rotate bits of a number.
6. Swap two numbers
7. Find Two's Complement of entered number
8. Convert Uppercase to Lower Case
9. C Program to round Floor of integer to next Lower Power of 2
10. C Program to Reverse all the Bits of a 32-bit Integer using Bitwise

Help Einstein!!





**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Problem Statement#4

Steve Jobs is a very lazy fellow. He owns a big business but still very sluggish in maintaining his documents.

You recently joined Steve's Company – 'Half Apple' as an intern. You saw that there are few files received via email which need to be read and there is some material which need to be written and saved on a file.

There are some alternate task as well- like counting a particular thing or similar.

You decided to create a c program which will do the following:

Show options:

1. Read a File
2. Write a File
3. Count specified character.

User may select any of these options and perform the task.

For any of these you need to mention the file name on which operation is to be performed.



Go help Steve organize his task !!



**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURANGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Problem Statement#5

Mohan, an office clerk is working on the DOS based systems since his career started and is familiar with the common commands of his use. Recently he is shifted to a new office where they have GUI based systems, and he is facing lots of confusion on how to move cursor, what are all these icons..... and the directory structure is going above his head.

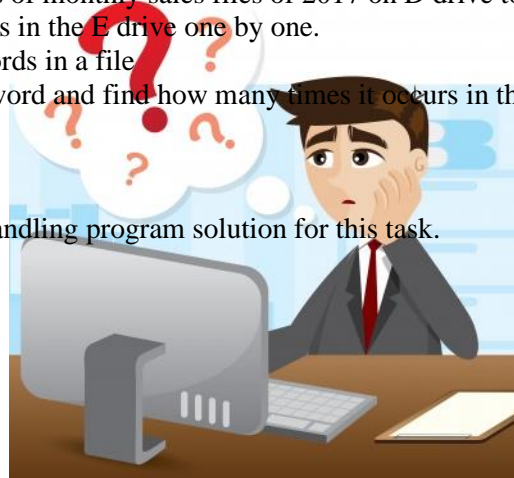


shutterstock.com • 337571150

This week's task is:

1. To copy the contents of monthly sales files of 2017 on D drive to the different named files in the E drive one by one.
2. Count number of words in a file
3. Enter a three-letter word and find how many times it occurs in the file

Try to come up with a file handling program solution for this task.





G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CLASS: S.Y. B. TECH
PART: 1 (2019-20)

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I

Problem Statement#6



After you developed that Gringotts Bank program for Professor Snape, using Dynamic Memory Allocation, the things were fine for few days and suddenly Professor McGonagall summoned you and Harry Potter in her office and told you that program works fine but when you re-execute it all the data goes away. The bank records need to kept stored somewhere and will be accessed and changed as per requirement.

Now you have a challenge- suddenly you got an idea of using File Handling in some way to solve the problem

Now sit back with your friends – Harry Potter and Hermione Granger and develop the Full-Proof Program for managing bank account of Gringotts Bank





**G.S. MANDAL'S
MAHARASHTRA INSTITUTE OF TECHNOLOGY, AURABGABAD**

LAB WORK INSTRUCTION SHEET

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**CLASS: S.Y. B. TECH
PART: 1 (2019-20)**

LAB:

SUBJECT: CSE 206- Advanced C, CSE 224: Computer Laboratory I