Semester-II

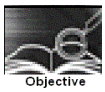
Assignment Completion Sheet

Sr. No.	Title of the Assignment	Performed on	Submitted on	Remark	Marks Obtained
	A) Advanced 'C' Programming				
1.	To demonstrate use of Pointer				
2.	To demonstrate dynamic memory allocation				
3.	To demonstrate String using standard library func				
4.	To demonstrate Structure and Unions.				
5.	File Handling.				
6.	To demonstrate C Preprocessors.				
	B) Relational Database Management System.				
1.	Assignment on Stored Function				
2.	Assignment on Cursor.				
3.	Assignment on Exception handling.				
4.	Assignment on Triggers				
5.	Assignment on Stored procedures				

Practical Incharge: 1)

2)

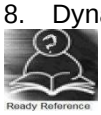
Exercise 1 **Start Date** / /



To demonstrate use of pointers in C.

You should read the following topics before starting this exercise

1. What is a pointer?
2. How to declare and initialize pointers.
3. '*' and '&' operators.
5. Pointer to a pointer.
6. Relationship between array and pointer.
7. Pointer to array and Array of pointers.
8. Dynamic memory allocation (malloc, calloc, realloc, free).



A Pointer is a variable that stores the memory address of another variable

Actions involving Pointers	**syntax**	**Example**
Declaration of pointers	data_type * pointer_name	int *p1,*p2; float *temp1;
Initialization of pointers	pointer =&variable p1=&n;	int a, *p= &a;
Pointer Arithmetic	The C language allow arithmetic operations to be performed on pointers: Increment, Decrement, Addition, Subtraction When a pointer is incremented (or decremented) by 1, it increments by sizeof(datatype). For example, an integer pointer increments by sizeof(int).	
Pointers and Functions	We can pass the address of a variable to a function. The function can accept this address in a pointer and use the pointer to access the variable's value.	
Arrays And Pointers	An array name is a pointer to the first element in the array. It holds the base address of the array. Every array expression is converted to pointer expression as follows: a[i] is same as *(a+i)	int n; *n , *(n + 0) represents 0^{th} element n[j], *(n+ j),* (j + n) , j [n] : represent the value of the j^{th} element of array n

Pointer To Pointer		int a; int * p; int **q; p = &a;

		q = *p ;

To allocate memory dynamically	The functions used are : malloc, calloc, realloc
	ptr = (cast-type *) malloc (byte-size) ;
	Allocates a block of contiguous bytes. If the space in heap is not sufficient to satisfy request, allocation fails, returns NULL.
	ptr1 = (cast-type *) calloc (byte-size);
	Similar to malloc, but initializes the memory block allocated to 0.
	ptr = realloc (ptr, new size);
	To increase / decrease memory size.
		int * p,*p1;
		p = (int *) malloc(10 * sizeof(int));
		p1 = (int *) calloc(10, sizeof(int));
		p1=realloc(p1,20*sizeof(int));

1. Sample program

```
/* Program to swap two numbers*/
main()
{
        int a = 10, b = 20;
        void swap1( int x, int y);
        void swap2( int *ptr1, int *ptr2);

        printf("\nBefore swapping : a=%d, b=%d", a,b);
        swap1(a, b);
        printf("\nAfter swapping by swap1 : a=%d, b=%d", a,b);
        swap2(&a, &b);
        printf("\nAfter swapping by swap2 : a=%d, b=%d", a,b);
}

void swap1( int x, int y)
{
        int temp;
        temp = x;
        x = y;
        y = temp;
}

void swap2( int *ptr1, int *ptr2)
{
        int temp;
        temp = *ptr1;
        *ptr1 = *ptr2;
        *ptr2 = temp;
}
```

2. Sample program

```
/* Program to  allocate memory for n integers dynamically*/
#include <stdlib.h>
void main()
{
        int *p, n,i;
        printf("How many elements :");
        scanf("%d",&n);

        p = (int *)malloc(n*sizeof(int));
        /* Accepting data */
        for(i=0; i<n;i++)
           scanf("%d",p+i);

        /* Displaying data */
        for(i=0; i<n;i++)
           printf("%d\t",*(p+i));
}
```


Self-Activity

1. Type the sample program 1 given above, execute it and write the output.

2. Sample program 2 allocates memory dynamically for n integers and accepts and displays the values. Type the sample program 2 given above, execute it. Modify the program to allocate memory such that the allocated memory is initialized to 0.


Assessment - work

Set A . Write C programs for the following problems.

1. Write a function which takes hours, minutes and seconds as parameters and an integer s and increments the time by s seconds. Accept time and seconds in main and Display the new time in main using the above function.

2. Write a program to display the elements of an array containing n integers in the reverse order using a pointer to the array.

3. Write a program to allocate memory dynamically for n integers such that the memory is initialized to 0. Accept the data from the user and find the range of the data elements.

Signature of the instructor | Date / /

Set B . Write C programs for the following problems.

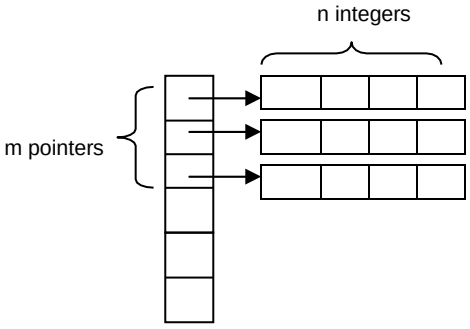
1. Accept n integers in array A. Pass this array and two counter variables to a function which will set the first counter to the total number of even values in the array and the other to the total number of odd values. Display these counts in main. (Hint: Pass the addresses of the counters to the function)

2. Write a function which accepts a number and three flags as parameters. If the number is even, set the first flag to 1. If the number is prime, set the second flag to 1. If the number is divisible by 3 or 7, set the third flag to 1. In main, accept an integer and use this function to check if it is even, prime and divisible by 3 or 7. (Hint : pass the addresses of flags to the function)

Signature of the instructor | Date / /

Set C. Write programs to solve the following problems

1. Accept the number of rows (m) and columns (n) for a matrix and dynamically allocate memory for the matrix. Accept and display the matrix using pointers. Hint: Use an array of pointers.



2. There are 5 students numbered 1 to 5. Each student appears for different number of subjects in an exam. Accept the number of subjects for each student and then accept the marks for each subject. For each student, calculate the percentage and display. (Hint: Use array of 5 pointers and use dynamic memory allocation)

Signature of the instructor ☐ Date / /

Assignment Evaluation		**Signature**			
0: Not done	☐	2: Late Complete	☐	4: Complete	☐
1: Incomplete	☐	3: Needs improvement	☐	5: Well Done	☐

Exercise 2-a **Start Date** / /

To demonstrate strings in C.

You should read the following topics before starting this exercise

1. String literals
2. Declaration and definition of string variables
3. The NULL character
4. Accepting and displaying strings
5. String handling functions

A string is an array of characters terminated by a special character called NULL character(\0). Each character is stored in 1 byte as its ASCII code.

Actions Involving strings	**Explanation**	**Example**
Declaring Strings		char message[80];
Initializing Strings		char message[]= { 'H', 'e', 'l', 'l', 'o', '\0' } ; char message [ ] = "Hello";

Accepting Strings	scanf and gets can be used to accept strings	char name[20], address[50];
		printf("\n Enter your name :);
		scanf("%s", name);
		printf("\n Enter your address :);
		gets(address);

Displaying Strings	printf and puts can be used to display strings.	printf("\n The name is %s:", name);
		printf("\n The address is :");
		puts(address);

String functions	All string operations are performed using functions in "string.h". Some of the most commonly used functions are
	a. strlen – Returns the number of characters in the string (excluding \0)
	b. strcpy – Copies one string to another
	c. strcmp – Compares two strings. Returns 0 (equal), +ve (first string > second), -ve (first string < second). It is case sensitive
	d. strcmpi – Same as strcmp but ignores case
	e. strcat – Concatenates the second string to the first.
		#include <string.h>
		main( )
		{
		char str1[50], str2[50],str3[100];
		printf("\n Give the first string:");
		gets(str1);
		printf("\n Give the second string string:");
		gets(str2);
		if (strlen(str1) == strlen(str2)
		{strcpy(str3, strrev(str1));
		strcat(str3, strupr(str2));
		puts(strupr(str3));
		}
		else
		puts(strlwr(str2);
		}

	Returns the concatenated string. f. strrev – Reverses a string and returns the reversed string. g. strupr – Converts a string to uppercase. h. strlwr - Converts a string to lowercase	

Sample program :

The following program demonstrates how to pass two strings to a user defined function and copy one string to other using pointers

```
void string_copy (char *t,char *s)
{
        while (*s !='\0')          /* while source string does not end */
        {       *t=*s;
                s++;
                t++;
        }
        *t ='\0';  /*       terminate target string */
}

void main()
{
        char str1[20], str2[20];
        printf("Enter a string :");
        gets(str1);
        string_copy(str2, str1);
        printf("The copied string is :");
        puts(str2);
}
```


Self-Activity

1. Write a program to accept two strings str1 and str2. Compare them. If they are equal, display their length. If str1 < str2, concatenate str1 and the reversed str2 into str3. If str1 > str2, convert str1 to uppercase and str2 to lowercase and display. Refer sample code for string functions above.

2. Type the sample program above and execute it. Modify the program to copy the characters after reversing the case. (Hint: First check the case of the character and then reverse it)

Signature of the instructor _____ Date / /


Assessment - work

Set A . Write C programs for the following problems.

1. Write a menu driven program to perform the following operations on strings using standard library functions:

Length	Copy	Concatenation	Compare
Reverse	Uppercase	Lowercase	Check case

2. Write a program that will accept a string and character to search. The program will call a function, which will search for the occurrence position of the character in the

string and return its position. Function should return –1 if the character is not found in the string.

3. A palindrome is a string that reads the same-forward and reverse. *Example*: “madam” is a Palindrome. Write a function which accepts a string and returns 1 if the string is a palindrome and 0 otherwise. Use this function in main.

4. For the following standard functions, write corresponding user defined functions and write a menu driven program to use them. strcat, strcmp, strrev, strupr

5. Write a program which accepts a sentence from the user and alters it as follows: Every space is replaced by *, case of all alphabets is reversed, digits are replaced by ?

Signature of the instructor | Date / /

Set B . Write C programs for the following problems.

1. Write a menu driven program which performs the following operations on strings. Write a separate function for each option. Use pointers
   i. Check if one string is a substring of another.
   ii. Count number of occurrences of a character in the string.
   iii. Replace all occurrences of a character by another.

2. Write a program in C to reverse each word in a sentence.

3. Write a function which displays a string in the reverse order. (Use recursion)

Signature of the instructor | Date / /

Set C. Write programs to solve the following problems

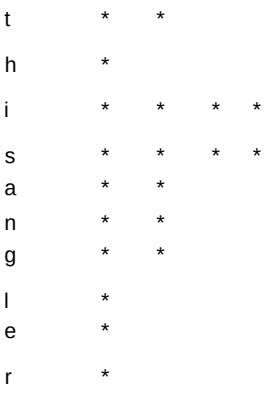
1. Write a program that accepts a sentence and returns the sentence with all the extra spaces trimmed off. (In a sentence, words need to be separated by only one space; if any two words are separated by more than one space, remove extra spaces)

2. Write a program that accepts a string and displays it in the shape of a kite. Example: “abcd” will be displayed as :

aa
abab
abcabc
abcdabcd
abcabc
abab
aa

3. Write a program that accepts a string and generates all its permutations. For example: ABC, ACB, BAC, BCA, CAB, CBA

4. Write a program to display a histogram of the frequencies of different characters in a sentence. Note: The histogram can be displayed as horizontal bars constructed using * character. Example: this is a single string



Signature of the instructor _____ Date / /

Assignment Evaluation **Signature**

0: Not done ☐	2: Late Complete ☐	4: Complete ☐
1: Incomplete ☐	3: Needs improvement ☐	5: Well Done ☐

Exercise 2.b **Start Date** / /

To demonstrate array of Strings.

You should read the following topics before starting this exercise
1. How to declare and initialize strings.
2. String handling functions
3. How to create and access an array of strings.
4. Dynamic memory allocation

An array of strings is a two dimensional array of characters. It can be treated as a 1-D array such that each array element is a string.

Actions Involving array of strings	**Explanation**	**Example**
Declaring String array	char array[size1][size2];	char cities[4][10]
Initializing String array		char cities[4][10] = { “Pune”, “Mumbai”, “Delhi”, “Chennai”};

Sample program-

The following program illustrates how to accept ‘n’ names , store them in an array of strings and search for a specific name.

```
/* Program to search for name from array */
#include <stdio.h>
void main( )
{
        char list[10][20]; /*list is an array of 10 strings */
        char name[20];
        int i,n;
        printf(“\n How many names ?:”);
        scanf(“%d”, &n);
        for (i=0;i<n; i++)
          {
                    printf(“\n Enter name %d,”i);
                    scanf(“%s”, list[i]);
          }
        printf(“\n The names in the list are : \n”);
        for (i=0; i<n; i++)
                    printf(”%s”, list[i]);
        printf(“\n Enter the name to be searched “);
        scanf(“%s”, name);
        for (i=0; i<n; i++)
          if(strcmp(list[i],name)==0)
               {
                  printf(“Match found at position %d”, i);
                  break;
               }
        if(i==n)
         printf(“Name is not found in the list”);
}
```

1. Type the above sample program and execute the same for different inputs.

Signature of the instructor | | Date | / /

Set A . Write C programs for the following problems.

1. Write a program that accepts n words and outputs them in dictionary order.
2. Write a program that accepts n strings and displays the longest string.
3. Write a program that accepts a sentence and splits the sentence into words. Sort each word and reconstruct the sentence.

Input – this is a string Output – hist is a ginrst

Signature of the instructor | | Date | / /

Set B . Write C programs for the following problems.

1. Write a function, which displays a given number in words.

For Example: 129 One Hundred Twenty Nine
2019 Two Thousand Nineteen

2. Define two constant arrays of strings, one containing country names (ex: India, France etc) and the other containing their capitals. (ex: Delhi, Paris etc). Note that country names and capital names have a one-one correspondence. Accept a country name from the user and display its capital. Example: Input: India , Output: Delhi.

Signature of the instructor | | Date | / /

Set C. Write programs to solve the following problems

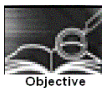
1. Create a mini dictionary of your own. Each entry in the dictionary contains three parts (word, its meaning, similar word). The entries are stored in the sorted order of words. Write a menu driven program, which performs the following operations.
   i. Add a new word (Insert new word and its details in the correct position)
   ii. Dictionary look-up
   iii. Find similar word
   iv. Delete word
   v. Display All words starting with a specific alphabet (along with their meaning).

(Hint: Use 2-D array of strings having n rows and 3 columns)

Signature of the instructor | | Date | / /

Assignment Evaluation		**Signature**			
0: Not done		2: Late Complete		4: Complete	
1: Incomplete		3: Needs improvement		5: Well Done	

Exercise 3-a **Start Date** / /



Structures in C

You should read the following topics before starting this exercise

1. Concept of structure
2. Declaring a structure
3. Accessing structure members
4. Array of structures
5. Pointer to a structure.
6. Passing structures to functions

A structure is a composition of variables possibly of different data types, grouped together under a single name. Each variable within the structure is called a 'member'.

Operations performed	**Syntax / Description**	**Example**

Declaring a structure	struct structure-name
	{
	type member-1 ;
	type member-2;
	.
	.
	type member-n ;
	};
		struct student
		{
		char name[20];
		int rollno;
		int marks;
		};

Creating structure variables	struct structurename variable;	struct student stud1;
Accessing structure members	variable.member	stud1.name stud1.rollno stud1.marks
initializing a structure variable	the initialization values have to be given in {} and in order	struct student stud1 = {"ABCD",10,95};
Pointer to a structure	struct structure-name * pointer-name;	struct student *ptr; ptr = &stud1;
Accessing members using Pointer	pointer-name -> member-name;	ptr->name; ptr->rollno;
Array of structures	struct structure-name array-name[size];	struct student stud[10];
passing Structures to Functions	return-type function-name (struct structure-name variable);	void display(struct student s);
pass an array of structures to a function	return-type function-name (struct structure-name array[size]);	void display(struct student stud[10]);

Sample Code:

```
/* Program for student structure  */

#include<stdio.h>
struct student
{       char name[20];
        int rollno;
        int marks[3];
        float perc;
};
void main( )
{
        int i, sum j;
        struct student s[10];
        printf("\n Enter the details of the 10 students \n");
        for (i=0;i<10;i++)
        {
                printf("\n Enter the name and roll number \n");
                scanf("%s%d",s[i].name, &s[i].rollno);
                printf("\n Enter marks for three subjects:");
                sum = 0 ;
                for { j=0;j<3;j++)
                {
                        scanf("%d",&s[i].marks[j]);
                        sum  = sum + s[i].marks[j];
                 }
                 s[i].perc = (float)sum/3;
        }
        /* Display details of students */
        printf("\n\n Name \t Roll no\t Percentage");
        printf("\n================================\n");
        for(i=0;i<10;i++)
        {
                printf("\n%s\t%d\t%f",s[i].name,s[i].rollno,s[i].perc);
        }
}
```


Self-Activity

1. The program in Sample code 1 demonstrates an array of structures of the type student. Type the above program and run it. Modify the program to display the details of the student having the highest percentage.

Signature of the instructor _____ Date / /


Assessment - work

Set A . Write C programs for the following problems.

1. Create a structure student (roll number, name, marks of 3 subjects, percentage)**.** Accept details of n students and write a menu driven program to perform the following operations. Write separate functions for the different options.

i) Search
ii) Modify
iii) Display all student details
iv) Display all student having percentage > _____
v) Display student having maximum percentage

2. Create a structure employee (id, name, salary)**.** Accept details of n employees and write a menu driven program to perform the following operations. Write separate functions for the different options.

i) Search by name
ii) Search by id
iii) Display all
iv) Display all employees having salary > _____
v) Display employee having maximum salary

Instructor should fill in the blanks with appropriate values.

Signature of the instructor | | Date | / /

Set B . Write C programs for the following problems.

1. Create a structure having the following fields:

Structure name:_____
Fields:_____,_____,_____, _____,_____,_____

Accept details of n variables of the above structure and write a menu driven program to perform the following operations. Write separate functions for the different options.

i)_____ ii) _____iii)_____iv) _____

2. Create a structure Fraction (numerator, denominator)**.** Accept details of n fractions and write a menu driven program to perform the following operations. Write separate functions for the different options. Use dynamic memory allocation. Note: While accepting fractions, store the fractions in the reduced form.

i) Display the largest fraction
ii) Display the smallest fraction
iii) Sort fractions
iv) Display all

Signature of the instructor | | Date | / /

Set C. Write programs to solve the following problems

1. Accept book details of 'n' books viz, book title, author, publisher and cost. Assign an accession numbers to each book in increasing order. (Use dynamic memory allocation). Write a menu driven program for the following options.
   i. Books of a specific author
   ii. Books by a specific publisher
   iii. All books having cost >= _____.
   iv. Information about a particular book (accept the title)
   v. All books.

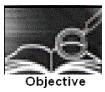
2. The government of a state wants to collect census information for each city and store the following information : city name, population of the city, literacy percentage. After collecting data for all cities in the state, the government wants to view the data according to :
   i. Literacy level
   ii. Population
   iii. Details of a specific city.

Write a C program using structures and dynamic memory allocation.

Signature of the instructor | | Date | / /

Assignment Evaluation			**Signature**		
0: Not done		2: Late Complete		4: Complete	
1: Incomplete		3: Needs improvement		5: Well Done	

Exercise 3-b **Start Date** / /



Nested Structures and Unions

You should read the following topics before starting this exercise
1. Creating and accessing structures
2. Array of structures
3. Dynamic memory allocation
4. Structure within a structure
5. Creating and accessing unions

Nested structures: The individual members of a structure can be other structures as well. This is called nesting of structures.

Operations performed	**Syntax**	**Example**

Creating a nested structure	struct structure1
	{
	. . .
	struct structure2
	{
	. . .
	} variable;
	. . .
	};
	
	Method 2
	struct structure2
	{
	. . .
	};
	
	struct structure1
	{
	. . .
	struct structure2
	variable;
	. . .
	};
		struct student
		{
		int rollno; char name[20];
		struct date
		{
		int dd, mm, yy;
		} bdate, admdate;
		};
		
		struct date
		{
		int dd, mm, yy;
		};
		
		struct student
		{
		int rollno; char name[20];
		struct date bdate, admdate;
		};

Accessing nested structure members	nested structure members can be accessed using the (.) operator repeatedly.	stud1.bdate.dd, stud1.bdate.mm
self referential structure	A structure containing a pointer to the same structure	struct node { int info; struct node *next;

		};

Unions	A union is a variable that contains multiple members of possibly different data types grouped together under a single name. However, only one of the members can be used at a time. They occupy the same memory area.	union u
		{
		char a;
		int b;
		};

Sample Code 1:
Example: The following structure is for a library book with the following details : id, title, publisher, code (1 – Text book, 2 – Magazine, 3 – Reference book). If the code is 1, store no-of-copies. If code = 2, store the issue month name. If code = 3, store edition number. Also store the cost.

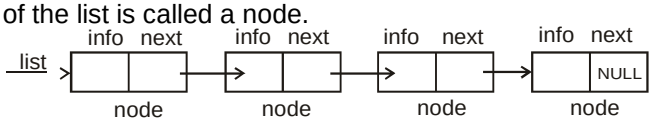
```
/* Program for demonstrating structure and union  */

struct library_book
{
        int id;
        char title[80],publisher[20] ;
        int code;
        union u
        {
                int no_of_copies;
                char month[10];
                int edition;
        }info;
        int cost;
};
void main( )
{
struct library_book book1;
printf("\n Enter the details of the book \n");

printf("\n Enter the id, title and publisher \n");
scanf("%d%s%s",&book1.id, book1.title, book1.publisher);
printf("\n Enter the code: 1-Text Book, 2-Magazine, 3-Reference");
        scanf("%d",book1.code);
switch(book1.code)
{
        case 1: printf("Enter the number of copies :");
                        scanf("%d",&book1.info.no_of_copies);
                        break;
    case 2:     printf("Enter the issue month name :");
                        scanf("%s",book1.info.month);
                        break;
    case 3:     printf("Enter the edition number:");
                        scanf("%d",&book1.info.edition);
                        break;
}
printf("Enter the cost :");
scanf("%d",&book1.cost);

/*      Display details of book */
printf("\n id = %d", book1.id);
```

```
printf("\n Title = %s", book1.title);
printf("\n Publisher = %s", book1.publisher);
switch(book1.code)
{
    case 1:        printf("Copies = %d:", book1.info.no_of_copies);
                        break;
     case 2:       printf("Issue month name = %s",book1.info.month);
                        break;
     case 3:       printf("Edition number =%d:",book1.info.edition);
                        break;
}
printf("\n Cost = %d", book1.cost);
}
```

Sample Code 2:

A linked list is a collection of data elements which are linked to one another by using pointers i.e. the every node stores the address of the next node. The advantage of using a linked list over an array is that it is easy to insert and delete elements from the list.

To create a linked list, we have to use a self referential structure (See table above). Each element of the list is called a node.



To create a node, we have to allocate memory dynamically. The following program creates 5 nodes , stores the numbers 1…5 in them and displays the data.

```
/* Program to create a linked list  of 5 nodes */

#include <stdio.h>
struct node
{
   int info;
   struct node *next;
};
struct node *list = NULL;  /* list is a pointer to the linked list */

void createlist()
{
  struct node *temp, *p;
  int i;
  for(i=1;i<=5;i++)
  {
     p=(struct node *)malloc(sizeof(struct node)); /* create a node */
     p->info = i;
     p->next=NULL;
    if(list == NULL)
         list=temp=p; /* list points to the first node */
    else
    {
        temp->next=p; /* link new node to the last */
        temp=p;
    }
}
void displaylist()
{
  struct node *temp;
```

```
   for(temp=list; temp!=NULL; temp=temp->next) /* use a temporary pointer */
        printf(%d \t", temp->info);
}

void main( )
{
 createlist();
 displaylist();
}
```


Self-Activity

1. The sample code 1 given above demonstrates how we can create a variable of the above structure and accept and display details of 1 book. Type the program and execute it. Modify the program to accept and display details of n books.

2. The sample code 2 given above demonstrates how we can create a linked list and traverse the list. Type the program and execute it. Modify the displaylist function to display only the even numbers from the list.

Signature of the instructor | | Date | / /


Assessment - work

Set A . Write C programs for the following problems.

1. Modify the sample program 1 above to accept details for n books and write a menu driven program for the following:

i) Display all text books
ii) Search Text Book according to Title
iii) Find the total cost of all books (Hint: Use no_of_copies).

2. Modify the sample program 1 to accept details for n books and write a menu driven program for the following:

i) Display all magazines
ii) Display magazine details for specific month.
iii) Find the “costliest” magazine.

3. Modify the sample program 1 to accept details for n books and write a menu driven program for the following:

i) Display all reference books
ii) Find the total number of reference books
iii) Display the edition of a specific reference book.

Signature of the instructor | | Date | / /

Set B. Write programs to solve the following problems

1. Create a structure named having the following fields:

Field name	Description

Write a menu driven program to perform the following operations :
i) _____ ii) _____iii)_____iv)_____ v) ___ _____

2. Write a program to create a linked list of n nodes and accept data from the user for each node. Display the list. Accept a number from the user and search for the element in the list.

Signature of the instructor | | Date | / / |

Set C. Write programs to solve the following problems

1. A shop sells electronic items. Each item has an id, company name, code (1-TV, 2-Mobile phones, 3-Camera) and cost. The following additional details are stored for each item.
   - TV - size, type (CRT-1 / LCD- 2 / Plasma-3)
   - Mobile Phone - type (GSM – 1 / CDMA – 2) , model number.
   - Camera – resolution, model number.

The shop wants to maintain a list of all items and perform the following operations for each of the item types:

i) Display all
ii) Search for specific item
iii) Sort according to cost

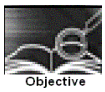
2. Write a program to create a linked list of n nodes and accept data from the user for each node. Write a menu driven program to perform the following operations:

i) Display the list
ii) Search for specific number
iii) Display the element after _____
iv) Find the maximum / minimum

Signature of the instructor | | Date | / / |

Assignment Evaluation			**Signature**		
0: Not done		2: Late Complete		4: Complete	
1: Incomplete		3: Needs improvement		5: Well Done	

Exercise 4-a **Start Date** / /



To demonstrate files using C

You should read the following topics before starting this exercise

1. Concept of streams
2. Declaring a file pointer
3. Opening and closing a file
4. Reading and Writing to a text file
5. Command line arguments

Operations performed	**Syntax**	**Example**
Declaring File pointer	FILE * pointer;	FILE *fp;
Opening a File	fopen(“filename”,mode); where mode = “r”, “w”, “a”, “r+”, “w+”, “a+”	fp=fopen(“a.txt”, “r”);
Checking for successful open	if (pointer==NULL)	if(fp==NULL) exit(0);
Checking for end of file	feof	if(feof(fp)) printf(“File has ended”);
Closing a File	fclose(pointer); fcloseall();	fclose(fp);

Character I/O	fgetc, fscanf
	fputc, fprintf
		ch=fgetc(fp);
		fscanf(fp, ”%c”,&ch);
		fputc(fp,ch);

String I/O	fgets, fscanf fputs, fprintf	fgets(fp,str,80); fscanf(fp, ”%s”,str);
Reading and writing formatted data	fscanf fprintf	fscanf(fp, ”%d%s”,&num,str); fprintf(fp, “%d\t%s\n”, num, str);
Random access to files	ftell, fseek, rewind	fseek(fp,0,SEEK_END); /* end of file*/ long int size = ftell(fp);

Sample Code 1

The following program reads the contents of file named a.txt and displays its contents on the screen with the case of each character reversed.

```
/* Program revrese case of characters in a file  */

#include <stdio.h>
#include <ctype.h>
void main()
{
        FILE * fp;
        fp = fopen(“a.txt”, “r”);
        if(fp==NULL)
        {
                printf(“File opening error”);
```

```
                exit(0);
        }
        while( !feof(fp))
        {
                ch = fgetc(fp);
                if(isupper(ch))
                      putchar(tolower(ch));
               else
                    if(islower(ch))
                          putchar(toupper(ch));
                    else
                          putchar(ch);
        }
        fclose(fp);
}
```

Sample Code 2

The following program displays the size of a file. The filename is passed as command line argument.

```
/* Program to display size of a file */

#include <stdio.h>
void main(int argc, char *argv[])
{
        FILE * fp;
        long int size;
        fp = fopen(argv[1], “r”);
        if(fp==NULL)
        {
                printf(“File opening error”);
                exit(0);
        }
        fseek(fp, 0, SEEK_END); /* move pointer to end of file */
        size = ftell(fp);
        printf(“The file size = %ld bytes”, size);
        fclose(fp);
}
```

Sample Code 3

The following program writes data (name, roll number) to a file named student.txt , reads the written data and displays it on screen.

```
#include <stdio.h>
void main()
{
        FILE * fp;
        char str[20]; int num;
        fp = fopen(“student.txt”, “w+”);
        if(fp==NULL)
        {
                printf(“File opening error”);
                exit(0);
        }
        fprintf(fp,“%s\t%d\n”, “ABC”, 1000);
        fprintf(fp,“%s\t%d\n”, “DEF”, 2000);
        fprintf(fp,“%s\t%d\n”, “XYZ”, 3000);
```

```
        rewind(fp);
        while( !feof(fp))
        {
                fscanf(fp,"%s%d", str, &num);
                printf("%s\t%d\n", str, num);
        }
        fclose(fp);
}
```

1. Create a file named a.txt using the vi editor. Type the sample program 1 given above and execute the program. Modify the program to accept a character from the user and count the total number of times character occurs in the file.

2. Type the sample program 2 above and execute it. Modify the program to display the last n characters from the file.

3. Type the sample program 3 above and execute it. Modify the program to accept details of n students and write them to the file. Read the file and display the contents in an appropriate manner.

Signature of the instructor | | Date | / /

Set A . Write C programs for the following problems.

1. Write a program to accept two filenames as command line arguments. Copy the contents of the first file to the second such that the case of all alphabets is reversed.

2. Write a program to accept a filename as command line argument and count the number of words, lines and characters in the file.

3. Write a program to accept details of n students (roll number, name, percentage) and write it to a file named "student.txt". Accept roll number from the user and search the student in the file. Also display the student details having the highest percentage.

Signature of the instructor | | Date | / /

Set B. Write programs to solve the following problems

1. A file named numbers.txt has a set of integers. Write a C program to read this file and convert the integers into words and write the integer and the words in another file named numwords.txt.

Example:

numbers.txt	numwords.txt
11	Eleven
261	Two hundred Sixty One
9	Nine

2. Write a program which accepts a filename and an integer as command line arguments and encrypts the file using the key. (Use any encryption algorithm)

Signature of the instructor | | Date | / /

Set C . Write C programs for the following problems.

1. A text file contains lines of text. Write a program which removes all extra spaces from the file.

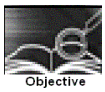
2. Write a menu driven program for a simple text editor to perform the following operations on a file, which contains lines of text.
   i. Display the file
   ii. Copy m lines from position n to p
   iii. Delete m lines from position p
   iv. Modify the nth line
   v. Add n lines

3. Write a program which reads the contents of a C program and replaces all macros occurring in the program with its value. Assume only simple substitution macros (ex: #define FALSE 0).

Signature of the instructor | | Date | / /

Assignment Evaluation			**Signature**		
0: Not done		2: Late Complete		4: Complete	
1: Incomplete		3: Needs improvement		5: Well Done	

Exercise 4-b **Start Date** / /



To demonstrate binary file handling using C.

You should read the following topics before starting this exercise

1. Concept of streams
2. Declaring a file pointer
3. Opening and closing files
4. File opening modes
5. Random access to files
6. Command line arguments

In binary files, information is written in the form of binary . All data is written and read with no interpretation and separation i.e. there are no special characters to mark end of line and end of file.

I/O operations on binary files

Reading from a binary file	fread(address,size-of-element,number of elements,pointer);	fread (&num,sizeof(int),1,fp);
		fread (&emp,sizeof(emp),1,fp);
		fread(arr,sizeof(int),10,fp);

Writing to a binary file	fwrite(address,size-of-element,number of elements,pointer);	fwrite (&num,sizeof(int),1,fp); fwrite (&emp,sizeof(emp),1,fp);

Sample Code

```
/* Program to demonstrate binary file */

struct employee
{       char name[20];
        float sal;
};
main( )
{
        FILE *fp;
        struct employee e;
        int i;
        if((fp=fopen (“employee.in”,“wb”))==NULL)
                {       printf(“Error opening file”);
                        exit( );
                }

        for(i=0;i<5;i++)
        {
                printf(”\n Enter the name and salary”);
                scanf(“%s%f”,e.name,&e.sal);
                fwrite(&e,sizeof(e),1,fp);
        }
```

```
        fclose(fp);

        fp=fopen("employee.in","rb");      /*        reopen file */
        if(fp==NULL)
        {       fprintf(stderr, "Error opening file);
        exit( );
        }
        for(i=0;i<5;i++)
        {
                fread(&e,sizeof(e),1,fp);
                printf("\n Name = %s Salary = %f",e.name,e.sal);
        }
        fclose(fp);
}
```


Self-Activity

1. Type program given above, writes data of 5 employees to a binary file and then reads the file. Modify the program to search an employee by name.

Signature of the instructor | | Date | / / |


Assessment - work

Set A . Write C programs for the following problems.

1. Create a structure student (roll number, name, percentage) Write a menu driven program to perform the following operations on a binary file- "student.dat". Write separate functions for the different options.
    1. Add a student (Note: Students should be assigned roll numbers consecutively)
    2. Search Student
        a. according to name
        b. according to roll number
    3. Display all students
2. Create a structure student (roll number, name, percentage) Write a menu driven program to perform the following operations on a binary file- "student.dat". Write separate functions for the different options.
    1. Add a student (Note: Students will be assigned roll numbers consecutively)
    2. Modify details
        a. according to name
        b. according to roll number
    3. Display all students
3. Create a structure student (roll number, name, percentage). Write a menu driven program to perform the following operations on a binary file- "student.dat". Write separate functions for the different options.
    1. Add a student (Note: Students will be assigned roll numbers consecutively)
    2. Delete student
        a. according to name
        b. according to roll number
    3. Display all students

Signature of the instructor | | Date | / /

Set B . Write C programs for the following problems.

1. Create two binary files such that they contain roll numbers, names and percentages. The percentages are in ascending orders. Merge these two into the third file such that the third file still remains sorted on percentage. Accept the three filenames as command line arguments.

2. Create a structure having the following fields:
Structure name:_____
Fields _____ _____
Store information for n variables of the above structure in a binary file. Write a menu driven program to perform the following operations Write separate functions for the different options.

i)_____ ii) _____iii)_____iv) _____

Signature of the instructor | | Date | / /

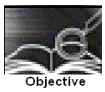
Set C . Write C programs for the following problems.

1. Create a binary file which contains details of student projects namely roll number, project name, project guide. The first line of the file contains an integer indicating the total number of students. When the program starts, read all these details into an array and perform the following menu driven operations. When the user selects Exit from the menu, store these details back into the file.
1. Add 2. Delete 3. Search 2. Modify 3. Display all 4. Exit

Signature of the instructor | | Date | / /

Assignment Evaluation			**Signature**		
0: Not done		2: Late Complete		4: Complete	
1: Incomplete		3: Needs improvement		5: Well Done	

Exercise 5 **Start Date** / /



Assignment to demonstrate command line arguments and preprocessor directives.

You should read the following topics before starting this exercise

1. Passing arguments from the command line to main
2. Accessing command line arguments
3. File inclusion, macro substitution and conditional compilation directives.
4. Argumented and Nested macros

Preprocessor directives	They begin with a # which must be the first non-space character on the line. They do not end with a semicolon.	
Macro Substitution Directive	# define MACRO value	# define PI 3.142
Argumented macro	# define MACRO(argument) value	# define SQR(x) x*x #define LARGER(x,y) ((x)>(y)?(x):(y))
Nested macro	one macro using another	#define CUBE(x) (SQUARE(x)*(x))
File Inclusion directive	#include <filename> #include "filename"	#include <stdio.h>
Conditional Compilation directive	# if, # else, # elif, # endif #ifdef	#ifdef PI #undef PI #endif

Command Line Arguments	int argc - argument counter
	char *argv[]-argument vector
		void main(int argc, char *argv[])
		{
		printf("There are %d arguments in all", argc);
		for (i=0; i<argc; i++)
		printf("Argument %d =%s",i,argv[i]);
		}

To run a program using command line arguments	Compile the program using cc Execute the program using a.out followed by command line arguments	Example: a.out ABC 20 Here, ABC and 20 are the two command line arguments which are stored in the form of strings. To use 20 as an integer, use function atoi . Example: int num = atoi(argv[2]);

Sample Code 1

```
/* Program for argumented macros  */

#define INRANGE(m) ( m >= 1 && m<=12)
#define NEGATIVE(m)  (m<0)
#define ISLOWER(c) (c>='a'&&c<='z')
#define ISUPPER(c)  (c>='A'&&c<='Z')
#define ISALPHA(c)  (ISUPPER(c)||ISLOWER(c))
#define ISDIGIT(c)  (c>='0'&&c<='9')

void main()
{
   int m; char c;
   printf("Enter an integer corresponding to the month");
   scanf("%d",&m);
   if(NEGATIVE(m))
      printf("Enter a positive number");
   else
   if(INRANGE(m))
      printf("You Entered a valid month");

   printf("Enter a character :");
   c=getchar();
   if(ISAPLHA(c))
      printf("You entered an alphabet");
   else
   if(ISDIGIT(c))
      printf("You Entered a digit");
}
```


Self-Activity

1. Write a program to display all command line arguments passed to main in the reverse order. Hint: See table above.

2. Sample code 1 above demonstrates the use of argumented and nested macros. Type the program and execute it.

Signature of the instructor _____ Date / /


Assessment - work

Set A . Write C programs for the following problems.

1. Write a program to accept three integers as command line arguments and find the minimum, maximum and average of the three. Display error message if invalid number of arguments are entered.

2. Write a program which accepts a string and two characters as command line arguments and replace all occurrences of the first character by the second.

3. Define a macro EQUALINT which compares two parameters x and y and gives 1 if equal and 0 otherwise. Use this macro to accept pairs of integers from the user. Calculate the sum ofdigits of both and continue till the user enters a pair whose sum of digits is not equal.

4. Define a macro EQUALSTR which compares two strings x and y and gives 1 if equal and 0otherwise. Use this macro to accept two strings from the user and check if they are equal.

Signature of the instructor | | Date | / /

Set B . Write C programs for the following problems.

1. Write a program to accept two strings as command line arguments and display the union and intersection of the strings. If the user enters invalid number of arguments, display appropriatemessage.

2. Write a program which accepts a string and an integer (0 or 1) as command line arguments.If the integer entered is 0, sort the string alphabetically in the ascending order and if it is 1, sort it in the descending order. If the user enters invalid number of arguments, display appropriate message. (Hint – use atoi)

Signature of the instructor | | Date | / /

Set C . Write C programs for the following problems.

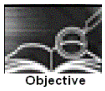
1. Create a header file “mymacros.h” which defines the following macros.
ii. SQR(x) ii. CUBE(x) - nested iii. GREATER2(x,y) iv. GREATER3 (x,y,z) – nested
v. FLAG (value = 1) (which may or may not be defined)

Include this file in your program. Write a menu driven program to use macros SQR, CUBE, GREATER2 and GREATER3. Your program should run the first two macros if the macro called FLAG has been defined. If it is not defined, execute the other two macros. Run the program twice
– with FLAG defined and with FLAG not defined.

Signature of the instructor | | Date | / /

Assignment Evaluation		**Signature**		
0: Not done		2: Late Complete		4: Complete	
1: Incomplete		3: Needs improvement		5: Well Done	

Exercise6 **Start Date** / /



Assignment to demonstrate bitwise operators.

You should read the following topics before starting this exercise
1. Bitwise operators and their usage (&, |, ^, ~, <<, >>)

1. **Bitwise operators:** C provides 6 operators to perform operations on bits. These operators operate on integer and character but not the float and double. Ones complement operator (~) isunary while the others are binary.

Operator	**Purpose**	**Example**
~	One's complement	~a : Complements each bit of variable a
	Right shift	a=a>>1; Shifts bits of a one position to the right
<<	Left Shift	a=a<<n; Shifts bits of a n positions to the left
&	Bitwise AND	a = b&c; performs bitwise AND on b and c a = a&0xFF00; Masks the lower order 8 bits of a
|	Bitwise OR	a = a!b; performs bitwise OR on b and c

^	Bitwise XOR	x = x^y; y=x^y; x=x^y; Swaps x and y by performing bitwise XOR.

Sample code: The following function accepts an integer argument and displays it in binaryformat. It uses shift operator and AND masking.

```
void displaybits(unsigned int n)
{
   unsigned int mask = 32768;
  /*set MSB of mask to 1 */
  while (mask>0)
  {
    if((n&mask)==0)
         printf("0");
   else
        printf("1");
    mask = mask >>1; /* shift mask right */
}
```


Self-Activity

1. Write a program to accept n integers and display them in binary. Use the function givenabove.

Signature of the instructor | | Date | / / |


Assessment - work

Set A . Write C programs for the following problems.

1. Write a program to accept 2 integers and perform bitwise AND, OR, XOR and Complement.Display the inputs and results in binary format. Use the function in the above exercise.

2. Write a program to swap two variables without using a temporary variable. (Hint: Use XOR)

3. Write a program which accepts two integers x and y and performs x<<y and x>>y. Displaythe result in binary.

Signature of the instructor | | Date | / / |

Set B . Write C programs for the following problems.

1. Write functions to calculate the size of an integer, character, long and short integer using

bitwise operators. Store their declaration in file “myfunctions.h” and their definitions in file “myfunctions.c”. Include these files in your program and use these functions to display the size ofeach.

2. Write a program to perform the following operations on an unsigned integer using bitwiseoperators and display the result in hexadecimal format.
   i. Swap the_____and____ nibble (4 bits)
   ii. Remove the lower order nibbles from the number.
      For example: Input: A3F1 Output 00A3
   iii. Reverse the nibbles
      For example: Input: A3F1 Output 1F3A

3. Write a program which accepts an integer and checks whether it is a power of 2.

Signature of the instructor | | Date | / /

Set C. Write programs to solve the following problems

1. Write a program to add, subtract, multiply and divide two integers using bitwise operators.
2. Packing and Unpacking Data: A date consists of three parts : day, month, year. To store thisinformation, we would require 3 integers. However, day and month can take only limited values. Hence, we can store all three in a single integer variable by packing bits together. If we are usingthe dd-mm-yy format, the date will be stored in memory as an unsigned integer (16 bits) in the following format. Year (Bits 15-9), Month (bits 8 – 5), Day (Bits 4 - 0).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
y	y	y	y	y	y	y	m	m	m	m	d	d	d	d	d
hour							Month				day				

Accept day, month and year from the user and pack them into a single unsigned int. Unpack anddisplay them in the binary format. (Hint: for packing, use: 512 * year + 32 * month + day)
The output should be:
Enter the date, month and year –dd mm
yy :31 12 89
Packed date =
1011001110011111Day = 31
0000000000011111

Month = 12
0000000000001100
Year = 89
0000000001011001

3. Packing and Unpacking Data: Time consists of three parts : hours, minutes, seconds. To store this information, we would require 3 integers. However, all these three variable take only limited values. Hence, we can store all three in a single integer variable by packing bits together. Time being 0 to 23 hours, it will require maximum 5 bits, minutes being 0 to 59 will require 6 bits. The two together take up 11 bits. The remaining 5 bits cannot store seconds which are also in therange 0 to 59 hence we store double seconds which are in the range 0 to 29

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
h	h	h	h	h	m	m	m	m	m	m	ds	ds	ds	ds	ds
hour					Minutes						Double seconds				

Accept hour, minute and double seconds from the user and pack them into a single unsigned int. Unpack and display them in the binary format.
The output should be:
Enter the hour, minutes and double seconds –hh mm ss :07 12 20
Packed date =
0011100110010100Hour = 07
0000000000000111
Minutes = 12
0000000000001100
Double seconds =
20
0000000000001010
0

Signature of the instructor _____ Date / /

Assignment Evaluation **Signature**

0: Not done ☐ 2: Late Complete ☐ 4: Complete ☐

1: Incomplete ☐ 3: Needs improvement ☐ 5: Well Done ☐