Department of Computer Science

Gujarat University

Certificate

Roll No: **30** Seat No: _____

This is to certify that Mr./Ms. ***Rathod Ajinkya*** student of MCA Semester – II has duly completed his/her term work for the semester ending in June 2020, in the subject of ***Advanced Programming*** towards partial fulfilment of his/her Degree of Masters in Computer Applications.

Date of Submission Internal Faculty

Head of Department

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. –II

R O L L NO : 30
N A M E : Ajinkya Rathod
S U B J E C T :Advanced Programming

NO.	TITLE	PAGE NO.	DATE	SIGN
	ASSIGNMENT:- 1	**7**		
1	**Create a structure name cricket and display the information**		**01/07/20**	
	team wise of a player.			
2	**Write a program to create a structure of a team i.e baseball**		**01/07/20**	
	team and football team and enter the details regarding it.			
3	**Write a program to create a structure of bank customer.**		**01/07/20**	
	And perform the functionality.			
4	**Write a program to maintain the inventory of the books in a**		**01/07/20**	
	bookshop. The details of the book using structure			
5	**Write a complete 'C' program that will accept the following**		**01/07/20**	
	information for each vehicle using structure			
	ASSIGNMENT:-2	**70**		
1	**Write a program using pointers to read array of integers**		**01/07/20**	
	and print its elements in reverse order.			
2	**Write a program using pointers to find minimum and**		**01/07/20**	
	maximum element of an array and display its address.			
3	**Write a program to count the number of vowels, consonants,**		**01/07/20**	
	digits and white space characters using pointers			
4	**WAP using pointers to implement the transpose of a matrix.**		**01/07/20**	
5	**WAP using pointers to implement the matrix multiplication.**		**01/07/20**	
6	**WAP to perform summation of a matrix using pointers.**		**01/07/20**	
7	**Write a program to sort the list of strings using pointers.**		**01/07/20**	
8	**Write a function using pointers to exchange the value stored**		**01/07/20**	

R O L L NO : 30
N A M E : Ajinkya Rathod
S U B J E C T :Advanced Programming

	in two locations in the memory			
	Perform following through pointers:-			
9	**Find the first occurrence of a character in the given**		**01/07/20**	
	string.The function should return the position in the string.			
10	**Find the first occurrence of a string in another string.The**		**01/07/20**	
	function should return the position in the string.			
11	**Delete all occurrences of a character from a string.**		**01/07/20**	
12	**Delete all occurrences of a string from another string**		**01/07/20**	
13	**Delete all occurrences of a character from a string. Ignore**		**01/07/20**	
	Case.			
14	**Delete all occurrences of a string from another string.**		**01/07/20**	
	Ignore Case			
15	**Copy one string to another string.**		**01/07/20**	
16	**Copy n characters of one string to another string.**		**01/07/20**	
17	**Find length of the string and toggle the characters of string**		**01/07/20**	
18	**Convert string to all upper case.**		**01/07/20**	
19	**Convert string to all lower case.**		**01/07/20**	
20	**Append one string to another string.**		**01/07/20**	
21	**Append at most n characters of one string to another string**		**01/07/20**	
22	**Reverse all the characters in the string.**		**01/07/20**	
23	**Compare two strings S1 and S2. The function should return**		**01/07/20**	
	-1, 0 or 1 if S1 < S2, S1 = S2 and S1 > S2 respectively.			
24	**Compare two strings S1 and S2. The function should return**		**01/07/20**	
	-1, 0 or 1 if S1 < S2, S1 = S2 and S1 > S2 respectively.			
25	**Compare at most n characters of two strings S1 and S2. The**		**01/07/20**	
	function should return -1, 0 or 1 if S1 < S2, S1 = S2 and S1 >			

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. –II

R O L L NO : 30
N A M E : Ajinkya Rathod
S U B J E C T :Advanced Programming

	S2 respectively			
26	**Compare at most n characters of two strings S1 and S2. The**		**01/07/20**	
	function should return -1, 0 or 1 if S1 < S2, S1 =S2 and S1 >			
	S2 respectively.Ignore case.			
	ASSIGNMENT:-3	**153**		
1	**Write a program to create a singly linked list and display its**		**01/07/20**	
	elements in FIFO pattern. Display the number of elements.			
2	**Write a program to create a singly linked list and display its**		**01/07/20**	
	elements in LIFO pattern. Display the number of elements.			
3	**create a singly linked list and perform: i) Insert an element**		**01/07/20**	
	ii)Delete an element iii) Display the list			
4	**Write a program to create an ordered linked list.**		**01/07/20**	
5	**Write a program to reverse a given linked list.**		**01/07/20**	
6	**Write a program to calculate the summation of all elements**		**01/07/20**	
	of the linked list.			
7	**Write a program to create two linked list and append the**		**01/07/20**	
	second list after the first.			
8	**Write a program to swap two consecutive elements of the**		**01/07/20**	
	given linked list. (Swap only values)			
9	**Write a program to swap two consecutive elements of the given linked list. (Swap only addresses)**		**01/07/20**	
10	**Write a C program to split a given linked list into two.**		**01/07/20**	
	ASSIGNMENT:-4	**255**		
1	**WAP to read line from input file & print alternate character**		**01/07/20**	

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. –II

R O L L NO : 30
N A M E : Ajinkya Rathod
S U B J E C T :Advanced Programming

	in the output file. Display message for file i/o errors.			
2	**Write a program to copy the contents of one file to another**		**01/07/20**	
	and also print the no. of lines in the first file.			
3	**Write a program to search a particular word in an existing**		**01/07/20**	
	file and display the no. of occurrences and the position of			
	first occurrence of that word. If the word is not found			
	display the appropriate message.			
4	**The files DATA1 and DATA2 contain sorted list of integers.**		**01/07/20**	
	Write a program to produce a third file DATA which holds a			
	single sorted merged list of these two lists.			
5	**WAP to read line by line from a file and print all repeated**		**01/07/20**	
	characters on the screen along with their frequency.			
6	**WAF to read a file and count the no. of characters, spaces,**		**01/07/20**	
	tabs, newlines and no. of words in text file.			
7	**Write a program to remove all the blank lines from a file.**		**01/07/20**	
8	**Write a function to accept a string from the keyboard and**		**01/07/20**	
	remove all occurrences of that string from a given file.			
9	**WAP program to remove all the comments from a C file**		**01/07/20**	
10	**Write a program that will generate a data file containing the**		**01/07/20**	
	list of customers and their corresponding telephone			
	numbers. Use a structure variable to store the name and			
	telephone number of each customer. Create a data file.			
11	**WAP menu driven program that will access the data file**		**01/07/20**	
	from above program and perform the functionality			
12	**Use a structure of Employee to write records of employee to**		**01/07/20**	
	a file. Include a menu that will allow the user to select any of			

R O L L NO : 30
N A M E : Ajinkya Rathod
S U B J E C T :Advanced Programming

	the following features			
13	**Write a program that will generate a data file containing the**		**01/07/20**	
	list of countries and their corresponding capitals. Place the			
	name of each country and its corresponding capital in a			
	separate structure. Treat each structure as a separate record			
14	**Write an interactive, menu-driven C program that will**		**01/07/20**	
	access the data file generated in the preceding problem and			
	then allow one of the following operations to be executed: a.			
	Determine the capital of a specified country. b. Determine			
	the country whose capital is specified. c.			
15	**Write a complete C program that can be used as a simple line-oriented text editor.**			
16	**Write a C Program to build utilities for performing following tasks (Use Command Line Arguments)** **a. For computing the average of given numbers** **b. For computing factorial of given numbers** **c. List all the files in current directory containing word ROLLWALA.** **d. Rename given file.** **e. List all EXE files in a given diectory.** **f. Merge two files into third file.**			

```
/*
**************************************************
**************************************************
**************************************************
**************************************************
**************************************************

******************** ASSIGNMENT: 1 *********************
******************** ASSIGNMENT: 1 *********************
******************** ASSIGNMENT: 1 *********************

**************************************************
**************************************************
**************************************************
**************************************************
**************************************************
*/


 /*
==========================================
==============
 * File:      p1.c
 * Copyright: 08-Feb-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:   Define a structure called Criket as:
 *                  a. Player Name
 *                  b. Team Name
 *                  c. Batting Average
 *
```

```
 *      Declare player with 50 elements. Read data and display
 *      team wise name of players with their batting avg.
 *
================================================================ */

// Header Files
#include <stdio.h>
#include <conio.h>
#include <string.h>

// Structure Declaration
struct cricket {
        char    teamName[50];
        char    playerName[50];
        float   average;
};

// Function inputData returing Strcuture of Cricket
struct cricket inputData()
{
        struct cricket c;
        static int count = 1;

        printf("\n Team %d \n", count++);

        printf(" Enter Team Name: ");
        scanf("%s",c.teamName);

        printf(" Enter Player Name: ");
```

```
	scanf("%s",c.playerName);

	printf(" Enter Batting Avg: ");
	scanf("%f",&c.average);

	return c; //Returing Entire Structure
}

// Function quicksortByTeamname
void quicksortByTeamname(struct cricket c[], int l, int r)
{
	struct cricket temp;
	char   pivot[50];
	int    i, cnt;

	if (l >= r) { return; }

	strcpy(pivot, c[r].teamName);
	cnt = l;

	for (i = l; i <= r; i++) {
		if (strcmp(c[i].teamName, pivot) <= 0) {

			// Swapping Strucutre
			temp = c[i];
			c[i] = c[cnt];
			c[cnt] = temp;
			// Swapping Strucutre Ends
```

```
                cnt++;
            }
        }

        quicksortByTeamname(c, l, cnt - 2);
        quicksortByTeamname(c, cnt, r);
}

// Function sortTeamThenAverage for Descending Sort according to Average
void sortTeamThenAverage(struct cricket c[], int num)
{
        struct cricket temp;
        int i, j, count = 1, index = 1, while_loop = 1;
        char tempTeam[50];

        while(while_loop) {
            strcpy(tempTeam, c[index].teamName);

            for(i = index; i < num; i++) {

                if(strcmp(tempTeam, c[i].teamName) == 0) {
                    count++;
                }
                else{ break; }

            }

            // Performing Bubble Sort for Average of Players
            for(i = index - 1; i < count - 1; i++) {
```

```
                for(j = index - 1; j < count - 1; j++) {

                        if(c[j].average < c[j + 1].average) {

                                // Swapping Strucutre
                                temp = c[j];
                                c[j] = c[j + 1];
                                c[j + 1] = temp;
                                // Swapping Strucutre Ends

                        }

                }
        }
        // Bubble Sort Over

        count++;
        index = count;

        // Stop the loop if count exceeds number of teams
        if(num <= count) {
                while_loop = 0;
        }

    }
}

// Function to display final result
int displaySortedData(struct cricket c[], int num)
```

```
{
	int i = 0, j;
	struct cricket temp	;

	// Function to sort by Team Name
	quicksortByTeamname(c, 0, num - 1);

	// Function to sort by Average
	sortTeamThenAverage(c, num);

	// Diplaying Data for 1st team
	printf(" * * * Team %s * * * \n", c[i].teamName);
	printf("  \tPlayer  :  %s\n",	c[i].playerName);
	printf("  \tAverage :  %.2f\n\n",  c[i].average);

	// Diplaying Data for rest of team
	for(i = 1; i < num; i++) {

		if(strcmp(c[i].teamName, c[i - 1].teamName) != 0) {
			printf(" * * * Team %s * * * \n", c[i].teamName);
		}

		printf("  \tPlayer  :  %s\n",	c[i].playerName);
		printf("  \tAverage :  %.2f\n\n", c[i].average);
	}

	return 0;
}
```

```
// Driver Program
int main()
{
	int	n, i;
	struct cricket c[50];

	while(1) {
		printf("Enter total number of teams: ");
		scanf("%d",&n);
		if(n > 0 && n < 51) { break; }
		else {
			printf("Invalid Input. Try Again\n");
		}
	}

	// Getting Data from user
	for(i = 0; i < n; ++i) {
		c[i] = inputData();
	}

	// Displaying Data
	displaySortedData(c, n);

	return 0;
}
```

/*
==========
==========

* Output :

Enter total number of teams: 10

Team 1

Enter Team Name: csk

Enter Player Name: Dhaval

Enter Batting Avg: 55

Team 2

Enter Team Name: dd

Enter Player Name: Kohli

Enter Batting Avg: 89

Team 3

Enter Team Name: mi

Enter Player Name: Ajinkya

Enter Batting Avg: 100

Team 4

Enter Team Name: csk

Enter Player Name: askdka

Enter Batting Avg: 45

Team 5

Enter Team Name: csk

Enter Player Name: askhdgsa

Enter Batting Avg: 47

Team 6

Enter Team Name: mi

Enter Player Name: askjdhskaj

Enter Batting Avg: 78

Team 7

Enter Team Name: mi

Enter Player Name: Yash

Enter Batting Avg: 85

Team 8

Enter Team Name: dd

Enter Player Name: Nirav

Enter Batting Avg: 95

Team 9

Enter Team Name: dd

Enter Player Name: Pradip

Enter Batting Avg: 67

Team 10

Enter Team Name: csk

Enter Player Name: Ghanshyam

Enter Batting Avg: 70

* * * Team csk * * *

Player : Ghanshyam

Average : 70.00

Player : Dhaval
Average : 55.00

Player : askhdgsa
Average : 47.00

Player : askdka
Average : 45.00

* * * Team dd * * *
Player : Nirav
Average : 95.00

Player : Kohli
Average : 89.00

Player : Pradip
Average : 67.00

* * * Team mi * * *
Player : Ajinkya
Average : 100.00

Player : Yash
Average : 85.00

Player : askjdhskaj

```
            Average : 78.00
================================================================================*/



/*
================================================================================
 * File: p2.c
 * Copyright: 09-Feb-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:    Define a structure as:
 *                  a. Team Name
 *                  b. City Name
 *                  c. No. of Wins
 *
 *     For Baseball Team, add
 *           i.      No. of Hits
 *           ii.     no. of runs
 *           iii.    no. of errors
 *           iv.     no. of extra timing games
 *
 *     For Baseball Team, add
 *           i.       No. of Ties
 *           ii.     no. of Goals
 *           iii.    no. of Touch Downs
 *           iv.     no. of Turnovers
 *
 *     Display Data for Baseball and Football seperately
```

```
 *     Also display data sorted by highest no. of wins
 *
================================================
================*/
#include <stdio.h>
#include <string.h>

// Struct Declaration for Union "game" Starts
// Struct Declaration for Baseball
struct baseball{
        int hits;
        int runs;
        int errors;
        int extra_timing_games;
};

// Struct Declaration for Football
struct football {
        int ties;
        int goals;
        int touchdowns;
        int turn_overs;
};
// Struct Declaration for Union "game" Ends

// Main Structure Declaration Starts
struct team {
        char teamName[50];
        char city[50];
        int  wins;
```

```
        int  option;

        // Union Declaration Starts
        union game {
                struct baseball;
                struct football;
        };
        // Union Declaration Ends

};
// Main Structure Declaration Ends

// Function "inputData" for taking details of all teams
struct team inputData()
{
        struct team t;
        union  game g;

        static int count = 1;

        printf("\n Team %d \n", count++);

        printf(" Enter Team Name: ");
        scanf("%s",t.teamName);

        printf(" Enter City Name: ");
        scanf("%s",t.city);

        printf(" Enter Wins: ");
```

```
scanf("%d",&t.wins);

// Selecting Game. Baseball or Football
while(1) {
	printf("Which game this team plays?. Choose option \n");

	printf("\t1. Baseball  \n");
	printf("\t2. Football  \n");

	scanf("%d",&t.option);

	if(t.option == 1 || t.option == 2) {
		break;
	} else {
		printf("\nInvalid Input. Try Again\n");
	}

}
// Selecting Game Ends

// Details required for Baseball
if(t.option == 1) {
	printf(" Enter Hits: ");
	scanf("%d",&t.hits);

	printf(" Enter Runs: ");
	scanf("%d",&t.runs);

	printf(" Enter Errors: ");
```

```
            scanf("%d",&t.errors);

            printf(" Enter Extra Timing Games: ");
            scanf("%d",&t.extra_timing_games);
        }
        // Details for Baseball Ends

        // Details required for Football
         else {
            printf(" Enter Ties: ");
            scanf("%d",&t.ties);

            printf(" Enter goals: ");
            scanf("%d",&t.goals);

            printf(" Enter Touchdowns: ");
            scanf("%d",&t.touchdowns);

            printf(" Enter Turn Overs: ");
            scanf("%d",&t.turn_overs);
        }
        // Details for Football Ends

        return t;
}


// Function "displayBaseball" to display Baseball Teams only
int displayBaseball(struct team t)
```

```
{
	static int count = 1;

	// option 1 is for baseball, so checking "1"
	if (t.option == 1) {
		printf("\n\n ===Team %d=== \n", count++);

		printf(" Team Name: %s\n", t.teamName);
		printf(" City:      %s\n", t.city);
		printf(" Wins:      %d\n", t.wins);

		printf("\n");

		printf(" Hits:        %d \n", t.hits);
		printf(" Runs:        %d \n", t.runs);
		printf(" Errors:      %d \n", t.errors);
		printf(" Extra Games: %d \n",t.extra_timing_games);

	}
	return 0;
}

// Function "displayFootball" to display Football Teams only
int displayFootball(struct team t)
{
	static int count = 1;

	// option 2 is for football, so checking "2"
	if (t.option == 2) {
```

```
            printf("\n\n ===Team %d=== \n", count++);

            printf(" Team Name: %s\n", t.teamName);
            printf(" City:      %s\n", t.city);
            printf(" Wins:      %d\n", t.wins);

            printf("\n");

            printf(" Ties:        %d \n", t.ties);
            printf(" Goals:       %d \n", t.goals);
            printf(" Touchdowns:  %d \n", t.touchdowns);
            printf(" Turnovers:   %d \n", t.turn_overs);

        }
        return 0;
}

// Function "quicksortByWins" for sorting Teams by
// highest no. of wins
void quicksortByWins(struct team c[], int l, int r)
{
        struct team temp;
        int pivot;
        int i, cnt;

        if (l >= r) { return; }

        pivot =  c[r].wins;
        cnt = l;
```

```
	for (i = l; i <= r; i++) {
		if (c[i].wins >= pivot) {

			// Swapping Structure Starts
			temp = c[i];
			c[i] = c[cnt];
			c[cnt] = temp;
			// Swapping Structure Ends

			cnt++;
		}
	}

	quicksortByWins(c, l, cnt - 2);
	quicksortByWins(c, cnt, r);
}


// Driver Program
int main()
{
	int n, i;
	struct team t[50];

	// Taking Number of teams
	while(1) {
		printf("Enter total number of teams: ");
		scanf("%d",&n);
```

```
        if(n > 0 && n < 51) {
            break;
        } else {
            printf("Invalid Input. Try Again\n");
        }
    }

    // Getting Data from user
    for (i = 0; i < n; ++i) {
        t[i] = inputData();
    }

    // Sorting Teams according to Highest Wins First
    quicksortByWins(t, 0, n - 1);

    // Displaying data for Baseball Team
    printf("\n\n\n * * * Baseball Team * * *");
    for (i = 0; i < n; ++i) {
        displayBaseball(t[i]);
    }

    // Displaying data for Football Team
    printf("\n\n\n * * * Football Team * * *");
    for (i = 0; i < n; ++i) {
        displayFootball(t[i]);
    }

    return 0;
}
```

```
/*
================================================================
 * Output :
        Enter total number of teams: 10

         Team 1
         Enter Team Name: Tigers
         Enter City Name: Banglore
         Enter Wins: 10
        Which game this team plays?. Choose option
              1. Baseball
              2. Football
        1
         Enter Hits: 12
         Enter Runs: 120
         Enter Errors: 2
         Enter Extra Timing Games: 2

         Team 2
         Enter Team Name: Lions
         Enter City Name: Hydrabad
         Enter Wins: 12
        Which game this team plays?. Choose option
              1. Baseball
              2. Football
        2
         Enter Ties: 1
         Enter goals: 22
```

Enter Touchdowns: 2

Enter Turn Overs: 8

Team 3

Enter Team Name: Warriors

Enter City Name: Ahmedabad

Enter Wins: 55

Which game this team plays?. Choose option

1. Baseball
2. Football

2

Enter Ties: 4

Enter goals: 102

Enter Touchdowns: 33

Enter Turn Overs: 42

Team 4

Enter Team Name: Winners

Enter City Name: Delhi

Enter Wins: 12

Which game this team plays?. Choose option

1. Baseball
2. Football

1

Enter Hits: 10

Enter Runs: 20

Enter Errors: 30

Enter Extra Timing Games: 40

Team 5

Enter Team Name: Savers

Enter City Name: Mumbai

Enter Wins: 33

Which game this team plays?. Choose option

1. Baseball
2. Football

1

Enter Hits: 23

Enter Runs: 12

Enter Errors: 2

Enter Extra Timing Games: 1

Team 6

Enter Team Name: Wellingtons

Enter City Name: Kolkatta

Enter Wins: 88

Which game this team plays?. Choose option

1. Baseball
2. Football

1

Enter Hits: 33

Enter Runs: 22

Enter Errors: 0

Enter Extra Timing Games: 7

Team 7

Enter Team Name: Heroes

Enter City Name: Pune

Enter Wins: 45

Which game this team plays?. Choose option

1. Baseball
2. Football

2

Enter Ties: 12

Enter goals: 44

Enter Touchdowns: 33

Enter Turn Overs: 22

Team 8

Enter Team Name: Unity

Enter City Name: Kochi

Enter Wins: 77

Which game this team plays?. Choose option

1. Baseball
2. Football

1

Enter Hits: 66

Enter Runs: 55

Enter Errors: 44

Enter Extra Timing Games: 33

Team 9

Enter Team Name: PenPencil

Enter City Name: Gandhidham

Enter Wins: 108

Which game this team plays?. Choose option

1. Baseball

2. Football

2

Enter Ties: 0

Enter goals: 1008

Enter Touchdowns: 0

Enter Turn Overs: 1

Team 10

Enter Team Name: Best

Enter City Name: Raipur

Enter Wins: 2

Which game this team plays?. Choose option

1. Baseball

2. Football

1

Enter Hits: 12

Enter Runs: 13

Enter Errors: 55

Enter Extra Timing Games: 33

* * * Baseball Team * * *

===Team 1===

Team Name: Wellingtons

City: Kolkatta

Wins: 88

Hits: 33

Runs: 22

Errors: 0

Extra Games: 7

===Team 2===

Team Name: Unity

City: Kochi

Wins: 77

Hits: 66

Runs: 55

Errors: 44

Extra Games: 33

===Team 3===

Team Name: Savers

City: Mumbai

Wins: 33

Hits: 23

Runs: 12

Errors: 2

Extra Games: 1

===Team 4===

Team Name: Winners

City: Delhi

Wins: 12

Hits: 10

Runs: 20

Errors: 30

Extra Games: 40

===Team 5===

Team Name: Tigers

City: Banglore

Wins: 10

Hits: 12

Runs: 120

Errors: 2

Extra Games: 2

===Team 6===

Team Name: Best

City: Raipur

Wins: 2

Hits: 12

Runs: 13

Errors: 55

Extra Games: 33

* * * Football Team * * *

===Team 1===

Team Name: PenPencil

City: Gandhidham

Wins: 108

Ties: 0

Goals: 1008

Touchdowns: 0

Turnovers: 1

===Team 2===

Team Name: Warriors

City: Ahmedabad

Wins: 55

Ties: 4

Goals: 102

Touchdowns: 33

Turnovers: 42

===Team 3===

Team Name: Heroes

City: Pune

Wins: 45

Ties: 12

Goals: 44

Touchdowns: 33

Turnovers: 22

===Team 4===

Team Name: Lions

City: Hydrabad

Wins: 12

Ties: 1

Goals: 22

Touchdowns: 2

Turnovers: 8

==========
========== */

/*
==========
==========

* File: p2.c

DEFINE :

3. Write a program that stores and displays the records of the customer.

The following information for account of the customer is to be stored.

Account no, account type, name, old balance, new balance, last payment, date of last payment.

Take structure for storing the date

in days, months and year.

Also display the current account status by comparing
current payment and previous balance.

Also calculate the current balance by subtracting
the current payment from the previous balance.

```
 *
================================================
================*/



#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>

typedef struct {
	int dd,mm,yy;
} date;

typedef struct {
	int acc_no;
	float old_balance,new_balance,last_payment;
	char name[15];
	char status[10];
	date dateofpay;
} customer;

customer getData();
```

```
void setPrint(customer[],int);
void menudriven(customer[],int);


customer getData()
{
	customer s;

	printf("\nEnter Account Number : ");
	scanf("%d",&s.acc_no);

	printf("Enter The Customer's Name : ");
	scanf("%s",s.name);

	printf("Enter The Old Balance : ");
	scanf("%f",&s.old_balance);

	printf("Enter The Last Payment : ");
	scanf("%f",&s.last_payment);

	s.new_balance = s.old_balance - s.last_payment;

	if(s.new_balance > 0)
		strcpy(s.status,"Pending");
	else
		strcpy(s.status,"Clear");

	printf("Enter Date of Last Payment [dd mm yy] : ");
	scanf("%d %d
%d",&s.dateofpay.dd,&s.dateofpay.mm,&s.dateofpay.yy);
```

```
	printf(" ========== ");
	return s;
}

void setPrint(customer s[],int tc)
{
	int i;
	printf("\n\tList Of Customers\n ========== ");
	for( i = 0; i < tc; i++ )
	{
		printf("\n\n\tId: %d \n\tName: %s\n\tOld Bal: %.2f\n\tLast
Payment: %.2f\n\tLast Pymnt Dt: %d/%d/%d\n\tNew Bal. = %.2f\n
==========
",s[i].acc_no,s[i].name,s[i].old_balance,s[i].last_payment,s[i].dateofpay.
dd,s[i].dateofpay.mm,s[i].dateofpay.yy,s[i].new_balance);
	}
}

void menudriven(customer s[],int tc)
{
	int i,j,option,check;
	char name[15];

	printf("\n\n1: Disp Customer Details\n2: Find Customer By
Name\n3: Disp Status\n4: Display Curr Bal.\n5: Exit\n\n");
	scanf("%d",&option);

	if(option == 1) {
		setPrint(s,tc);
		menudriven(s,tc);
	}
```

```
	else if(option == 2) {
		printf("Enter Customer Name : ");
		scanf("%s",name);
		for( i = 0; i < tc; i++ ) {
			check = strcmp(s[i].name,name);
			if( check == 0 )
				printf("\n\n\tCustomer Id = %d \n\tCustomer
Name = %s\n\tOld Balance = %.2f\n\tLast Payment = %.2f\n\tLast
Payment Date = %d/%d/%d\n\tNew Balance = %.2f \n\tStatus = %s
\n\n ==========
",s[i].acc_no,s[i].name,s[i].old_balance,s[i].last_payment,s[i].dateofpay.
dd,s[i].dateofpay.mm,s[i].dateofpay.yy,s[i].new_balance,s[i].status);
		}
		menudriven(s,tc);
	}

	else if(option == 3) {
		for( i = 0; i < tc; i++ )
			printf("\nCustomer Name = %s\n Status = %s\n
========== ",s[i].name,s[i].status);
		menudriven(s,tc);
	}

	else if(option == 4) {
		for( i = 0; i < tc; i++ )
			printf("\nCustomer Name = %s\n Current Balance =
%.2f\n ========== ",s[i].name,s[i].new_balance);
		menudriven(s,tc);
	}

	else if(option == 5) {
```

```
            exit(0);
        }

        else {
            printf("Invalid Option");
            menudriven(s,tc);
        }

        printf("\nNo such Customer\n");

        menudriven(s,tc);
}


int main()
{
        customer c[50];
        int i,total_cust;
        printf("Enter Number of Customers: ");
        scanf("%d",&total_cust);
        for( i = 0; i < total_cust; i++ )
        {
            c[i] = getData();
        }

        setPrint(c,total_cust);
        menudriven(c,total_cust);

        return 0;
```

```
}



/*
================================================
===============
 * Output :


        Enter Number of Customers: 3


        Enter Account Number : 10
        Enter The Customer's Name : Ajinkya
        Enter The Old Balance : 5000
        Enter The Last Payment : 5000
        Enter Date of Last Payment [dd mm yy] : 12 12 12
         ==========
        Enter Account Number : 20
        Enter The Customer's Name : Pradip
        Enter The Old Balance : 10000
        Enter The Last Payment : 1200
        Enter Date of Last Payment [dd mm yy] : 5 8 13
         ==========
        Enter Account Number : 30
        Enter The Customer's Name : Nirav
        Enter The Old Balance : 5600
        Enter The Last Payment : 5700
        Enter Date of Last Payment [dd mm yy] : 26 1 14
         ==========
```

List Of Customers

==========

Id: 10

Name: Ajinkya

Old Bal: 5000.00

Last Payment: 5000.00

Last Pymnt Dt: 12/12/12

New Bal. = 0.00

==========

Id: 20

Name: Pradip

Old Bal: 10000.00

Last Payment: 1200.00

Last Pymnt Dt: 5/8/13

New Bal. = 8800.00

==========

Id: 30

Name: Nirav

Old Bal: 5600.00

Last Payment: 5700.00

Last Pymnt Dt: 26/1/14

New Bal. = -100.00

==========

1: Disp Customer Details

2: Find Customer By Name

3: Disp Status

4: Display Curr Bal.

5: Exit

1

List Of Customers

==========

Id: 10

Name: Ajinkya

Old Bal: 5000.00

Last Payment: 5000.00

Last Pymnt Dt: 12/12/12

New Bal. = 0.00

==========

Id: 20

Name: Pradip

Old Bal: 10000.00

Last Payment: 1200.00

Last Pymnt Dt: 5/8/13

New Bal. = 8800.00

==========

Id: 30

Name: Nirav

Old Bal: 5600.00

Last Payment: 5700.00

Last Pymnt Dt: 26/1/14

New Bal. = -100.00

==========

1: Disp Customer Details

2: Find Customer By Name

3: Disp Status

4: Display Curr Bal.

5: Exit

2

Enter Customer Name : Nirav

Customer Id = 30

Customer Name = Nirav

Old Balance = 5600.00

Last Payment = 5700.00

Last Payment Date = 26/1/14

New Balance = -100.00

Status = Clear

==========

1: Disp Customer Details

2: Find Customer By Name

3: Disp Status

4: Display Curr Bal.

5: Exit

3

Customer Name = Ajinkya
 Status = Clear
 ==========
Customer Name = Pradip
 Status = Pending
 ==========
Customer Name = Nirav
 Status = Clear
 ==========

1: Disp Customer Details
2: Find Customer By Name
3: Disp Status
4: Display Curr Bal.
5: Exit

4

Customer Name = Ajinkya
 Current Balance = 0.00
 ==========
Customer Name = Pradip
 Current Balance = 8800.00
 ==========
Customer Name = Nirav
 Current Balance = -100.00

==========

1: Disp Customer Details

2: Find Customer By Name

3: Disp Status

4: Display Curr Bal.

5: Exit

5

==========
========== */

/*
==========
==========

* File: p4.c

* Copyright: 09-Feb-2020 by Ajinkya Rathod(ajinzrathod)

*

* Content: Define a structure as:

* a. Author

* b. Title

* c. Price

* d. Publisher

* e. Stock Position

*

```
 *
==========================================
==========*/



#include <stdio.h>
#include <string.h>

// Main Structure Declaration Starts
struct books {

	char author[50];
	char title[50];
	float price;
	char publisher[50];
	int stock;

};
// Main Structure Declaration Ends



// Function "inputData" for taking details of all Books
void inputData(struct books b[], int n)
{
	int i;

	for (i = 0; i < n; ++i) {

		printf("\n ========== Details of Books : %d
========== \n", i + 1);
```

```
            printf("\nEnter Name of Author\n");
            scanf("%s",b[i].author);

            printf("\nEnter Title of Book\n");
            scanf("%s",b[i].title);

            printf("\nEnter price of Book\n");
            scanf("%f",&b[i].price);

            printf("\nEnter name of Publisher\n");
            scanf("%s",b[i].publisher);

            printf("\nEnter Stock\n");
            scanf("%d",&b[i].stock);

        }
}

void searchBook(struct books b[], int n)
{

        int i, j, copies, authorFound = 0, titleFound = 0;
        char tempAuthor[50], tempTitle[50];
        float amt;

        printf("\n ========== Search any Book ========== \n");

        printf("\nEnter Author Name : ");
```

```
        scanf("%s",tempAuthor);

        for(i = 0; i < n; ++i) {

                // Checking if author exists
                if(strcmp(b[i].author, tempAuthor) == 0) {

                        authorFound = 1;

                        printf("\nEnter Book Title : ");
                        scanf("%s",tempTitle);

                        // Checking if such Book Title exists
                        if(strcmp(b[i].title, tempTitle) == 0) {

                                titleFound = 1;

                                printf("\nEnter Number of Copies You Want : ");
                                scanf("%d",&copies);

                                if(b[i].stock >= copies)  {

                                        amt = b[i].price * copies; // Bill amount

                                        printf("\n\tTotal Payable Amount For %d
Copies Will Be :\n", copies);
                                        printf("\t[%.2f x %d] = $ %.2f\n",
b[i].price, copies, amt);

                                }
```

```
                        // when available copies are less than demanded
                        else {
                                printf("Out of Stock\n");
                        }

                }
        }
    }

    // if no such author exists
    if(!authorFound) {
        printf("< %s > author not found in list\n", tempAuthor);
    }

    // if so such book title exists
    else if(!titleFound) {
        printf("< %s > has no such book as < %s >\n", tempAuthor,
tempTitle);
    }
}



// Driver Program
int main()
{
    int n, i;
    struct books b[50];

    // Taking Number of Books
```

```
	while(1) {

		printf("Enter no of books: ");
		scanf("%d",&n);

		if(n > 0 && n < 51) {
			break;
		}

		else {
			printf("Invalid Input. Try Again\n");
		}

	}

	// Getting data for books
	inputData(b, n);

	// Search any book you want
	searchBook(b, n);

	return 0;
}



 /*
================================================
===============
 * Output :
```

Enter no of books: 4

========== Details of Books : 1 ==========

Enter Name of Author

Ajinkya

Enter Title of Book

Python

Enter price of Book

2601

Enter name of Publisher

Ghanshyam

Enter Stock

812

========== Details of Books : 2 ==========

Enter Name of Author

Lakshya

Enter Title of Book

Kierra

Enter price of Book

521.125

Enter name of Publisher

Advani

Enter Stock

125

========== Details of Books : 3 ==========

Enter Name of Author

Balaguruswamy

Enter Title of Book

C

Enter price of Book

256.85

Enter name of Publisher

Hills

Enter Stock

71

========== Details of Books : 4 ==========

Enter Name of Author

Devangi

Enter Title of Book

Riverfront

Enter price of Book

999.99

Enter name of Publisher

Pabari

Enter Stock

99

========== Search any Book ==========

Enter Author Name : Balaguruswamy

Enter Book Title : C

Enter Number of Copies You Want : 25

Total Payable Amount For 25 Copies Will Be :

[256.85 x 25] = $ 6421.25

==========
==========*/

```
/*
================================================
===============
 * File: p5.c
 * Copyright: 09-Feb-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:    Define a structure as:
 *                  a. Vehicle Name
 *                  b. Vehicle Price
 *                  c. Vehicle Type [2 / 4]
 *
 *     For 2 Wheeler, add
 *           i.      Mileage
 *           ii.     Type [Geared / Gearless]
 *
 *     For 4 Wheeler, add
 *           i.       Usage [Auto / Manual]
 *           ii.     Engine No.
 *           iii.    Type [Heavy / Light]
 *
 *  Display Data for 2 and 4 wheelers seperately
 *
================================================
===============*/



#include <stdio.h>
#include <string.h>
#include <stdbool.h>
```

```
// Struct Declaration for Union "count_wheels" Starts
// Struct Declaration for two_wheeler
struct two_wheeler {

        float  mileage;
        bool    gear; // true for gear, false for gearless

};


// Struct Declaration for four_wheeler
struct four_wheeler {

        bool usage; // true for auto, false for manual
        char engine_no[20];
        bool weight; // true for heavy, false for light

};
// Struct Declaration for Union "count_wheels" Ends


// Main Structure Declaration Starts
struct vehicle {

        char  vehicleName[50];
        float price;
        bool  wheels; // true for 2 wheeler, false for 4 wheeler
```

```
        // Union Declaration Starts
        union count_wheels {
                struct two_wheeler;
                struct four_wheeler;
        };
        // Union Declaration Ends


};
// Main Structure Declaration Ends


// Function "inputData" for taking details of all vehicles
struct vehicle inputData()
{
        struct vehicle v;
        int    wheels, gearType, usageType, vehicleWeight;

        static int i = 0;

        printf("\nDetails for Vehicle No: %02d\n", ++i);

        printf(" Enter Vehicle Name: ");
        scanf("%s",v.vehicleName);

        printf(" Enter Vehicle Price: ");
        scanf("%f",&v.price);

        while(1) {
```

```
printf(" Enter No. of Wheels: ");
scanf("%d",&wheels);
// printf("size = %d\n", sizeof(wheels)); // prints 4

if(wheels == 2) {
	v.wheels = true;
	// printf("size = %d\n", sizeof(v.wheels)); // prints 1

	printf(" Enter Mileage: ");
	scanf("%f",&v.mileage);

	printf(" Enter Gear Type \n");
	printf("  Press 1 for Geared. 0 for Gearless: ");
	scanf("%d",&gearType);

	if(gearType) {
		v.gear = true; // Vehicle is Geared
	}

	else {
		v.gear = false; // Vehicle is Gearless
	}
	break;

}
else if(wheels == 4) {
	v.wheels = false;
	// printf("size = %d\n", sizeof(v.wheels)); // prints 1
```

```
        printf(" Enter Usage \n");
        printf("  Press 1 for Auto. 0 for Manual: ");
        scanf("%d",&usageType);

        if(usageType) {
                v.usage = true; // Usage is Auto
        }

        else {
                v.usage = false; // Usage is Manual
        }

        printf(" Enter Engine No: ");
        scanf("%s",v.engine_no);

        printf(" Enter Weight \n");
        printf("  Press 1 for Heavy. 0 for Light: ");
        scanf("%d",&vehicleWeight);

        if(vehicleWeight) {
                v.weight = true; // Weight is Heavy
        }

        else {
                v.weight = false; // Weight is Light
        }

        break;
}
```

```
		else {
			printf("Invalid no of Wheels. Try Again \n");
		}
	}

	return v;
}


// Function "displayTwoWheelers" to display 2 Wheelers only
int displayTwoWheelers(struct vehicle v)
{

	// Prints only when v.wheels is true
	// Thus, prints only 2 wheelers
	if(v.wheels) {
		printf("\n");

		printf(" Name:    %s\n", v.vehicleName);
		printf(" Price:  %0.2f\n", v.price);

		printf("\n");
		printf(" Mileage:    %0.2f\n",v.mileage);

		if(v.gear) { //if v.gear is true
			printf(" Gear Type: Geared\n");
		}
```

```
            else { // if v.gear is false
                    printf(" Gear Type: Gearless\n");
            }

            printf("\n====================\n");
    }

    return 0;
}


// Function "displayFourWheelers" to display 4 Wheelers only
int displayFourWheelers(struct vehicle v)
{

    // Prints only when v.wheels is false
    // Thus, prints only 4 wheelers
    if(!v.wheels) {
            printf("\n");

            printf(" Name:   %s\n", v.vehicleName);
            printf(" Price:  %0.2f\n", v.price);

            printf("\n");

            if(v.usage) { // if v.usage is true
                    printf(" Usage:     Auto\n");
            }
```

```
            else { // if v.usage is false
                printf(" Usage:     Manual\n");
            }

            printf(" Engine No. %s\n", v.engine_no);

            if(v.weight) { // if v.weight is true
                printf(" Weight:    Heavy\n");
            }

            else { // if v.weight is false
                printf(" Weight:    Light\n");
            }

            printf("\n====================\n");
        }
        return 0;
}


// Driver Program
int main()
{
        int n, i;
        struct vehicle v[50];


        // Taking Number of teams
        while(1) {
```

```
        printf("Enter total number of Vehicles: ");
        scanf("%d",&n);

        if(n > 0 && n < 51) {
                break;
        }

        else {
                printf("Invalid Input. Try Again\n");
        }

}


// Getting Data from user
for (i = 0; i < n; ++i) {
        v[i] = inputData();
}


// Displaying data for 2 Wheelers
printf("\n\n * * * * * Two Wheelers * * * *  *\n");
for (i = 0; i < n; ++i) {
        displayTwoWheelers(v[i]);
}


// Displaying data for 4 Wheelers
```

```
	printf("\n\n * * * * * Four Wheelers * * * * *\n");
	for (i = 0; i < n; ++i) {
		displayFourWheelers(v[i]);
	}

	return 0;
}

 /*
================================================
===============
 * Output :
	 Enter total number of Vehicles: 10

	Details for Vehicle No: 01
	 Enter Vehicle Name: BMW
	 Enter Vehicle Price: 96387.789
	 Enter No. of Wheels: 4
	 Enter Usage
	  Press 1 for Auto. 0 for Manual: 1
	 Enter Engine No: EAF9TY
	 Enter Weight
	  Press 1 for Heavy. 0 for Light: 1

	Details for Vehicle No: 02
	 Enter Vehicle Name: HarleyDavidson
	 Enter Vehicle Price: 15354.1565
	 Enter No. of Wheels: 2
	 Enter Mileage: 8.7
	 Enter Gear Type
```

Press 1 for Geared. 0 for Gearless: 1

Details for Vehicle No: 03
Enter Vehicle Name: Mercedes
Enter Vehicle Price: 20012
Enter No. of Wheels: 4
Enter Usage
Press 1 for Auto. 0 for Manual: 0
Enter Engine No: ABC1DE
Enter Weight
Press 1 for Heavy. 0 for Light: 0

Details for Vehicle No: 04
Enter Vehicle Name: Swift
Enter Vehicle Price: 1560.589
Enter No. of Wheels: 4
Enter Usage
Press 1 for Auto. 0 for Manual: 0
Enter Engine No: EGJ89Y
Enter Weight
Press 1 for Heavy. 0 for Light: 0

Details for Vehicle No: 05
Enter Vehicle Name: Splendor
Enter Vehicle Price: 1545
Enter No. of Wheels: 2
Enter Mileage: 60
Enter Gear Type
Press 1 for Geared. 0 for Gearless: 1

Details for Vehicle No: 06

Enter Vehicle Name: AstonMartin

Enter Vehicle Price: 99878.98754

Enter No. of Wheels: 4

Enter Usage

Press 1 for Auto. 0 for Manual: 1

Enter Engine No: WER997

Enter Weight

Press 1 for Heavy. 0 for Light: 0

Details for Vehicle No: 07

Enter Vehicle Name: Pulsar

Enter Vehicle Price: 2568.258

Enter No. of Wheels: 2

Enter Mileage: 25.265

Enter Gear Type

Press 1 for Geared. 0 for Gearless: 1

Details for Vehicle No: 08

Enter Vehicle Name: Ninja

Enter Vehicle Price: 78965.256

Enter No. of Wheels: 2

Enter Mileage: 4.25

Enter Gear Type

Press 1 for Geared. 0 for Gearless: 0

Details for Vehicle No: 09

Enter Vehicle Name: Passion

Enter Vehicle Price: 782.256

Enter No. of Wheels: 2

Enter Mileage: 89

Enter Gear Type

Press 1 for Geared. 0 for Gearless: 1

Details for Vehicle No: 10

Enter Vehicle Name: PorscheCarerraGT

Enter Vehicle Price: 963879.99979

Enter No. of Wheels: 4

Enter Usage

Press 1 for Auto. 0 for Manual: 1

Enter Engine No: AJ1NK

Enter Weight

Press 1 for Heavy. 0 for Light: 1

* * * * * Two Wheelers * * * * *

Name: HarleyDavidson

Price: 15354.16

Mileage: 8.70

Gear Type: Geared

==========

Name: Splendor

Price: 1545.00

Mileage: 60.00
Gear Type: Geared

==========

Name: Pulsar
Price: 2568.26

Mileage: 25.26
Gear Type: Geared

==========

Name: Ninja
Price: 78965.26

Mileage: 4.25
Gear Type: Gearless

==========

Name: Passion
Price: 782.26

Mileage: 89.00
Gear Type: Geared

==========

* * * * * Four Wheelers * * * * *

Name: BMW
Price: 96387.79

Usage: Auto
Engine No. EAF9TY
Weight: Heavy

==========

Name: Mercedes
Price: 20012.00

Usage: Manual
Engine No. ABC1DE
Weight: Light

==========

Name: Swift
Price: 1560.59

Usage: Manual
Engine No. EGJ89Y
Weight: Light

```
      ====================

       Name:   AstonMartin
       Price:  99878.98

       Usage:     Auto
       Engine No. WER997
       Weight:    Light

      ====================

       Name:   PorscheCarerraGT
       Price:  963880.00

       Usage:     Auto
       Engine No. AJ1NK
       Weight:    Heavy

      ====================


================================================
===============*/

/*
**************************************************
**************************************************
**************************************************
**************************************************
**************************************************
```

```
****************** ASSIGNMENT: 2 ********************
****************** ASSIGNMENT: 2 ********************
****************** ASSIGNMENT: 2 ********************

**********************************************************
**********************************************************
**********************************************************
**********************************************************
**********************************************************
*/

#include<stdio.h>
int getData(int myArray[], int n)
{
	int i;
	int *p = myArray;
	for (i = 0; i < n; ++i) {
		printf("Enter number %d: ", i + 1);
		scanf("%d",p);
		p++;
	}
}

int reverseData(int myArray[], int n)
{
	int i, *p = myArray + n - 1;

	printf("\n ===== Numbers in reverse Order ===== \n");
```

```
	for (i = 0; i < n; ++i) {
		printf("%d: ", *p);
		p--;
	}
}

int main()
{
	int myArray[10], n, i;
	int *p = myArray;
	printf("Enter N: ");
	scanf("%d",&n);

	getData(myArray, n);
	reverseData(myArray, n);

	printf("\n");
	return 0;
}



/*
================================================
=====================
Output:
Enter N: 5
Enter number 1: 10
Enter number 2: 20
Enter number 3: 30
```

```
Enter number 4: 40
Enter number 5: 50

 ===== Numbers in reverse Order =====
50: 40: 30: 20: 10:
================================================
====================== */

/*
================================================
=====================
Max-min from Pointer
================================================
====================== */



#include<stdio.h>

int findmax(int max, int myArray[], int n)
{
	int i;
	int *p = myArray;

	for (i = 1; i < n; ++i) {
		printf("Enter number %02d: ", i + 1);
		scanf("%d",p);

		if(*p > max) {
			max  = *p;
		}
		p++;
```

```
        }
        return max;
}

int main()
{
        int myArray[10], n, i, max;
        int *p = myArray;

        printf("Enter N: ");
        scanf("%d",&n);

        printf("Enter number 01: ");
        scanf("%d",p);

        max = *p;
        max = findmax(max, myArray, n);

        printf("%d is max element\n", max);
        return 0;
}
```

```
/*
==========================================
=====================
```

Output:

```
        Enter N: 5
        Enter number 01: 10
        Enter number 02: 20
        Enter number 03: 40
        Enter number 04: 30
        Enter number 05: 5
        40 is max element

================================================
====================== */



/*
================================================
======================
Read String Using Pointers
================================================
====================== */

#include<stdio.h>
#include<string.h>

char strupr(char *str)
{

  unsigned char *p = (unsigned char *)str;

  while (*p) {
     *p = toupper((unsigned char)*p);
      p++;
```

```
  }


}


int filterString(char string1[])
{
	char *cptr;
	int vowels = 0, consonants = 0, numbers = 0, whitespaces = 0, special_characters = 0;


	// string1 = strupr(string1);


	cptr = string1;


	while(*cptr != '\0') {


		// Vowels should always come before consonants, else consonants will always be increased
		if(*cptr == 65 || *cptr == 69 || *cptr == 73 || *cptr == 79 || *cptr == 85) {
			printf("Vowel: %c is stored at %d \n", *cptr, cptr);
			vowels++;
		}


		else if (*cptr > 47 && *cptr < 58) {
			printf("Number: %c is stored at %d \n", *cptr, cptr);
			numbers++;
		}


		else if(*cptr == 9 || *cptr == 32) {
```

```
                printf("Whitespace: %c is stored at %d \n", *cptr,
cptr);
                whitespaces++;
            }

            else if(*cptr > 64 && *cptr < 91){
                printf("Consonant: %c is stored at %d \n", *cptr, cptr);
                consonants++;
            }

            else {
                printf("Special Character: %c is stored at %d \n", *cptr,
cptr);
                special_characters++;
            }

            cptr++;
        }

        printf("\n Vowels: %d\n", vowels);
        printf("\n Numbers: %d\n", numbers);
        printf("\n Whitespaces: %d\n", whitespaces);
        printf("\n Consonants: %d\n", consonants);
        printf("\n Special Characters: %d\n", special_characters);

        return 0;
}


int main()
```

```
{
	char string1[50];

	printf("Enter String\n");
	scanf("%[^\n]",string1);

	strupr(string1);

	filterString(string1);

	printf("\n");

	return 0;
}




/*
================================================
======================
================================================
====================== */




/*
================================================
======================
Transpose using pointer
================================================
====================== */
```

```
#include <stdio.h>


void readData(int matrix[][5], int rows, int cols)
{
	int i, j;

	int (*arrptr)[5] = matrix;

	for (i = 0; i < rows; ++i) {
		printf("Enter row %d: \n", i + 1);

		for (j = 0; j < cols; ++j) {
			printf("Enter Col %d: ", j + 1);
			// scanf("%d",arrptr + (i * j) + j);
			scanf("%d",*(arrptr + i) + j);
			printf("Value: %d \n",*(*(arrptr + i) + j));
		}

	}
}

void displayData(int matrix[][5], int rows, int cols)
{
	int i, j;
	int (*arrptr)[5] = matrix;

	printf("\nMatrix Entered: \n");
```

```
        for (i = 0; i < rows; ++i) {

                for (j = 0; j < cols; ++j) {
                        printf("%d \t ",*(*(arrptr + i) + j));
                }

                printf("\n");
        }
}

void transpose(int matrix[][5], int rows, int cols)
{
        int i, j;
        int (*arrptr)[5] = matrix;

        printf("\nOutput: \n");

        for (i = 0; i < cols; ++i) {

                for (j = 0; j < rows; ++j) {
                        printf("%d \t ",*(*(arrptr + j) + i));
                }

                printf("\n");
        }
}

int main()
```

```
{
	int matrix[5][5];
	int rows, cols;

	printf("Enter Rows and cols\n");
	scanf("%d %d",&rows, &cols);

	if(rows > 5 || cols > 5) {
		printf("Invalid Input \n");
		// exit(0);
	}

	readData(matrix, rows, cols);
	displayData(matrix, rows, cols);
	transpose(matrix, rows, cols);

	printf("\n");
}



/*
==========================================
====================
	Output:
	Enter Rows and cols
	3
	3
	Enter row 1:
	Enter Col 1: 10
```

Value: 10

Enter Col 2: 20

Value: 20

Enter Col 3: 30

Value: 30

Enter row 2:

Enter Col 1: 40

Value: 40

Enter Col 2: 50

Value: 50

Enter Col 3: 60

Value: 60

Enter row 3:

Enter Col 1: 70

Value: 70

Enter Col 2: 80

Value: 80

Enter Col 3: 90

Value: 90

Matrix Entered:

10	20	30
40	50	60
70	80	90

Output:

10	40	70
20	50	80
30	60	90

```
================================================
====================== */



/*
================================================
======================
Matrix Multiplication using Pointers
================================================
====================== */



#include <stdio.h>

void readData(int matrix[][5], int rows, int cols)
{
	int i, j;

	int (*arrptr)[5] = matrix;

	for (i = 0; i < rows; ++i) {
		printf("Enter row %d: \n", i + 1);

		for (j = 0; j < cols; ++j) {
			printf("Enter Col %d: ", j + 1);
			// scanf("%d",*(arrptr + (i * j) + j));
			scanf("%d",*(arrptr + i) + j);
			printf("Value: %d \n",*(*(arrptr + i) + j));
		}

	}
```

```
}

void displayData(int matrix[][5], int rows, int cols)
{
	int i, j;
	int (*arrptr)[5] = matrix;

	printf("\nMatrix Entered: \n");

	for (i = 0; i < rows; ++i) {

		for (j = 0; j < cols; ++j) {
			printf(" %3d \t",*(*(arrptr + i) + j));
		}
		printf("\n");
	}

	printf("\n");

}

void multiply(int matrix1[][5], int matrix2[][5], int rows1, int cols1, int
cols2)
{
	int i, j, k, total, pdt;
	int matrix3[5][5];

	int (*arrptr1)[5] = matrix1;
	int (*arrptr2)[5] = matrix2;
	int (*arrptr3)[5] = matrix3;
```

```
        printf("\nMultiplication : \n");

        for (i = 0; i < rows1; ++i) {

                for (j = 0; j < cols2; ++j) {

                        total = 0;

                        for (k = 0; k < cols1; ++k) {
                                total += *(*(arrptr1 + i) + k) * *(*(arrptr2 + k)
+ j);
                        }

                        *(*(arrptr3 + i) + j) = total;

                }

        }


        for (i = 0; i < rows1; i++) {

                for (j = 0; j < cols2; j++) {

                        printf ("%3d\t", *(*(arrptr3 + i) + j));
                }

                printf("\n");
```

```
        }

}

int main()
{

        int matrix1[5][5];
        int matrix2[5][5];

        int rows1, cols1, rows2, cols2;

        printf("Enter Rows and cols of matrix1\n");
        scanf("%d %d",&rows1, &cols1);

        printf("Enter Rows and cols of matrix1\n");
        scanf("%d %d",&rows2, &cols2);

        if(rows2 != cols1) { printf("Invalid Input \n"); }

        readData(matrix1, rows1, cols1);
        displayData(matrix1, rows1, cols1);

        readData(matrix2, rows2, cols2);
        displayData(matrix2, rows2, cols2);

        multiply(matrix1, matrix2, rows1, cols1, cols2);
```

```
        printf("\n");

        return 0;

}
```

```
/*
================================================
======================
```

Output:

```
        Enter Rows and cols of matrix1
        2
        3
        Enter Rows and cols of matrix1
        3
        2
        Enter row 1:
        Enter Col 1: 10
        Value: 10
        Enter Col 2: 20
        Value: 20
        Enter Col 3: 30
        Value: 30
        Enter row 2:
        Enter Col 1: 40
        Value: 40
        Enter Col 2: 50
        Value: 50
```

```
Enter Col 3: 60
Value: 60

Matrix Entered:
  10      20      30
  40      50      60

Enter row 1:
Enter Col 1: 70
Value: 70
Enter Col 2: 80
Value: 80
Enter row 2:
Enter Col 1: 90
Value: 90
Enter Col 2: 100
Value: 100
Enter row 3:
Enter Col 1: 110
Value: 110
Enter Col 2: 120
Value: 120

Matrix Entered:
  70      80
  90     100
 110     120
```

```
	Multiplication :
	5800    6400
	13900   15400
========================================================
====================== */



/*
========================================================
======================
Sum of Matrix Using Pointers
========================================================
====================== */



#include <stdio.h>

void readData(int matrix[][5], int rows, int cols)
{
	int i, j;

	int (*arrptr)[5] = matrix;

	for (i = 0; i < rows; ++i) {
		printf("\nEnter row %d: \n", i + 1);

		for (j = 0; j < cols; ++j) {
			printf("Enter Col %d: ", j + 1);
			scanf("%d",*(arrptr + i) + j);
		}
```

```
	}
}

void displayData(int matrix[][5], int rows, int cols)
{
	int i, j, rowTotal, colTotal;
	int (*arrptr)[5] = matrix;

	printf("\nMatrix Entered: \n");

	for (i = 0; i < rows; ++i) {

		rowTotal = 0;
		for (j = 0; j < cols; ++j) {
			rowTotal += *(*(arrptr + i) + j);
			printf(" %3d \t",*(*(arrptr + i) + j));
		}
		printf("| %d\n", rowTotal);
	}

	for (i = 0; i <= cols; ++i) {
		printf(" --- \t");
	}

	printf("\n");

	for (i = 0; i < cols; ++i) {

		colTotal = 0;
```

```
		for (j = 0; j < rows; ++j) {
			colTotal += *(*(arrptr + j) + i);
		}

		printf(" %3d \t", colTotal);
	}

	printf("\n");

}

void sum(int matrix[][5], int rows, int cols)
{
	int i, j, total = 0;

	int (*arrptr)[5] = matrix;

	for (i = 0; i < rows; ++i) {

		for (j = 0; j < cols; ++j) {
			total = total + *(*(arrptr + i) + j);
		}

	}

	printf("\nAddition : %d\n", total);

}
```

```
int main()
{

	int matrix[5][5];

	int rows, cols;

	printf("Enter Rows and cols of matrix1\n");
	scanf("%d %d",&rows, &cols);

	readData(matrix, rows, cols);
	displayData(matrix, rows, cols);

	sum(matrix, rows, cols);

	printf("\n");

	return 0;

}

/*
==========================================
====================
Output:
	Enter Rows and cols of matrix1
	3
	3

	Enter row 1:
```

```
Enter Col 1: 10
Enter Col 2: 20
Enter Col 3: 30

Enter row 2:
Enter Col 1: 40
Enter Col 2: 50
Enter Col 3: 60

Enter row 3:
Enter Col 1: 80
Enter Col 2: 70
Enter Col 3: 90

Matrix Entered:
 10    20    30   | 60
 40    50    60   | 150
 80    70    90   | 240
---   ---   ---   ---
130   140   180

Addition : 450
========================================================== */


//
==========================================================**
// 8. Write function that receives a sorted array of integers and an integer value,
```

```
// and inserts the value in correct place.
//
==========================================
==========================================*
*


#include<stdio.h>
#include<conio.h>

void readnum(int[],int,int*);
void addnum(int[],int*,int*);
void display(int[],int);


void main()
{
	int num[10],limit,n=0;
	printf("Enter how many numbers are in your array: ");
	scanf("%d",&limit);
	readnum(num,limit,&n);
	addnum(num,&limit,&n);
	display(num,limit);
	getch();
}


void readnum(int num[],int limit,int *new_num)
	//passing the number which user want ot add in a pointer so
{                                                       //it can be
used in other function without returning from here..
	int i,*iptr;
	iptr=num;
	printf("Enter %d sorted numbers: ",limit);
```

```
	for(i=0;i<limit;i++)
	{
		scanf("%d",iptr);
		iptr++;
	}
		printf("Enter number to add in your sorted array: ");
		scanf("%d",new_num);

}

void addnum(int num[],int *new_limit,int *new_num)
	//Again passing address of limit and new number..
{
	int *iptr,i,j,flag=0;
	iptr=num;
	for(i=0;i<*new_limit;i++)
	{
		if(*new_num < *(iptr+i))                    //finding the greater number than our number so we can add new number in the sorted array
		{
			for(j=*(new_limit)-1;j>=i;j--)
			{
				*(iptr+j+1)=*(iptr+j);
			}
			*(iptr+i)=*new_num;
			(*new_limit)++;
	//increasing limit by 1
			flag=1;
			break;
```

```
            }
        }
        if(!flag)                                   //if flag is off
that means new number is gretest among sorted array..
        {
            *(iptr+i)=*new_num;
            *new_limit=*new_limit+1;
        }
}

void display(int num[],int limit)
{
        int i,*iptr;

        iptr=num;
        printf("New Array: \n");
        for(i=0;i<limit;i++)
        {
            printf("%d ",*iptr);
            iptr++;
        }
}

/*
============================================
============================================*
*
// output:

Enter how many numbers are in your array: 5
Enter 5 sorted numbers: 10 20 30 50 60
```

Enter number to add in your sorted array: 40

New Array:

10 20 30 40 50 60

```
//
==========================================
==========================================*
/
```

```
/*
==========================================
==========================================*
*
```

9. Write a function that will round a floating point number to an indicated decimal place

eg: The number 17.457 would yield the

value 17.46 when it is rounded off to two decimal places

```
==========================================
==========================================*
*/
```

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

void round(char[],int);

void main()
```

```
{
	char num[10];
	int digits;
	printf("Enter number: ");
	gets(num);
	printf("Enter number of decimal places you want: ");
	scanf("%d",&digits);
	round(num,digits);
	printf("Rounded off number: ");
	puts(num);
	getch();
}


void round(char num[],int digits)
{
	int decimal=0,i=0;
	char *cptr;
	cptr=num;
	while(*cptr != '.')
	{
		i++;
		cptr++;
	}
	if(*(cptr+digits+1)>52 )
	{
		(*(cptr+digits))++;
	}
	*(cptr+digits+1)='\0';
```

```
}



/*
================================================
================================================*
*


output:

Enter number: 23.786

Enter number of decimal places you want: 2

Rounded off number: 23.79


================================================
================================================*
*/







/*
================================================
================================================*
*
10. Write a function using pointers to exchange the value stored in two
locations in the memory.
================================================
================================================*
*/



#include<stdio.h>
```

```
#include<conio.h>
void input_data10(int *a,int *b)
{
	int *ptra,*ptrb;
	ptra=a;
	ptrb=b;
	printf("Enter Number 1 : ");
	scanf("%d",ptra);
	printf("Enter Number 2 : ");
	scanf("%d",ptrb);
}
void disp_data10(int a,int b)
{
	printf("\nA = %d",a);
	printf("\nB = %d\n",b);
}
void swap(int *a,int *b)
{
	int *ptra,*ptrb;
	int temp;
	ptra=a;
	ptrb=b;
	temp=*ptra;
	*ptra=*ptrb;
	*ptrb=temp;

}

int main()
```

```
{
	int a=0,b=0;

	input_data10(&a,&b);

	printf("\tBEFORE SWAP");

	disp_data10(a,b);

	swap(&a,&b);

	printf("\tAFTER SWAP");

	disp_data10(a,b);

	return 0;
}

 /*
==============================================
==============================================*
*
output:

Enter Number 1 : 10
Enter Number 2 : 20

	BEFORE SWAP

A = 10
```

B = 20

AFTER SWAP

A = 20

B = 10

==========
==========*
*/

/*
==========
==========*
*

11_a. Find the first occurrence of a character in the given string. The function should return the

position in the string.

==========
==========*
*/

```
#include<stdio.h>
#include<conio.h>
void disp_str(char *ptr)
{
	//char *ptr;
	//ptr=str;
	printf("String = ");
	while(*ptr != '\0')
	{
```

```
            printf("%c",*ptr);
            ptr++;
        }
}

int search_char(char *ptr, char ch)
{
        int i,val = -1;
        //char *ptr;
        //ptr = str;
        for(i = 0 ; *ptr != '\0' ; i++)
        {
            if(*ptr == ch)
            {
                val = i;
                break;
            }
            ptr++;
        }
        return val;
}


int main()
{
        char str[50],ch,*ptr;
        int search_val;

        printf("Enter String :");
```

```
	gets(str);

	ptr = str;

	disp_str(ptr);

	printf("\nWhich Character You Find :");
	scanf("%c",&ch);

	search_val = search_char(ptr,ch);

	if (search_val == -1)
		printf("\nCharacter %c is not found..",ch);

	else
		printf("Character %c is found at %d",ch,search_val);

	return 0;
}

/*
==============================================
==============================================*
*

Enter String :Ajinkya
String = Ajinkya
Which Character You Find :k
Character k is found at 4
```

```
==========================================
==========================================*
*/
```

```
/*==========================================
==========================================
=**
```

11_b. Find the first occurrence of a string in another string. The function should return the position in the string.

```
==========================================
==========================================*
*/

#include<stdio.h>
#include<conio.h>
#include<string.h>

int search(char str1[],char str2[])
{
	int i,j,found=1,index,itemp;
	char *sptr1,*sptr2;
	sptr1=str1;
	sptr2=str2;

	for(i=0; *(sptr1+i) != '\0'; i++)
	{
		if(*(sptr1+i) == *(sptr2))
		{
```

```
                index=i;
                itemp=i;
                itemp++;
                for(j=1;*(sptr2+j) != '\0';j++,itemp++)
                {
                        if(*(sptr2+j) != *(sptr1+itemp))
                        {
                                found=0;
                                break;
                        }
                }
                if(found)
                {
                        return index;
                }
                found=1;
        }
    }

    return -1;
}

void main()
{
    char str1[10],str2[10];
    int index;
    printf("Enter first string: ");
    gets(str1);
    printf("Enter second string: ");
```

```
	gets(str2);

	index=search(str1,str2);
	if(index == -1)
	{
		printf("Second string not found in first string!!");
	}
	else
	{
		printf("Second string found at index %d in first string",index);
	}
	getch();
}



/*==========================================
==========================================
=**
Enter first string: Ajinkya
Enter second string: ink
Second string found at index 2 in first string
==========================================
==========================================*
*/



/*
==========================================
==========================================*
*
```

11_c. Delete all occurrences of a character from a string.

```
================================================
================================================*
*/

#include<stdio.h>
#include<conio.h>
void input_data(char str[])
{
	printf("Enter String =");
	gets(str);
}
void display11_c(char *ptr)
{
	printf("\nString = ");
	while(*ptr != '\0')
	{
		printf("%c",*ptr);
		ptr++;
	}
	printf("\n");
}
void delete_char(char *ptr,char ch,char str2[])
{
	char *ptr2;
	ptr2 = str2;
	while(*ptr != '\0')
	{
		//printf("aer");
```

```
            if(*ptr != ch)
            {
                  *ptr2 = *ptr;
                  ptr2++;
            }
            ptr++;
      }
      *ptr2 = '\0';
}

int main()
{
      char str[50],str2[50],ch;
      char *ptr;

      input_data(str);

      ptr = str;

      display11_c(ptr);

      printf("\nEnter Which Character You want to delete :");
      scanf("%c",&ch);

      delete_char(ptr,ch,str2);

      printf("\nAfter Deleting Character:");
      display11_c(str2);
```

```
	return 0;
}

/*==========================================
==========================================
=**

Enter String =ajinkya

String = ajinkya

Enter Which Character You want to delete :a

After Deleting Character:
String = jinky

==========================================
==========================================*
*/

/*
==========================================
==========================================*
*
11_d. Delete all occurrences of a string from another string.

==========================================
==========================================*
*/
#include<stdio.h>
#include<string.h>
```

```
void remove_s(char str1[],char str2[])
{
	int i,j,k,found=1,index,itemp,len1,len2;
	char *sptr1,*sptr2;
	len1=strlen(str1);
	len2=strlen(str2);
	sptr1=str1;
	sptr2=str2;

	for(i=0; i<len1; i++)

	{
		if(*(sptr1+i) == *(sptr2))
		{
			index=i;
			itemp=i;
			itemp++;
			for(j=1;j<len2;j++,itemp++)
			{
				if(*(sptr2+j) != *(sptr1+itemp))
				{
					found=0;
					break;
				}
			}
			if(found)
			{
				for(k=0;k<len1;k++)
				{
```

```
                        *(sptr1+index+k)=*(sptr1+index+len2+k);

                    }
                    len1-=len2;
                    *(str1+len1)='\0';
                    i--;
                }
                found=1;
            }
        }
}

int main()
{
    char str1[50],str2[40];
    int index;

    printf("Enter first string: ");
    gets(str1);

    printf("Enter second string: ");
    gets(str2);

    remove_s(str1,str2);

    printf("After removing second string: \n");

    puts(str1);
```

```
	return 0;

 }


/*
================================================
================================================*
*
	Enter first string: Ajinkya
	Enter second string: ky
	After removing second string:
	Ajina
================================================
================================================*
*/




/*
================================================
================================================*
*
11_e. Delete all occurrences of a character from a string. Ignore Case.

================================================
================================================*
*/

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

```
void delete_char(char str[],char ch)
{
	char *sptr=str;
	int i,j,length,index;
	length=strlen(str);

	for(i=0;i<length;i++)
	{
		if(*(sptr+i) == ch ||*(sptr+i) == ch-32 || *(sptr+i) ==
ch+32  )
		{
			j=i;
			while(j<length-1)
			{
				*(sptr+j)=*(sptr+j+1);
				j++;
			}
			*(sptr+length-1)='\0';
			length--;
			i--;
		}
	}
}

int main()
{
	char str[10],ch;

	printf("Enter string: ");
```

```
    gets(str);

    printf("Enter character to delete its all occurrences: ");
    scanf("%c",&ch);

    delete_char(str,ch);

    printf("Result: ");
    puts(str);

    return 0;

}



/*
================================================
================================================*
*
Enter string: aajjiinnkkyyaa
Enter character to delete its all occurrences: i
Result: aajjnnkkyyaa
================================================
================================================*
*/




/*================================================
================================================
=**
```

11_f. Delete all occurrences of a string from another string. Ignore Case.

```
==========================================
==========================================*
*/



#include<stdio.h>
#include<string.h>

void remove_str(char str1[],char str2[])
{
	int i,j,k,found=1,index,itemp,len1,len2;
	char *sptr1,*sptr2;
	len1=strlen(str1);
	len2=strlen(str2);
	sptr1=str1;
	sptr2=str2;

	for(i=0; i<len1; i++)
	{
		if(*(sptr1+i) == *(sptr2)|| *(sptr1+i) == *(sptr2)-32
||*(sptr1+i) == *(sptr2)+32)
		{
			index=i;

			itemp=i;
			itemp++;
			for(j=1;j<len2;j++,itemp++)
			{
```

```
                        if(*(sptr2+j) != *(sptr1+itemp) && *(sptr2+j) !=
*(sptr1+itemp)-32 && *(sptr2+j) != *(sptr1+itemp)+32)

                        {
                                found=0;
                                break;
                        }
                }
                if(found)
                {
                        for(k=0;k<len1;k++)

                        {
                                *(sptr1+index+k)=*(sptr1+index+len2+k);

                        }
                        len1-=len2;

                        *(str1+len1)='\0';
                        i--;
                }
                found=1;
            }
        }
}



int main()
{
        char str1[50],str2[50];
        int index;
```

```
	printf("Enter first string: ");
	gets(str1);

	printf("Enter second string: ");
	gets(str2);

	remove_str(str1,str2);

	printf("After removing second string: \n");
	puts(str1);


	return 0;
}


/*
================================================
===============

	Output:
	Enter first string: Ajinkya
	Enter second string: AJi
	After removing second string:
	nkya


================================================
=============== */
```

```
/*
================================================
================================================*
*

11_g. Copy one string to another string.

================================================
================================================*
*/

#include<stdio.h>

void get_input(char str1[])
{
	printf("Enter String =");
	gets(str1);
}



void display(char *ptr)
{
	while(*ptr != '\0')
	{
		printf("%c",*ptr);
		ptr++;
	}
	printf("\n");
}
```

```
void copy(char str2[],char *ptr)
{
	char *ptr2;
	ptr2 = str2;
	while(*ptr != '\0'){
		*ptr2 = *ptr;
		printf("\n%c: %c",(*ptr),(*ptr2));
		ptr2++;
		ptr++;
	}
	*ptr2 = '\0';
}

int main()
{
	char str1[50],str2[50];
	char *ptr;

	get_input(str1);

	ptr = str1;

	display(ptr);

	printf("Copied String: ");
	copy(str2,ptr);

	printf("\n\nString :");
	display(str2);
```

```
	return 0;
}

/*
================================================
================================================*
*
output:
	Enter String =Ajinkya
	Ajinkya
	Copied String:
	A: A
	j: j
	i: i
	n: n
	k: k
	y: y
	a: a

	String :Ajinkya
================================================
================================================*
*/



/*
================================================
================================================
==

11_h.Copy n characters of one string to another string.
```

```
==========================================
==========================================*
*/

#include<stdio.h>

void input11_h(char str[])
{
	printf("Enter String :");
	gets(str);
}



void disp11_h(char *ptr)
{
	printf("String = ");
	while(*ptr != '\0')
	{
		printf("%c",*ptr);
		ptr++;
	}
	printf("\n");
}

void copych (char *ptr,char *cptr,int n)
{
	int i = 0;
	for(i = 0; i<n ; i++)
	{
```

```
            //printf("asd");
            *cptr = *ptr;
            cptr++;
            ptr++;
        }
        *cptr = '\0';
}

int main()
{
        char str[50],cstr[50];
        char *ptr,*cptr;
        int n;//i = 0;

        input11_h(str);

        ptr = str;

        disp11_h(ptr);

        printf("Enter How Many Characters You want to Copy: ");
        scanf("%d",&n);

        cptr = cstr;
        printf("Cloned String: ");

        copych(ptr,cptr,n);
        disp11_h(cptr);
```

```
	return 0;
}

/*
================================================
================================================*
*



Enter String :aaajjiinnkkyyaa
String = aaajjiinnkkyyaa
Enter How Many Characters You want to Copy: 7
Cloned String: String = aaajjii



================================================
================================================*
*/



/*
================================================
================================================*
*

11_i. Find length of the string and toggle the characters of the string.

================================================
================================================*
*/

#include<stdio.h>
```

```
void input11_i(char str[])
{
	printf("Enter String :");
	gets(str);
}
void disp(char *ptr)
{
	printf("\nString = ");
	while(*ptr != '\0')
	{
		printf("%c",*ptr);
		ptr++;
	}
	printf("\n");
}
void toggle(char *ptr)
{
	while(*ptr != '\0')
	{
		if(*ptr >= 65 && *ptr <= 90)
		{
			*ptr = *ptr + 32;
		}
		else if(*ptr >= 97 && *ptr <= 122)
		{
			*ptr = *ptr - 32;
		}
		else
		{
```

```
            }
            ptr++;
        }
}

int main()
{
        char str[50];
        char *ptr;

        input11_i(str);

        ptr = str;

        disp(ptr);

        toggle(ptr);

        disp(ptr);

        return 0;
}

/*
================================================
================================================*
*
        Enter String :ajink
```

```
        String = ajink

        String = AJINK
================================================
================================================*
*/



/*================================================
================================================
=**

11_j.Convert string to all upper case.

================================================
================================================*
*/

#include<stdio.h>

void input(char str[])
{
        printf("Enter String :");
        gets(str);
}

void display11_j(char *ptr)
{
        printf("String = ");
        while(*ptr != '\0')
        {
```

```
            printf("%c",*ptr);
            ptr++;
        }
        printf("\n");
}

void toupper(char *ptr)
{
        while(*ptr != '\0')
        {
            if(*ptr >= 97 && *ptr <=122)
            {
                *ptr = *ptr - 32;
                //*ptr++;
            }
            ptr++;
            //printf("%c",*ptr2);
        }
}

int main()
{
        char str[100];
        char *ptr;

        input(str);

        ptr = str;
```

```
        display11_j(ptr);

        printf("\nToUpper: ");

        toupper(ptr);

        display11_j(ptr);

        return 0;
}

/*==========================================
==========================================
=**

Enter String :Ajinkya
String = Ajinkya

ToUpper: String = AJINKYA

==========================================
==========================================*
*//*======================================
======================================
==**

11_k.Convert string to all Lower case.

==========================================
==========================================*
*/
```

```
#include<stdio.h>
#include<conio.h>
void inputk(char str[])
{
	printf("Enter String: ");
	gets(str);
}

void display11_k(char *ptr)
{
	printf("String = ");
	while(*ptr != '\0')
	{
		printf("%c",*ptr);
		ptr++;
	}
	printf("\n");
}

void tolower(char *ptr)
{
	while(*ptr != '\0')
	{
		if(*ptr >= 65 && *ptr <=90)
			*ptr = *ptr + 32;
		ptr++;
	}
}
```

```
int main()
{
	char str[100];
	char *ptr;

	inputk(str);

	ptr = str;

	display11_k(ptr);

	printf("\nAfter Converting ToLower Case\n ");

	tolower(ptr);

	display11_k(ptr);

	return 0;
}



/*==========================================
==========================================
=**
output:
Enter String :Ajinkya
String = Ajinkya

After Converting ToLower Case:
```

ajinkya

```
==========================================
==========================================*
*/



/*==========================================
==========================================
=**
```

11_L. Sort an array of string.

```
==========================================
==========================================*
*/



#include<stdio.h>
#include<string.h>

void readnames(char names[][20],int limit)
{
	char (*cptr)[20];
	int i;
	cptr=names;
	fflush(stdin);
	printf("Enter %d names: \n",limit);
	for(i=0;i<limit;i++)
	{
		gets(*(cptr+i));
	}
```

```
}

void sort(char names[][20],int limit)
{
	char (*cptr)[20],min[20];
	int i,index,j;

	cptr=names;

	for(i=0;i<limit-1;i++)
	{
		strcpy(min,*(cptr+i));
		index=i;

		for(j=i+1;j<limit;j++)
		{
			if(strcmp(min,*(cptr+j))>0)
			{
				strcpy(min,*(cptr+j));
				index=j;
			}

		}
		strcpy(*(cptr+index),*(cptr+i));
		strcpy(*(cptr+i),min);
	}

}
void print(char names[][20],int limit)
```

```
{
        char (*cptr)[20];
        int i;
        cptr=names;
        printf("Sorted names:\n");
        for(i=0;i<limit;i++)
        {
                puts(*(cptr+i));
        }
}

int main()
{
        char names[5][20];
        int limit;

        printf("How many names you have: ");
        scanf("%d",&limit);

        readnames(names,limit);

        sort(names,limit);

        print(names,limit);

        return 0;
}
```

```
/*
==========================================
==========================================*
*

Enter 3 names:
ajinkya
thindo
jinki
Sorted names:
ajinkya
jinki
thindo

==========================================
==========================================*
*/




/*
==========================================
==========================================*
*
11_m.       (m)Append one string to another string.

==========================================
==========================================*
*/
```

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

void append(char str1[],char str2[])
{
	int len1,len2,i;
	char *sptr1,*sptr2;

	sptr1=str1;
	sptr2=str2;
	len1=strlen(sptr1);
	len2=strlen(sptr2);
	*(sptr1+len1)=' ';
	for(i=0;i<len2;i++)
	{
		*(sptr1+len1+i+1)=*(sptr2+i);
	}

	*(sptr1+len1+i+1)='\0';

}

int main()
{
	char str1[30],str2[30];

	printf("Enter first string: ");
	gets(str1);
```

```
        printf("Enter second string: ");
        gets(str2);

        append(str1,str2);

        printf("String after appending: ");
        puts(str1);



        return 0;
}




/*
================================================
================================================*
*

output:
Enter first string: Rollwala
Enter second string: Computer Center
String after appending: Rollwala Computer Center



================================================
================================================*
*/
```

```
/*
==========================================
==========================================*
*
11_n. Append at most n characters of one string S2 to another string S1.


==========================================
==========================================*
*/



#include<stdio.h>
#include<string.h>

void append_n(char str1[],char str2[],int n)
{
	int len1,len2,i;
	char *sptr1,*sptr2;

	sptr1=str1;
	sptr2=str2;
	len1=strlen(sptr1);
	len2=strlen(sptr2);
	*(sptr1+len1)=' ';
	for(i=0;i<n;i++)
	{
		*(sptr1+len1+i+1)=*(sptr2+i);
	}

	*(sptr1+len1+i+1)='\0';
```

```
}

int main()
{
	int n;
	char str1[30],str2[30];

	printf("Enter first string: ");
	gets(str1);

	printf("Enter second string: ");
	gets(str2);

	printf("Enetr how many characters you want to append: ");
	scanf("%d",&n);

	append_n(str1,str2,n);

	printf("After appending %d characters: ",n);
	puts(str1);

	return 0;
}


/*
==========================================
==========================================*
*
```

output:

Enter first string: ajinkya

Enter second string: rathod

Enetr how many characters you want to append: 5

After appending 5 characters: ajinkya ratho

==========
==========*
*/

/*
==========
==========
==

11_o. Reverse all the characters in the string.

==========
==========*
*/

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

void reverse_string(char *str)
{
	int len , i ;
```

```
	char *start, *end, ch;

	len=strlen(str);
	//printf("count = %d",len);

	start = str;
	end = str;
	for(i = 0 ; i < len-1 ; i++)
	{
		end++;
	}
	//printf("end = %s",*end);
	for(i=0 ; i < len/2 ; i++)
	{
		ch = *start;
		*start = *end;
		*end = ch;
		start++;
		end--;
	}
	//printf("Reverse String = %s",*str);
}

int main()
{
	char str[100];

	printf("Enter String :");
```

```
	gets(str);

	printf("Entered: %s",str);

	reverse_string(str);

	printf("\nReversed: %s\n",str);

	return 0;
}



/*
==============================================
==============================================*
*



Enter String :ajinkya
Entered: ajinkya
Reversed: ayknija


==============================================
==============================================*
*//*
==============================================
==============================================*
*
```

11_p. Compare two strings S1 and S2. The function should return -1, 0 or 1 if S1 < S2, S1 = S2 and S1 > S2 respectively.

```
==========================================
==========================================*
*/



#include<stdio.h>
#include<string.h>

int str_compare(char s1[],char s2[])
{
	char *sp1,*sp2;
	int i=0;

	sp1=s1;
	sp2=s2;

	while(*(sp1+i) != '\0' && *(sp2+i) != '\0')
	{
		if(*(sp1+i) < *(sp2+i))
		{
			return -1;
		}
		else if(*(sp1+i) > *(sp2+i))
		{
			return 1;
		}
		i++;
	}

	if(*(sp1+i) != '\0' && *(sp2+i) == '\0')
```

```
	{
		return 1;
	}
	else if(*(sp1+i) == '\0' && *(sp2+i) != '\0')
	{
		return -1;
	}
	else
		return 0;

}

int main()
{
	char s1[30],s2[30];
	int result;

	printf("Enter string1: ");
	gets(s1);

	printf("Enter string 2: ");
	gets(s2);

	result=str_compare(s1,s2);

	printf("result is %d",result);

	return 0;
}
```

/*==========
==========
=**

output1:

Enter string1: Ajin

Enter string 2: ajin

result is -1

==========
==========*
*

output2:

Enter string1: aj

Enter string 2: aj

result is 0

==========
==========*
*/

/*
==========
==========*
*

11_q.(q) Compare two strings S1 and S2. The function should return -1, 0 or 1 if S1 < S2, S1 = S2 and S1 > S2 respectively.

Ignore case.

```
==========================================
==========================================*
*/

#include<stdio.h>
#include<string.h>

int str_icompare(char s1[],char s2[])
{
	char *sp1,*sp2;
	int i=0;

	sp1=s1;
	sp2=s2;

	while(*(sp1+i) != '\0' && *(sp2+i) != '\0')
	{
		if(*(sp1+i) < *(sp2+i))
		{
			if( *(sp1+i) < *(sp2+i)-32)
			return -1;
		}
		else if(*(sp1+i) > *(sp2+i))
		{
			if( *(sp1+i) < *(sp2+i)+32)
			return 1;
		}
		i++;
	}
```

```
        if(*(sp1+i) != '\0' && *(sp2+i) == '\0')
        {
                return 1;
        }
        else if(*(sp1+i) == '\0' && *(sp2+i) != '\0')
        {
                return -1;
        }
        else
                return 0;

}

int main()
{
        char s1[30],s2[30];
        int result;

        printf("Enter string1: ");
        gets(s1);

        printf("Enter string 2: ");
        gets(s2);

        result=str_icompare(s1,s2);

        printf("result is %d",result);

        return 0;
```

```
}
```

```
/*==========================================
==========================================
=**
```

output1:

Enter string1: AJIN

Enter string 2: ajin

result is 0

```
==========================================
==========================================*
*/
```

```
/*==========================================
==========================================
=**
```

11_r. Compare at most n characters of two strings S1 and S2. The function should return -1, 0 or 1 if S1 < S2, S1 = S2 and

S1 > S2 respectively.

```
==========================================
==========================================*
**/
```

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

```
int str_ncompare(char s1[],char s2[],int n)
{
	char *sp1,*sp2;
	int i=0;

	sp1=s1;
	sp2=s2;

	while((*(sp1+i) != '\0' && *(sp2+i) != '\0') && i<n)
	{
		if(*(sp1+i) < *(sp2+i))
		{
			return -1;
		}
		else if(*(sp1+i) > *(sp2+i))
		{
			return 1;
		}
		i++;
	}

	if(*(sp1+i) != '\0' && *(sp2+i) == '\0' && i<n)
	{
		return 1;
	}
	else if(*(sp1+i) == '\0' && *(sp2+i) != '\0' && i<n)
	{
		return -1;
```

```
	}
	else
		return 0;

}

int main()
{
	char s1[30],s2[30];
	int result,n;

	printf("Enter string1: ");
	gets(s1);

	printf("Enter string 2: ");
	gets(s2);

	printf("Enter how many characters you want to compare: ");
	scanf("%d",&n);

	result=str_ncompare(s1,s2,n);
	printf("result is %d",result);

	return 0;
}


/*==========================================
==========================================
=**
```

output:

Enter string1: rollwala

Enter string 2: rollwala computer

Enter how many characters you want to compare: 8

result is 0

```
==========================================
==========================================*
*/



/*
==========================================
==========================================*
*
```

11_s. Compare at most n characters of two strings S1 and S2. The function should return -1, 0 or 1 if S1 < S2, S1 = S2 and

S1 > S2 respectively.Ignore case.

```
==========================================
==========================================*
*/



#include<stdio.h>
#include<string.h>

int str_incompare(char s1[],char s2[],int n)
{
    char *sp1,*sp2;
    int i=0;
```

```
sp1=s1;
sp2=s2;

while((*(sp1+i) != '\0' && *(sp2+i) != '\0') && i<n)
{
	if(*(sp1+i) < *(sp2+i))
	{
		if( *(sp1+i) < *(sp2+i)-32)
		return -1;
	}
	else if(*(sp1+i) > *(sp2+i))
	{
		if( *(sp1+i) < *(sp2+i)+32)
		return 1;
	}
	i++;
}

if(*(sp1+i) != '\0' && *(sp2+i) == '\0' && i<n)
{
	return 1;
}
else if(*(sp1+i) == '\0' && *(sp2+i) != '\0' && i<n)
{
	return -1;
}
else
	return 0;
```

```
}

void main()
{
	char s1[30],s2[30];
	int result,n;

	printf("Enter string1: ");
	gets(s1);
	printf("Enter string 2: ");
	gets(s2);
	printf("Enter how many characters you want to compare: ");
	scanf("%d",&n);
	result=str_incompare(s1,s2,n);
	printf("result is %d",result);

	getch();
}



/*
==========================================
==========================================*
*
output:
Enter string1: Ajinkya
Enter string 2: ajinkya12345
Enter how many characters you want to compare: 6
result is 0
```

```
==========================================
==========================================*
*/



/*
**********************************************************
**********************************************************
**********************************************************
**********************************************************
**********************************************************

******************** ASSIGNMENT: 3 *********************
******************** ASSIGNMENT: 3 *********************
******************** ASSIGNMENT: 3 *********************

**********************************************************
**********************************************************
**********************************************************
**********************************************************
**********************************************************
*/



 /*
==========================================
==============

 * Roll No: 30
```

```
 *
 * File:      1-fifo.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to create a singly linked list
 *                  and display its elements in FIFO pattern.
 *                  Also display the number of elements in the list
 *
 *
================================================================ */


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


// Structure -> Node
struct Node
{
	int data;
	struct Node *next;
};


// Function for -> Getting N
int getN()
{
	int n;
```

```
	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	return n;
}


// Function for -> Getting data
int getData()
{
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}




// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;
	struct Node *new_node;
	struct Node *temp;
```

```
	if(pHead == NULL) {
		pHead = createNode();

		pHead -> data = getData();
		pHead -> next = NULL;

		if (n == 1) { return pHead; } // only 1 element
	}

	for(i = 1; i < n; i++) {

		temp = pHead;

		// Creating new node
		new_node = createNode();

		new_node -> data = getData();
		new_node -> next = NULL;

		while(temp->next != NULL) {
			temp = temp->next;
		}

		temp->next = new_node;

	}

	return pHead;
}
```

```
// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
	struct Node *curr = pHead;

	printf("\n ========== Result Starts ========== \n");

	while(curr != NULL) {
		printf("\ncurr -> data: %d\n", curr -> data);
		printf("curr -> next: %d\n", curr -> next);
		curr = curr -> next;
	}

	printf("\n ========== Result Ends========== \n");
}


// Function for -> Counting nodes
int countNodes(struct Node *p)
{
	int count = 0;

	while (p != NULL) {
		count ++;
		p = p -> next;
	}
```

```
	return count;
}



// Driver Code
int main()
{
	struct Node *pHead = NULL;
	int num = getN();


	pHead = insertData(pHead, num);


	display(pHead); //Display all Nodes


	printf("\nThere are %d elements in the list\n",
countNodes(pHead));


	return 0;
}


/*
================================================
====================
 * Output :


	Enter No. of nodes you want to enter: 6


	Enter Data: 10


	Enter Data: 50
```

Enter Data: 20

Enter Data: 30

Enter Data: 40

Enter Data: 90

========== Result Starts ==========

curr -> data: 10
curr -> next: 8197576

curr -> data: 50
curr -> next: 8200880

curr -> data: 20
curr -> next: 8200896

curr -> data: 30
curr -> next: 8200912

curr -> data: 40
curr -> next: 8200928

curr -> data: 90
curr -> next: 0

========== Result Ends==========

There are 6 elements in the list

```
================================================
==================== */



 /*
================================================
==============

 * Roll No: 30
 *
 * File:      10-split.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a C program to split a given linked list into two.
 *
 *
================================================
============= */

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



// Structure -> Node
```

```
struct Node
{
	int data;
	struct Node *next;
};


// Function for -> Getting N
int getN()
{
	int n;

	printf(" Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	return n;
}


// Function for -> Getting data
int getData()
{
	int data;

	printf("\n Enter Data: ");
	scanf("%d",&data);

	return data;
}
```

```
// Function for -> Creating Node
struct Node *createNode()
{
	struct  Node *new_node;

	if((new_node = (struct Node *) malloc(sizeof(struct Node))) == NULL) {
		perror ("Error");
		exit(1);
	}

	return new_node;
}


// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;
	struct Node *new_node;
	struct Node *temp;

	if(pHead == NULL) {
		pHead = createNode();

		pHead -> data = getData();
		pHead -> next = NULL;
```

```
        if (n == 1) { return pHead; }
    }

    for(i = 1; i < n; i++) {

        temp = pHead;

        new_node = createNode();

        new_node -> data = getData();
        new_node -> next = NULL;

        while(temp->next != NULL) {
            temp = temp->next;
        }

        temp->next = new_node;

    }

    return pHead;
}



int countNodes(struct Node *p)
{
    int count = 0;
```

```
	while (p != NULL) {
		count ++;
		p = p -> next;
	}

	return count;
}



void printStylish(int count)
{
	printf("\n ********");
	for (int i = 0; i < count; ++i) {
		printf("************");
	}
	printf("\n");
}

// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
	int i = 0, count;
	struct Node *curr = pHead;

	count = countNodes(pHead);

	if(pHead == NULL) {
		printf(" ********* Empty List ********* \n");
		return;
```

```
}

printStylish(count);

printf(" * Index *");
while(curr != NULL) {
	printf(" %8d * ", ++i);
	curr = curr -> next;
}

printStylish(count);
curr = pHead;

printf(" * Data  *");
while(curr != NULL) {
	printf(" %8d * ", curr -> data);
	curr = curr -> next;
}

printStylish(count);
curr = pHead;

printf(" * Next  *");
while(curr != NULL) {
	printf(" %8d * ", curr -> next);
	curr = curr -> next;
}

printStylish(count);
```

```
}


int askSplitType()
{
	int splitType;

	printf(" How do you want to split \n\t 1. Using Index  \n\t 2. Using Value: \n \t ");
	scanf ("%d",&splitType);

	if(splitType == 1 || splitType == 2) {
		return splitType;
	}
	else {
		printf(" *** Enter valid option *** \n");
		splitType = askSplitType();
	}

}


int askIndex(int count)
{
	int n, index;

	printf("\n From which Index do you want to create a new List: ");
	scanf("%d",&index);
```

```
	if(index > 0 && index <= count) {
		return index;
	}
	printf("Enter valid Index\n");
	n = askIndex(count);

}

struct Node *split(struct Node **pHead, int type)
{
	struct Node *curr = *pHead;
	struct Node *prev = *pHead;
	struct Node *newList;

	int i = 0, index, count, data, foundData;

	count = countNodes(*pHead);

	if(type == 1) {
		index = askIndex(count);
		while (curr != NULL) {
			if(++i == index){
				if(index == 1) {
					newList = *pHead;
					*pHead = NULL;
					return newList;
				}
				newList = prev -> next;
				prev -> next = NULL;
```

```
                return newList;
            }
            prev = curr;
            curr = curr -> next;
        }
    }

    else if(type == 2) {
        foundData = 0;
        data = getData();
        while (curr != NULL) {
            i++;
            if(data == curr -> data){
                foundData = 1;

                // Finding Duplicate Data
                curr = curr -> next;
                while (curr != NULL) {
                    if(data == curr -> data) {
                        printf(" ********* Duplicate record found. Enter Index no to split *********\n");
                        newList = split(&(*pHead), 1);

                        return newList;
                    }
                    curr = curr -> next;
                }

                if(i == 1) {
                    newList = *pHead;
```

```
                        *pHead = NULL;
                        return newList;
                    }
                    newList = prev -> next;
                    prev -> next = NULL;
                    return newList;
                }
                prev = curr;
                curr = curr -> next;
            }
            if(!foundData) {
                printf(" \n ********* Data not found ********* \n");
            }

        }
}


int main()
{
    struct Node *pHead = NULL;
    struct Node *newList;

    int type;
    int num = getN();

    pHead = insertData(pHead, num);

    display(pHead);
```

```
        type = askSplitType();

        newList = split(&pHead, type);

        printf(" \n ========== List 1 ==========\n");
        display(pHead);

        printf(" \n ========== List 2 ==========\n");
        display(newList);

        return 0;
}


/*
================================================
====================
* Output :

         Enter No. of nodes you want to enter: 5

         Enter Data: 10

         Enter Data: 20

         Enter Data: 30

         Enter Data: 20
```

Enter Data: 50

```
**********************************************************************
*********
      * Index *       1 *       2 *       3 *       4 *       5 *
**********************************************************************
*********
      * Data  *      10 *       20 *       30 *       20 *       50 *
**********************************************************************
*********
      * Next  * 12785096 *  12788400 *  12788416 *  12788432 *
0 *
**********************************************************************
*********
```

How do you want to split

1. Using Index
2. Using Value:

2

Enter Data: 20

********* Duplicate record found. Enter Index no to split *********

From which Index do you want to create a new List: 3

========== List 1 ==========

```
**********************************
* Index *       1 *       2 *
```

```
      *********************************
      * Data  *      10 *       20 *
      *********************************
      * Next  * 12785096 *        0 *
      *********************************

      ========== List 2 ==========

      ********************************************
      * Index *       1 *        2 *        3 *
      ********************************************
      * Data  *      30 *       20 *       50 *
      ********************************************
      * Next  * 12788416 *  12788432 *        0 *
      ********************************************

================================================
======================== */




 /*
================================================
==============

 * Roll No: 30
 *
 * File:      2-lifo.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
```

```
 *
 * Content: Write a program to create a singly linked list
 *                  and display its elements in LIFO pattern.
 *                  Also display the number of elements in the list
 *
 *
==========================================
============= */

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

struct Node
{
	int data;
	struct Node *next;
};

// Function for -> Asking number of Elements
int num()
{
	int n;

	printf("How much elements you wanna enter: ");
	scanf("%d",&n);

	return n;
}
```

```
// Function for -> Getting Data from user
int getElement()
{
	int data;

	printf("Enter Data: ");
	printf("     ^^^^^\n");

	scanf("%d",&data);

	return data;
}

// Function for -> Creating new node
struct Node *createNode()
{
	struct  Node *new_node;

	// Sufficient memory not available
	if((new_node = (struct Node *) malloc(sizeof(struct Node))) ==
NULL ) {
		perror ("Error");
		exit(1);
	}

	return new_node;
}
```

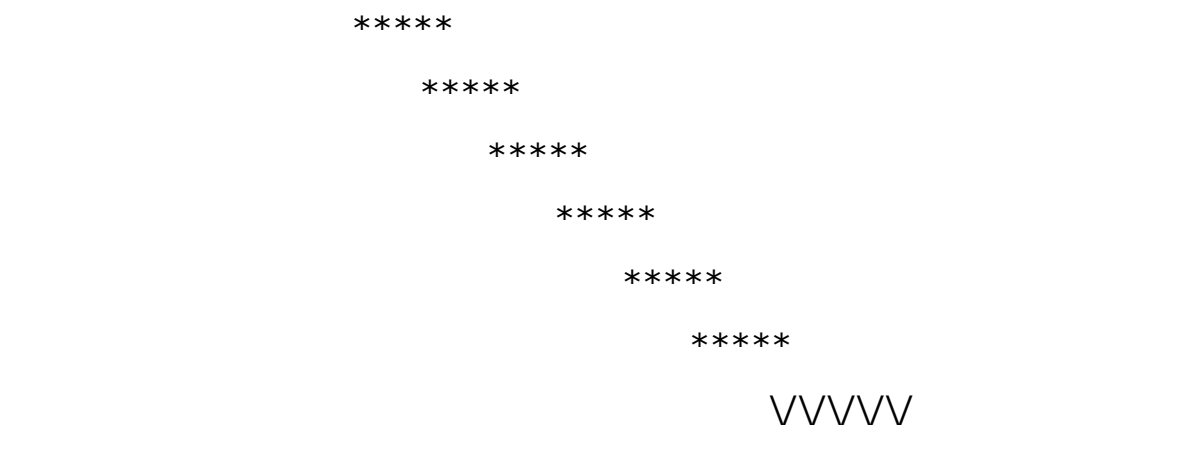
```
void printStylish(struct Node *new_node)
{
	int j, k;

	for(j = 17; j < 24; j++) {

		for (k = 0; k < j; k++) {
			printf(" ");
		}

		for (k = 0; k < j - 17; k++) {
			printf("     ");
		}

		printf("*****\n");

	}

	printf("                                                  \\/\\/\\/\\/\\/ \n");
	printf("**************************************************
**********************\n");
	printf("** new_node: %8d || ", new_node);
	printf(" data    : %8d || ", new_node->data);
	printf(" next    : %8d ** ", new_node->next);
	printf("\n**************************************************
**********************\n");
}


// Function for -> Inserting Data
```

```
struct Node *insert(struct Node *pHead, int n)
{
	int i, j, k, data;
	struct  Node *new_node;
	struct Node *temp;

	new_node = createNode();

	printf("\n              ********\n");
	printf("               * NULL *\n");
	printf("               ********\n");

	if(pHead == NULL) {

		pHead = new_node;

		new_node -> data = getElement();
		new_node -> next = NULL;

		if(n == 1) { return pHead; }
		printStylish(new_node);

	}

	for (i = 1; i < n; ++i) {

		temp = new_node;

		new_node = createNode();
```

```
            new_node -> data = getElement();
            new_node -> next = temp;

            printStylish(new_node);

        }

        return new_node;
}


// Function for -> Displaying Nodes
void display(struct Node *first)
{
        struct Node *disp = first;

        printf("\n========== Result ==========\n");

        while(disp != NULL) {

            printf("\ndisp: %d\n", disp);
            printf("disp -> data: %d\n", disp -> data);
            printf("disp -> next: %d\n", disp -> next);

            disp = disp -> next;
        }

        printf("\ndisp: %d\n", disp);
}
```

```
// Function for -> Counting nodes
int countNodes(struct Node *p)
{
	int count = 0;

	while (p != NULL) {
		count ++;
		p = p -> next;
	}

	return count;
}


// Driver Code
int main()
{
	struct Node *pHead = NULL;
	struct Node *first;

	first = insert(pHead, num());

	display(first);

	printf("\nThere are %d elements in the list\n", countNodes(first));
```

```
    return 0;
}

/*
================================================
====================
 * Output :

     How much elements you wanna enter: 5

                ********
                * NULL *
                ********
    Enter Data:     ^^^^^
    20
                *****
                    *****
                        *****
                            *****
                                *****
                                    *****
                                        *****
                                            VVVVV
    ************************************************************
****************
    ** new_node:  7083416 ||  data    :      20 ||  next    :      0 **
    ************************************************************
****************
    Enter Data:     ^^^^^
    10
                *****
```



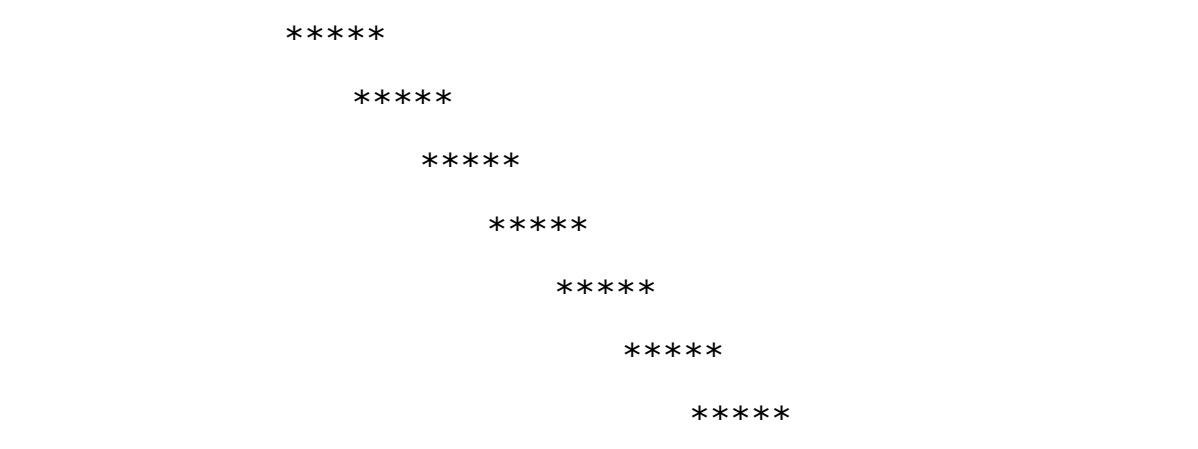
****

** new_node: 7083464 || data : 10 || next : 7083416 **

****

Enter Data: ^^^^^

30



****

** new_node: 7086768 || data : 30 || next : 7083464 **

****

Enter Data: ^^^^^

40

*****

*****

*****

*****

*****

*****

*****

VVVVV

****

** new_node: 7086784 || data : 40 || next : 7086768 **

****

Enter Data: ^^^^^

50

*****

*****

*****

*****

*****

*****

*****

VVVVV

****

** new_node: 7086800 || data : 50 || next : 7086784 **

****

========== Result ==========

disp: 7086800

disp -> data: 50

disp -> next: 7086784

disp: 7086784

disp -> data: 40

disp -> next: 7086768

disp: 7086768

disp -> data: 30

disp -> next: 7083464

disp: 7083464

disp -> data: 10

disp -> next: 7083416

disp: 7083416

disp -> data: 20

disp -> next: 0

disp: 0

There are 5 elements in the list

==========
========== */

```
/*
================================================================

 * Roll No: 30
 *
 * File:      3-insert-delete.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:  Write a menu driven program to create a singly linked list
 *                   and perform following operations on it:
 *
 *             a. Insert an element
 *                   b. Delete an element
 *                   c. Display the list
 *
 *
================================================================ */



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

// Structure
struct Node
{
      int data;
```

```
	struct Node *next;
};



// Function for -> asking choice
int askChoice()
{
	int choice;


	// printf("\n ================ Asking Choice ================\n");
	printf("\n Enter your Choice\n\n\t 0. Exit\n\t 1. Insert\n\t 2. Display\n\t 3. Delete\n");


	printf("\n\t My choice is: ");
	scanf("%d",&choice);


	return choice;
}



// Function for -> Style
void printStylish(int count)
{
	printf("\n ********");
	for (int i = 0; i < count; ++i) {
		printf("************");
	}
	printf("\n");
}
```

```
// Function for -> Couting Nodes
int countNodes(struct Node *p)
{
	int count = 0;

	while (p != NULL) {
		count ++;
		p = p -> next;
	}
	return count;
}


// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
	int i = 0, count;
	struct Node *curr = pHead;

	count = countNodes(pHead);

	if(pHead == NULL) {
		printf("\n ********* Empty List ********* \n");
		return;
	}

	printStylish(count);
```

```
	printf(" * Index *");
	while(curr != NULL) {
		printf(" %8d * ", ++i);
		curr = curr -> next;
	}

	printStylish(count);
	curr = pHead;

	printf(" * Data  *");
	while(curr != NULL) {
		printf(" %8d * ", curr -> data);
		curr = curr -> next;
	}

	printStylish(count);
	curr = pHead;

	printf(" * Next  *");
	while(curr != NULL) {
		printf(" %8d * ", curr -> next);
		curr = curr -> next;
	}

	printStylish(count);

}
```

```
// Function for -> asking any element to Insert
int askElement ()
{
	int element;

	printf("Enter data: ");
	scanf("%d",&element);

	return element;
}

void insertAtLast(struct Node **pHead, int data)
{
	struct Node *new_node = (struct Node*)malloc(sizeof(struct
Node));
	struct Node *last = *pHead;

	new_node->data = data;
	new_node->next = NULL;

	while(last->next != NULL) {
		last = last->next;
	}

	last->next = new_node;

	return;
}
```

```
void insertAtStart(struct Node **pHead, int data)
{
	struct Node *new_node = (struct Node*)malloc(sizeof(struct
Node));

	new_node->data = data;
	new_node->next = *pHead;
	*pHead = new_node;

	return;
}



int askIndex(int count)
{
	int n, index;

	printf("\n After which Index do you want to create a new Node: ");
	scanf("%d",&index);

	if(index > 0 && index <= count) {
		return index;
	}
	printf("\n Enter valid Index");
	n = askIndex(count);
}



int askIndexToDel(int count)
{
```

```
	int n, index;

	printf("\n Which Index do you want to delete: ");
	scanf("%d",&index);

	if(index > 0 && index <= count) {
		return index;
	}
	printf("Enter valid Index\n");
	n = askIndex(count);
}



void insertAtSpecific(struct Node **pHead, int data)
{
	int position, index, i = 0;
	struct Node *curr = *pHead;
	struct Node *temp;
	struct Node *new_node = (struct Node*)malloc(sizeof(struct
Node));

	display(*pHead);
	index = askIndex(countNodes(*pHead));



	while (curr != NULL) {

		if(++i == index) {

			new_node -> data = data;
```

```
                if(curr -> next == NULL) {
                        curr -> next = new_node;
                        new_node -> next = NULL;
                }

                else {
                        temp = curr -> next;
                        curr -> next = new_node;
                        new_node -> next = temp;
                }

        }

        curr = curr -> next     ;

    }

}


static void delete(struct Node **pHead)
{
    int position, index, i = 0, count;
    struct Node *curr = *pHead;
    struct Node *prev = NULL;
    if(*pHead == NULL) {
        return;
    }
```

```
display(*pHead);

count = countNodes(*pHead);
index = askIndexToDel(count);

while (curr != NULL) {

        if(++i == index) {

                if(i == 1) {

                        if(count == 0) {
                                free(*pHead);
                                *pHead = NULL;
                                printf("\n\t *** Index Deleted *** \n");
                                return;
                        }

                        else {
                                *pHead = curr -> next;
                                free(curr);
                                printf("\n\t *** Index Deleted *** \n");
                                return;
                        }

                }

                else {
```

```
                        if(curr -> next == NULL) {
                                free(curr);
                                prev -> next = NULL;

                                printf("\n\t *** Index Deleted *** \n");
                                return;
                        }

                        else {
                                prev -> next = curr -> next;
                                free(curr);
                                printf("\n\t *** Index Deleted *** \n");
                                return;
                        }

                }

        }

        prev = curr;
        curr = curr -> next     ;

    }

}


void insert(struct Node **pHead, int data)
{
```

```
	struct Node *new_node = (struct Node*)malloc(sizeof(struct
Node));
	int insertChoice;

	// If linked List is empty
	if(*pHead == NULL) {
		*pHead = new_node;
		new_node->data = data;
		new_node->next = NULL;
		display(*pHead);
		return;
	}

	printf("\nWhere do you want to insert \n\t 1. Enter at Beginning
\n\t 2. Enter at desired location \n"
		" \t 3. Enter at  end \n\n\t Enter Option: ");
	scanf("%d",&insertChoice);

	if(insertChoice == 1) {
		insertAtStart(pHead, data);
	}

	else if (insertChoice == 2) {
		insertAtSpecific(pHead, data);
	}

	else if (insertChoice == 3) {
		insertAtLast(pHead, data);
	}
```

```
	display(*pHead);
}



// Function for -> performing tasks as per choice
void doTask(struct Node **pHead, int choice)
{
	if(choice == 1) {
		printf("\n ========== Enter data you want to insert ==========\n");
		insert(pHead, askElement());
	}

	else if(choice == 2) {
		display(*pHead);
	}

	else if(choice == 3) {
		delete(&(*pHead));
		display(*pHead);
	}

	else {
		printf("\n * * * Invalid Choice * * * \n");
	}
}

// Driver Function
int main()
```

```
{
	struct Node *pHead = NULL;
	int choice = 1;

	while(choice = askChoice()) {
		doTask(&pHead, choice);
	}

	return 0;
}
```

```
/*
================================================
====================
```

Output :

```
	 Enter your Choice

		0. Exit
		1. Insert
		2. Display
		3. Delete

		My choice is: 1
```

========== Enter data you want to insert ==========

Enter data: 20

****

* Index * 1 *

****

* Data * 20 *

****

* Next * 0 *

****

Enter your Choice

0. Exit
1. Insert
2. Display
3. Delete

My choice is: 1

========== Enter data you want to insert ==========

Enter data: 10

Where do you want to insert

1. Enter at Beginning
2. Enter at desired location
3. Enter at end

Enter Option: 1

```
*********************************
* Index *       1 *        2 *
*********************************
* Data  *      10 *       20 *
*********************************
* Next  *  7607704 *        0 *
*********************************
```

Enter your Choice

0. Exit
1. Insert
2. Display
3. Delete

My choice is: 1

========== Enter data you want to insert ==========

Enter data: 40

Where do you want to insert

1. Enter at Beginning
2. Enter at desired location
3. Enter at end

Enter Option: 3

```
*********************************************
* Index *       1 *       2 *       3 *
*********************************************
* Data  *      10 *      20 *      40 *
*********************************************
* Next  *  7607704 *   7611088 *       0 *
*********************************************
```

Enter your Choice

0. Exit
1. Insert
2. Display
3. Delete

My choice is: 1

========== Enter data you want to insert ==========

Enter data: 30

Where do you want to insert

1. Enter at Beginning
2. Enter at desired location
3. Enter at end

Enter Option: 2

```
*********************************************
* Index *       1 *       2 *       3 *
```

```
*********************************************
* Data  *      10 *       20 *       40 *
*********************************************
* Next  *  7607704 *   7611088 *        0 *
*********************************************
```

After which Index do you want to create a new Node: 3

```
*****************************************************
* Index *       1 *       2 *       3 *       4 *
*****************************************************
* Data  *      10 *       20 *       40 *       30 *
*****************************************************
* Next  *  7607704 *   7611088 *   7611120 *        0 *
*****************************************************
```

Enter your Choice

0. Exit
1. Insert
2. Display
3. Delete

My choice is: 1

========== Enter data you want to insert ==========

Enter data: 50

Where do you want to insert

1. Enter at Beginning
2. Enter at desired location
3. Enter at end

Enter Option: 2

```
*************************************************************
      * Index *       1 *       2 *       3 *       4 *
*************************************************************
      * Data  *      10 *      20 *      40 *      30 *
*************************************************************
      * Next  *  7607704 *  7611088 *  7611120 *       0 *
*************************************************************
```

After which Index do you want to create a new Node: 4

```
*************************************************************************
*********
      * Index *       1 *       2 *       3 *       4 *       5 *
*************************************************************************
*********
      * Data  *      10 *      20 *      40 *      30 *      50 *
*************************************************************************
*********
```

```
      * Next  *  7607704 *   7611088 *   7611120 *   7611152 *       0
*

*********************************************************************
*********

       Enter your Choice

            0. Exit
            1. Insert
            2. Display
            3. Delete

            My choice is: 1

       ========== Enter data you want to insert ==========
       Enter data: 60

       Where do you want to insert
            1. Enter at Beginning
            2. Enter at desired location
            3. Enter at  end

            Enter Option: 2


*********************************************************************
*********
      * Index *       1 *       2 *       3 *       4 *       5 *

*********************************************************************
*********
```

```
      * Data  *      10 *       20 *       40 *       30 *       50 *

*************************************************************
*********
      * Next  *  7607704 *   7611088 *   7611120 *   7611152 *       0
*

*************************************************************
*********

      After which Index do you want to create a new Node: 6

      Enter valid Index
      After which Index do you want to create a new Node: 0

      Enter valid Index
      After which Index do you want to create a new Node: 5


*************************************************************
**********************
      * Index *       1 *       2 *       3 *       4 *       5 *       6 *

*************************************************************
**********************
      * Data  *      10 *       20 *       40 *       30 *       50 *       60
*

*************************************************************
**********************
      * Next  *  7607704 *   7611088 *   7611120 *   7611152 *
7611184 *       0 *
```

****
****

Enter your Choice

0. Exit
1. Insert
2. Display
3. Delete

My choice is: 3

****
****

Index	1	2	3	4	5	6
Data	10	20	40	30	50	60
Next	7607704	7611088	7611120	7611152	7611184	0

****
****

Which Index do you want to delete: 6

*** Index Deleted ***

```
*************************************************************
*********
      * Index *      1 *       2 *       3 *       4 *       5 *
*************************************************************
*********
      * Data  *     10 *      20 *      40 *      30 *      50 *
*************************************************************
*********
      * Next  *  7607704 *  7611088 *  7611120 *  7611152 *      0
*
*************************************************************
*********
```

Enter your Choice

0. Exit
1. Insert
2. Display
3. Delete

My choice is: 3

```
*************************************************************
*********
      * Index *      1 *       2 *       3 *       4 *       5 *
```

```
**************************************************************
*********
      * Data  *      10 *       20 *       40 *       30 *       50 *

**************************************************************
*********
      * Next  *  7607704 *   7611088 *   7611120 *   7611152 *       0
*

**************************************************************
*********

      Which Index do you want to delete: 1

            *** Index Deleted ***


**********************************************************
      * Index *       1 *        2 *        3 *        4 *

**********************************************************
      * Data  *      20 *       40 *       30 *       50 *

**********************************************************
      * Next  *  7611088 *   7611120 *   7611152 *       0 *

**********************************************************

      Enter your Choice

            0. Exit
            1. Insert
```

2. Display

3. Delete

My choice is: 3

```
**************************************************
      * Index *       1 *       2 *       3 *       4 *
**************************************************
      * Data  *      20 *      40 *      30 *      50 *
**************************************************
      * Next  *  7611088 *  7611120 *  7611152 *       0 *
**************************************************
```

Which Index do you want to delete: 3

*** Index Deleted ***

```
      **********************************************
      * Index *       1 *       2 *       3 *
      **********************************************
      * Data  *      20 *      40 *      50 *
      **********************************************
      * Next  *  7611088 *  7611152 *       0 *
      **********************************************
```

Enter your Choice

0. Exit

1. Insert

2. Display

3. Delete

My choice is: 3

```
*********************************************
* Index *       1 *       2 *       3 *
*********************************************
* Data  *      20 *      40 *      50 *
*********************************************
* Next  *  7611088 *   7611152 *       0 *
*********************************************
```

Which Index do you want to delete: 3

*** Index Deleted ***

```
*********************************
* Index *       1 *       2 *
*********************************
* Data  *      20 *      40 *
*********************************
* Next  *  7611088 *       0 *
*********************************
```

Enter your Choice

0. Exit

1. Insert

2. Display

3. Delete

My choice is: 3

```
*********************************
* Index *       1 *       2 *
*********************************
* Data  *      20 *      40 *
*********************************
* Next  *  7611088 *       0 *
*********************************
```

Which Index do you want to delete: 2

*** Index Deleted ***

```
*********************
* Index *       1 *
*********************
* Data  *      20 *
*********************
* Next  *       0 *
*********************
```

Enter your Choice

0. Exit

1. Insert

2. Display

3. Delete

My choice is: 2

****

* Index * 1 *

****

* Data * 20 *

****

* Next * 0 *

****

Enter your Choice

0. Exit

1. Insert

2. Display

3. Delete

My choice is: 3

****

* Index * 1 *

****

* Data * 20 *

****

```
* Next  *       0 *
********************

Which Index do you want to delete: 1

      *** Index Deleted ***

********* Empty List *********

Enter your Choice

      0. Exit
      1. Insert
      2. Display
      3. Delete

      My choice is: 3

********* Empty List *********

Enter your Choice

      0. Exit
      1. Insert
      2. Display
      3. Delete

      My choice is: 4
```

```
        * * * Invalid Choice * * *

        Enter your Choice

                0. Exit
                1. Insert
                2. Display
                3. Delete

                My choice is: 0


================================================
==================== */



 /*
================================================
==============

 * Roll No: 30
 *
 * File:      4-ordered.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to create an ordered linked list
 *
 *
============================================
============ */
```

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

// Structure created
struct Node
{
	int data;
	struct Node *next;
};

// Function for -> Getting Number of elements user wants to enter
int getN()
{
	int n;

	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	if(n < 1) {
		printf("Enter positive natural number\n");
		n = getN();
	}

	return n;
}
```

```
// Function for -> Getting element
int getData()
{
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}



// Function for -> Creating Node
struct Node *createNode()
{
	struct  Node *new_node;

	if((new_node = (struct Node *) malloc(sizeof(struct Node))) ==
NULL) {
		perror ("Error");
		exit(1);
	}

	return new_node;
}



// Function for inserting data order wise
struct Node *insert(struct Node *pHead, int n)
{
```

```
int i, data;

struct Node *new_node;
struct Node *curr;
struct Node *prev = NULL;

if(pHead == NULL) {
        pHead = createNode();

        pHead -> data = getData();
        pHead -> next = NULL;

        if(n == 1) { return pHead; }
}

for(i = 1; i < n; i++) {

        curr = pHead;

        data = getData();
        new_node = createNode();

        while(curr != NULL) {

                if(curr -> data >= data) {
                        new_node -> data = data;
                        new_node -> next = curr;

                        if(curr == pHead) {
```

```
                        pHead = new_node;
                    }

                    else {
                        prev -> next = new_node;
                    }

                    break;
                }

                prev = curr;
                curr = curr ->next;

            }

            if(curr == NULL) {
                prev -> next = new_node;
                new_node -> data = data;
                new_node -> next = NULL;
            }

        }

        return pHead;
}

// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
```

```
	struct Node *curr = pHead;

	printf("\n ========== Result Starts ========== \n");

	while(curr != NULL) {
		printf("\n \t	curr: %10d\n", curr);
		printf(" \t curr -> data: %10d\n", curr -> data);
		printf(" \t curr -> next: %10d\n", curr -> next);
		curr = curr -> next;
	}

	printf("\n ========== Result Ends========== \n");
}

// Driver Function
int main()
{
	struct Node *pHead = NULL;

	pHead = insert(pHead, getN());

	display(pHead);

	return 0;
}

/*
================================================
====================
 * Output :
```

```
 Enter No. of nodes you want to enter: 7

Enter Data: 30

Enter Data: 40

Enter Data: 20

Enter Data: 10

Enter Data: 60

Enter Data: 50

Enter Data: 70

 ========== Result Starts ==========

            curr:   10363584
       curr -> data:        10
       curr -> next:   10363568

            curr:   10363568
       curr -> data:        20
       curr -> next:   10360216

            curr:   10360216
       curr -> data:        30
       curr -> next:   10360264
```

```
            curr:   10360264
        curr -> data:       40
        curr -> next:   10363616

            curr:   10363616
        curr -> data:       50
        curr -> next:   10363600

            curr:   10363600
        curr -> data:       60
        curr -> next:   10363632

            curr:   10363632
        curr -> data:       70
        curr -> next:        0

    ========== Result Ends==========


================================================
==================== */



 /*
================================================
==============

 * Roll No: 30
 *
```

```
 * File:      5-reverse.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to reverse a given linked list
 *
 *
================================================
============== */

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


// Structure -> Node
struct Node
{
	int data;
	struct Node *next;
};


// Function for -> Getting Number of elements user wants to enter
int getN()
{
	int n;

	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);
```

```
	if(n < 1) {
		printf("Enter positive natural number\n");
		n = getN();
	}

	return n;
}


// Function for -> Getting element
int getData()
{
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}


// Function for -> Creating Node
struct Node *createNode()
{
	struct  Node *new_node;

	if((new_node = (struct Node *) malloc(sizeof(struct Node))) ==
NULL) {
		perror ("Error");
		exit(1);
```

```
	}

	return new_node;
}

// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;

	struct Node *new_node;
	struct Node *temp;

	if(pHead == NULL) {

		pHead = createNode();

		pHead -> data = getData();
		pHead -> next = NULL;

		if (n == 1) { return pHead; }
	}

	for(i = 1; i < n; i++) {

		temp = pHead;

		new_node = createNode();
```

```
            new_node -> data = getData();
            new_node -> next = NULL;

            while(temp->next != NULL) {
                  temp = temp->next;
            }

            temp->next = new_node;

      }

      return pHead;
}


/* gcc file-1.c file-2.c
It will give error if any static func() exists

Static word is used to avoid conflict
By default, all functions are global.
Reverse function may be used by any other program.
By writing static, this function will work on this file only */


// Function for -> Reversing data
static struct Node *reverse(struct Node *pHead)
{
   struct Node *temp;
      struct Node *prev = NULL;
```

```
        struct Node *current = pHead;

    while (current != NULL) {

        temp = current -> next;

        // Reverse current node's pointer
        current -> next = prev;

        // Move pointer one position ahead.
        prev = current;

        current = temp;

    }

    pHead = prev;

    return pHead;
}

// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
        struct Node *rev = pHead;

        printf("\n ========== Result Starts ========== \n");

        while(rev != NULL) {
```

```
            printf("\n \t rev -> data: %d\n", rev -> data);
            rev = rev -> next;
        }

        printf("\n ========== Result Ends========== \n");
}


// Driver Code
int main()
{
        struct Node *pHead = NULL;

        int num = getN();
        pHead = insertData(pHead, num);

        pHead = reverse(pHead);

        display(pHead);

        return 0;
}


/*
================================================
====================
Output :

        Enter No. of nodes you want to enter: 5
```

```
Enter Data: 10

Enter Data: 20

Enter Data: 30

Enter Data: 40

Enter Data: 50

 ========== Result Starts ==========

      rev -> data: 50

      rev -> data: 40

      rev -> data: 30

      rev -> data: 20

      rev -> data: 10

 ========== Result Ends==========


================================================
====================*/
```

```
/*
================================================================

 * Roll No: 30
 *
 * File:      6-sum.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:  Write a program to calculate the summation
 *           of all elements of the linked list.
 *
 *
================================================================ */



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



// Structure -> Node
struct Node
{
	int data;
	struct Node *next;
};
```

```
// Function for -> Getting N
int getN()
{
	int n;

	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	return n;
}


// Function for -> Getting data
int getData()
{
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}

// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;
```

```
	struct Node *new_node;
	struct Node *temp;

	if(pHead == NULL) {
		if((pHead = (struct Node *) malloc(sizeof(struct Node))) ==
NULL) {
			perror("\n Error");
			exit(1);
		}

		pHead -> data = getData();
		pHead -> next = NULL;

		if (n == 1) { return pHead; }
	}

	for(i = 1; i < n; i++) {

		temp = pHead;

		if((new_node = (struct Node *) malloc(sizeof(struct Node)))
== NULL) {
			perror("\n Error");
			exit(1);
		}

		new_node -> data = getData();
		new_node -> next = NULL;

		while(temp->next != NULL) {
```

```
                temp = temp->next;
            }

            temp->next = new_node;

        }

        return pHead;
}


// Function for -> Add all elements of Linked List
void add(struct Node *pHead)
{
        struct Node *add = pHead;
        int total = 0;

        printf("\n");
        while(add != NULL) {
            printf("%d ", add -> data);
            if(add -> next != NULL) {
                printf(" + ");
            }

            total = total + add->data;
            add = add -> next;
        }
        printf("= %d\n", total);
}
```

```
int main()
{
	struct Node *pHead;
	int num = getN();

	pHead = NULL;

	pHead = insertData(pHead, num);

	add(pHead); //Add all Nodes

	return 0;
}



/*
==========================================
====================
```

Output :

```
	Enter No. of nodes you want to enter: 6

	Enter Data: 10

	Enter Data: 50

	Enter Data: 30
```

```
        Enter Data: 40

        Enter Data: 20

        Enter Data: 60

        10  + 50  + 30  + 40  + 20  + 60 = 210



================================================
====================*/




 /*
================================================
==============

 * Roll No: 30
 *
 * File:      7-append.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:  Write a program to create two linked list and
 *                    append the second list after the first
 *
 *
================================================
============= */
```

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


// Structure -> Node
struct Node
{
	int data;
	struct Node *next;
};


// Function for -> Getting N
int getN()
{
	int n;

	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	return n;
}


// Function for -> Getting data
int getData()
{
```

```
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}

// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;
	struct Node *new_node;
	struct Node *temp;

	if(pHead == NULL) {
		if((pHead = (struct Node *) malloc(sizeof(struct Node))) == NULL) {
			perror("\n Error");
			exit(1);
		}

		pHead -> data = getData();
		pHead -> next = NULL;

		if (n == 1) { return pHead; }
	}

	for(i = 1; i < n; i++) {
```

```
		temp = pHead;

		if((new_node = (struct Node *) malloc(sizeof(struct Node)))
== NULL) {
			perror("\n ** Sufficient Memory Not Available ** \n");
			exit(1);
		}

		new_node -> data = getData();
		new_node -> next = NULL;

		while(temp->next != NULL) {
			temp = temp->next;
		}

		temp->next = new_node;

	}

	return pHead;
}


// Function for -> Appending List2 with List1
void appendList(struct Node *L1, struct Node *L2)
{
	struct Node *temp = L1;
	while(temp != NULL) {

		if(temp -> next == NULL) {
```

```
                temp -> next = L2;
                break;
            }

            temp = temp -> next;
        }
}

// Function for -> Displaying all elements of Linked List 1
void display(struct Node *pHead)
{
        struct Node *append = pHead;

        printf("\n ========== Result Starts ========== \n");

        while(append != NULL) {
                printf("\nappend -> data: %d\n", append -> data);
                append = append -> next;
        }

        printf("\n ========== Result Ends========== \n");
}

int main()
{
        struct Node *list1 = NULL;
        struct Node *list2 = NULL;

        printf("\n ========== Linked List 1 ==========\n");
```

```
    int num = getN();
    list1 = insertData(list1, num);

    printf("\n ========== Linked List 2 ==========\n");

    num = getN();
    list2 = insertData(list2, num);

    appendList(list1, list2);

    display(list1);

    return 0;
}

/*
================================================
====================
```

Output :

```
     ========== Linked List 1 ==========
    Enter No. of nodes you want to enter: 4

    Enter Data: 50

    Enter Data: 20

    Enter Data: 10
```

Enter Data: 60

========== Linked List 2 ==========

Enter No. of nodes you want to enter: 3

Enter Data: 30

Enter Data: 40

Enter Data: 70

========== Result Starts ==========

append -> data: 50

append -> data: 20

append -> data: 10

append -> data: 60

append -> data: 30

append -> data: 40

append -> data: 70

========== Result Ends==========

```
================================================
==================== */




/*
================================================
==============

 * Roll No: 30
 *
 * File:      8-swap-consecutive-using-values.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:  Write a program to swap two consecutive elements
 *                    ( Swap only values )
 *
 *
================================================
============= */



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



// Structure -> Node
```

```
struct Node
{
	int data;
	struct Node *next;
};


// Function for -> Getting N
int getN()
{
	int n;

	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	return n;
}


// Function for -> Getting data
int getData()
{
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}
```

```
// Function for -> Creating Node
struct Node *createNode()
{
	struct  Node *new_node;

	if((new_node = (struct Node *) malloc(sizeof(struct Node))) == NULL) {
		perror ("Error");
		exit(1);
	}

	return new_node;
}


// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;
	struct Node *new_node;
	struct Node *temp;

	if(pHead == NULL) {

		pHead = createNode();

		pHead -> data = getData();
		pHead -> next = NULL;
```

```
        if (n == 1) { return pHead; }
    }

    for(i = 1; i < n; i++) {

        temp = pHead;

        new_node = createNode();

        new_node -> data = getData();
        new_node -> next = NULL;


        while(temp->next != NULL) {
            temp = temp->next;
        }

        temp->next = new_node;
    }

    return pHead;
}


// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
    struct Node *iterate = pHead;
```

```
        while(iterate != NULL) {
                printf("\niterate -> data: %d\n", iterate -> data);
                printf("iterate -> data: %d\n", iterate -> next);
                iterate = iterate -> next;
        }

}


// Function for -> Swapping consecutive elements
void swap(struct Node *pHead)
{
        struct Node *curr;
        struct Node *prev = pHead;

        int swap;

        if(pHead -> next == NULL) {
                printf("Swap not possible in 1 element \n");
                exit(0);
        }

        curr = pHead -> next;

        while(curr != NULL) {

                swap = prev -> data;
                prev -> data = curr -> data;
```

```
            curr -> data = swap;

            if(curr -> next == NULL || curr -> next -> next == NULL) {
                  return;
            }

            prev = curr -> next;
            curr = curr -> next -> next;

      }

}


// Driver Code
int main()
{
      struct Node *pHead = NULL;
      int num = getN();

      pHead = insertData(pHead, num);

      printf("\n ========== Before Swap ========== \n");
      display(pHead); //Display all Nodes before swap

      swap(pHead);

      printf("\n ========== After Swap ========== \n");
      display(pHead); //Display all Nodes after swap
```

```
    return 0;
}

/*
==========
==========
```

Output :

```
    Enter No. of nodes you want to enter: 5

    Enter Data: 10

    Enter Data: 20

    Enter Data: 30

    Enter Data: 40

    Enter Data: 50

     ========== Before Swap ==========

    iterate -> data: 10
    iterate -> data: 7083464

    iterate -> data: 20
    iterate -> data: 7086768
```

```
iterate -> data: 30
iterate -> data: 7086784

iterate -> data: 40
iterate -> data: 7086800

iterate -> data: 50
iterate -> data: 0

 ========== After Swap ==========

iterate -> data: 20
iterate -> data: 7083464

iterate -> data: 10
iterate -> data: 7086768

iterate -> data: 40
iterate -> data: 7086784

iterate -> data: 30
iterate -> data: 7086800

iterate -> data: 50
iterate -> data: 0


================================================
==================== */
```

```
/*
================================================================

 * Roll No: 30
 *
 * File:      9-swap-consecutive-using-addr.c
 * Copyright: 29-Apr-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content:  Write a program to swap two consecutive elements
 *                    ( Swap only address )
 *
 *
================================================================ */



#include <stdio.h>
// #include <conio.h>
#include <stdlib.h>



// Structure -> Node
struct Node
{
	int data;
	struct Node *next;
};
```

```
// Function for -> Getting N
int getN()
{
	int n;

	printf("Enter No. of nodes you want to enter: ");
	scanf("%d",&n);

	return n;
}


// Function for -> Getting data
int getData()
{
	int data;

	printf("\nEnter Data: ");
	scanf("%d",&data);

	return data;
}


// Function for -> Creating Node
struct Node *createNode()
{
```

```
	struct  Node *new_node;

	if((new_node = (struct Node *) malloc(sizeof(struct Node))) ==
NULL) {
		perror ("Error");
		exit(1);
	}

	return new_node;
}



// Function For -> Inserting Data
struct Node *insertData(struct Node *pHead, int n)
{
	int i, data;
	struct Node *new_node;
	struct Node *temp;

	if(pHead == NULL) {

		pHead = createNode();

		pHead -> data = getData();
		pHead -> next = NULL;

		if (n == 1) { return pHead; }
	}

	for(i = 1; i < n; i++) {
```

```
            temp = pHead;

            new_node = createNode();

            new_node -> data = getData();
            new_node -> next = NULL;


            while(temp->next != NULL) {
                  temp = temp->next;
            }

            temp->next = new_node;
      }

      return pHead;
}


// Function for -> Displaying all elements of Linked List
void display(struct Node *pHead)
{
      struct Node *iterate = pHead;

      while(iterate != NULL) {
            printf("\niterate -> data: %d\n", iterate -> data);
            printf("iterate -> next: %d\n", iterate -> next);
            iterate = iterate -> next;
```

```
	}

}


// Function for -> Swapping consecutive elements
void swap(struct Node **pHead)
{
	struct Node *curr;
	struct Node *temp;
	struct Node *prev = *pHead;

	if((*pHead) -> next == NULL) {
		printf("Swap not possible in 1 element \n");
		exit(0);
	}

	curr = (*pHead) -> next;

	*pHead = curr;

	while(curr != NULL) {

		if(curr -> next == NULL) {
			curr -> next = prev;
			prev -> next = NULL;
			return;
		}
```

```
            else {

                if(curr -> next -> next ==  NULL) {
                    prev -> next = curr ->next;
                    curr -> next = prev;
                    return;
                }

                else {
                    temp = curr -> next;
                    prev -> next = curr -> next -> next;
                    curr -> next = prev;

                    if(prev -> next -> next == NULL) {
                        prev -> next -> next = temp;
                        temp -> next = NULL;
                        return;
                    }

                }

            }

            prev = temp;
            curr = curr -> next -> next;

        }

}
```

```
// Driver Code
int main()
{
	struct Node *pHead = NULL;
	int num = getN();

	pHead = insertData(pHead, num);

	printf("\n ========== Before Swap ========== \n");
	display(pHead); //Display all Nodes before swap

	swap(&pHead);

	printf("\n ========== After Swap ========== \n");
	display(pHead); //Display all Nodes after swap

	return 0;
}

/*
==========================================
==================
```

Output :

```
	Enter No. of nodes you want to enter: 7

	Enter Data: 10
```

Enter Data: 20

Enter Data: 30

Enter Data: 40

Enter Data: 50

Enter Data: 60

Enter Data: 70

========== Before Swap ==========

iterate -> data: 10
iterate -> next: 29098032

iterate -> data: 20
iterate -> next: 29098064

iterate -> data: 30
iterate -> next: 29098096

iterate -> data: 40
iterate -> next: 29098128

iterate -> data: 50
iterate -> next: 29098160

iterate -> data: 60

iterate -> next: 29098192

iterate -> data: 70

iterate -> next: 0

========== After Swap ==========

iterate -> data: 20

iterate -> next: 29098000

iterate -> data: 10

iterate -> next: 29098096

iterate -> data: 40

iterate -> next: 29098064

iterate -> data: 30

iterate -> next: 29098160

iterate -> data: 60

iterate -> next: 29098128

iterate -> data: 50

iterate -> next: 29098192

iterate -> data: 70

iterate -> next: 0

```
==========================================================
========== */


/*
**********************************************************
**********************************************************
**********************************************************
**********************************************************
**********************************************************

********** ASSIGNMENT: 4 **********
********** ASSIGNMENT: 4 **********
********** ASSIGNMENT: 4 **********

**********************************************************
**********************************************************
**********************************************************
**********************************************************
**********************************************************
*/


 /*
==========================================================
==============


 * Roll No: 30
 *
```

```
 * File:      1-alternate.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to read a line from input file
 *                  and print alternate characters in the output file.
 *                  Display appropriate message for file i/o errors.
 *
 *
================================================================ */

#include <stdio.h>

int main()
{
   int a = 1;

   char mystring[50];
   char *cptr;

   FILE *f1 = fopen("alternate.txt", "w");

   if(f1) {
      printf ("Enter a string to copy alternate texts on file: ");
      scanf  ("%s",mystring);

      cptr = mystring;
      while(*cptr != '\0') {
         if(a) {
            fprintf(f1, "%c", *cptr);
```

```
            printf("%c", *cptr);
            a = 0;
         } else {
            a = 1;
         }
         cptr++;
      }

      printf(" (write succesfull)");
      fclose (f1);
   }

   else {
      printf("Unable to open file");
   }

   printf("\n");
}



/*
================================================
=====================
 * Output :

        Enter a string to copy alternate texts on file: Test12345
        Ts135 (write succesfull)


================================================
==================== */
```

```
/*
================================================================

 * Roll No: 30
 *
 * File:      2-copy-line-by-line.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to copy the contents
 *                  of one file to another and also
 *                  print the no. of lines in the first file.
 *
 *
================================================================ */


 /* * * * * * * * * * * * * N O T E * * * * * * * * * * * * * * * *
 *
 * 1. Make sure file name "readlinebyline.txt" exists
 * 2. If "copy-readlinebyline.txt" exists, data will be overwritten
 *
 /* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */


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

int main()
{
```

```
    int no_of_lines = 0;
    char line[1000];
    FILE *f1 = fopen("readlinebyline.txt", "r");
    FILE *f2 = fopen("copy-readlinebyline.txt", "w");
    if(f1 && f2) {
                if (fgetc(f1) == EOF) {
                        printf ("No data found\n");
                    exit(0);
        }

        while(fgets(line, sizeof line, f1)) {
            no_of_lines++;
            // fputs (line, stdout);
            fputs (line, f2);
        }

        printf("\n%d lines yanked and pasted", no_of_lines);

        fclose(f1);
        fclose(f2);

    }

    else {
        ferror(f1);
        ferror(f2);
    }
    puts ("\n");
}
```

```
/*
================================================
====================
 * Output :

   5 lines yanked and pasted


================================================
==================== */
 /*
================================================
==============

 * Roll No: 30
 *
 * File:      3-find-word.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to search a particular word
 *                  in an existing file and display the
 *                  no. of occurrences and the position
 *                  of first occurrence of that word.
 *                  If the word is not found
 *                  display the appropriate message
 *
 *
================================================
============ */

  /* * * * * * * * * * * * N O T E * * * * * * * * * * * * * * * *
```

```
 *
 * 1. Make sure file name "readlinebyline.txt" exists
 *
 /* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */

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

int main()

{
    int i = 0, length = 0, count = 0, finding = 0, first_occurance = 0;
    char search_this_word[50], search;
    char c;
    FILE *f = fopen ("readlinebyline.txt", "r");

    printf("Enter the word you want to search: ");
    scanf("%s", search_this_word);

    length = strlen (search_this_word);

    if (f) {
        if(!length) {
            exit(0);
        }

        search = search_this_word[0];
```

```
        while((c = fgetc(f)) != EOF){
            if (count == 0) {
                first_occurance++;
            }

            if (search == c) {
                finding = 1;

                if(length == i + 1) {

                    i = finding = 0;
                    count++;
                    search = search_this_word[i];
                }

                else {
                    search = search_this_word[++i];
                }

            }
            else {
                finding = 0;
            }
        }

        if (count > 0) {
            printf ("\"%s\" found 1st time at %d position \n",
search_this_word, first_occurance - length + 1);
            printf ("\"%s\" found %d times\n", search_this_word, count);
        }
```

```
        else {
            printf ("\"%s\" not found\n", search_this_word);
        }
    }
    else {
        puts ("Cannot open file to read");
    }
    printf ("\n");
    return 0;
}

/*
================================================
====================
 * Data in file :

    Lorem Ipsum dolor set amet.
    Lorem Ipsum.
    dolor set amet.
    Ipsumset amet.

 * Output :



    Enter the word you want to search: set
    "set" found 1st time at 19 position
    "set" found 3 times


================================================
==================== */
```

```
/*
================================================
==============

 * Roll No: 30
 *
 * File:      4-sort-merge.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: The files DATA1 and DATA2
 *                  contain sorted list of integers.
                    Write a program to produce a third file DATA
                    which holds a single sorted merged list
                     of these two lists.
 *
 *
================================================
============= */



 /* * * * * * * * * * * * * N O T E * * * * * * * * * * * * * * * *
 *
 * 1. If       "file1.txt" exists, data will be overwritten
 * 2. If       "file2.txt" exists, data will be overwritten
 * 3. If "sorted-data.txt" exists, data will be overwritten
 *
 /* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */



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

```
#define DIGIT 3

int * readandsort()
{
    int i = 0, j, index;
    static int myarray[10];
    int swap;
    char line[1000];

    FILE *f1 = fopen ("file1.txt", "r");
    FILE *f2 = fopen ("file2.txt", "r");

    if (f1 && f2) {

        while (fgets(line, sizeof line, f1)) {
            myarray[i++] = atoi(line);
        }

        while (fgets(line, sizeof line, f2)) {
            myarray[i++] = atoi(line);
        }

        for (i = 0; i < DIGIT * 2 - 1; i++) {

            index = i;

            for (j = i + 1; j < DIGIT * 2; j++) {
```

```
            if (myarray[j] < myarray[index]) {
                index = j;
            }

        }

        swap = myarray[i];
        myarray[i] = myarray[index];
        myarray[index] = swap;
    }

}

else {
    printf("Unable to open files");
    exit (1);
}

return myarray;
}


void writeSortedData(int *p)
{
    int i;
    FILE *f = fopen ("sorted-data.txt", "w");

    for (i = 0; i < DIGIT * 2; i++) {
        fprintf(f, "%d ", *(p + i));
```

```
    }

    printf ("Write success to \"sorted-data.txt\"");
}


int main()
{
    int num = DIGIT, data, *p;
    FILE *f1 = fopen ("file1.txt", "w");
    FILE *f2 = fopen ("file2.txt", "w");


    if(f1 && f2) {
        printf("Enter %d numbers in FILE1\n", DIGIT);
        while(num--) {
            scanf   ("%d",&data);
            fprintf (f1, "%d\n", data);
        }

        num = DIGIT;

        printf("Enter %d numbers in FILE2\n", DIGIT);
        while(num--) {
            scanf   ("%d",&data);
            fprintf (f2, "%d\n", data);
        }

        fclose (f1);
```

```
        fclose (f2);

        p = readandsort();
        writeSortedData(p);
    }

    else {
        puts ("Unable to open files");
        exit(1);
    }

    printf("\n");

    return 0;
}



/*
================================================
====================

 * Output :
     Enter 3 numbers in FILE1
    12 22 11
    Enter 3 numbers in FILE2
    1 212 121
    Write success to "sorted-data.txt"


================================================
==================== */
```

```
/*
================================================
==============

 * Roll No: 30
 *
 * File:      5-repeated-characters.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to read line by line from a file
 *                  and print all the repeated characters
 *                  on the screen along with their frequency
 *
 *
================================================
============== */

#include <stdio.h>
#include <string.h>

void findOccurences(char *mystring, char data[][1])
{
   FILE *f1 = fopen("readlinebyline.txt", "r");
   char line[1000], *cptr;
   int i, break_me, count = 0, numbers[50] = {0};

   while(fgets(line, sizeof line,f1)) {
      cptr = line;
      fputs (line, stdout);
      while (*cptr != '\0') {
         break_me = 0;
```

```
        if(*cptr == '\n') {
            *cptr = '^';
        } else if(*cptr == ' ') {
            *cptr = '_';
        }
        for (i = 0; i < count; i++) {
            if(data[i][0] == (*cptr)) {
                numbers[i]++;
                break_me = 1;
                break;
            }
        }

        if (!break_me) {
            data[count][0] = (*cptr);
            count++;
            numbers[i]++;
        }

        cptr++;

    }
}
printf ("\n\n ^ is for new line\n _ stand for spaces");
printf ("\n\n");
for (i = 0; i < count; i++) {
    printf ("%2c : %2d\t\t", data[i][0], numbers[i]);
    if((i + 1) % 4 == 0) {
```

```
            printf ("\n");
            printf
("==========================================
==================\n");
        }
    }
}

int main()
{
    char mystring[50], data[50][1];

    findOccurences(mystring, data);

    printf("\n");
    return 0;
}



/*
==========================================
====================
 * In "readlinebyline.txt"
    Lorem Ipsum dolor set amet.
    Lorem Ipsum.
    dolor set amet.
    Ipsumset amet.

 * Output :
```

Lorem Ipsum dolor set amet.

Lorem Ipsum.

dolor set amet.

Ipsumset amet.

^ is for new line

_ stand for spaces

L : 2 o : 6 r : 4 e : 8

==========

m : 8 _ : 8 I : 3 p : 3

==========

s : 6 u : 3 d : 2 l : 2

==========

t : 6 a : 3 . : 4 ^ : 3

==========

==========

==========

*/

/*

==========

```
 * Roll No: 30
 *
 * File:      6-count-characters-whitespaces.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a function to read a file and
 *          count the no. of
 *          characters,
 *          spaces,
 *          tabs,
 *          newlines and
 *          no. of words in a given text file.
 *
 *
================================================================ */

#include <stdio.h>
#include <string.h>

void findOccurences(char *mystring, char data[][1])
{
   FILE *f1 = fopen("readlinebyline.txt", "r");
   char line[1000], *cptr;
   int new_lines = -1, tabs = 0, spaces = 0, words = 0, characters = 0,
others = 0;

   while(fgets(line, sizeof line,f1)) {
      cptr = line;
```

```
    while (*cptr != '\0') {

        if(*cptr == '\n') {
            new_lines++;
            cptr++;
            if(*cptr != EOF) {
                words++;
            }
            cptr--;
        }
        else if(*cptr == '\t') {
            tabs++;
        }
        else if(*cptr == ' ') {
            spaces++;
            words++;
        }
        else if((*cptr > 64 && *cptr < 91) || (*cptr > 96 && *cptr <
124)) {
            characters++;
        }
        else {
            others++;
        }
        cptr++;

    }
  }
  printf ("new_lines = %d\ntabs = %d\nspaces = %d\n"
       "words = %d\ncharacters = %d\nothers = %d\n",
```

```
        new_lines, tabs, spaces, words, characters, others);
}

int main()
{
   char mystring[50], data[50][1];

   findOccurences(mystring, data);

   printf("\n");
   return 0;
}

/*
==========================================================
====================
 * Data in txt file

   Lorem Ipsum dolor set amet.
Lorem Ipsum.
dolor set amet.
Ipsumset    amet.

 * Output :

new_lines = 2
tabs = 2
spaces = 8
words = 11
characters = 56
```

others = 4

```
================================================================ */
/*
================================================================

 * Roll No: 30
 *
 * File:      1-remove-occurances.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a function to accept a string
 *          from the keyboard and
 *          remove all occurrences of that string
 *          from a given file
 *
 *
================================================================ */

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

void copyFile()
{
   int delFile;
   char line[1000];
```

```
    FILE *f1 = fopen("tempFile.txt", "r");
    FILE *f2 = fopen("readlinebyline.txt", "w");
    if(f1 && f2) {

        while(fgets(line, sizeof line, f1)) {
            fputs (line, f2);
        }

        fclose(f1);
        fclose(f2);

        delFile = remove("tempFile.txt");

        if(delFile) {
            printf ("File not deleted");
        }
    }

    else {
        printf ("Error: ");
        ferror(f1);
        ferror(f2);
    }
}
int main()

{
    int i = 0, j, length = 0, count = 0, finding = 0;
    char search_this_word[50], search;
```

```
char c;
FILE *f = fopen ("readlinebyline.txt", "r+");
FILE *f1 = fopen ("tempFile.txt", "w");

printf("Enter the word you want to search: ");
scanf("%s", search_this_word);

length = strlen (search_this_word);

if (f) {
    if(!length) {
        exit(0);
    }

    search = search_this_word[0];

    while((c = fgetc(f)) != EOF){
            fprintf (f1, "%c", c);
        if (search == c) {
            finding = 1;
            if(length == i + 1) {
                i = finding = 0;
                count++;

                fseek (f1, -length, SEEK_CUR);
                for (j = 0; j < length; j++) {
                    //fprintf (f1, "*");
                }
```

```
                search = search_this_word[i];
            }

            else {
                search = search_this_word[++i];
            }

        }
        else {
            finding = 0;
        }
    }
    printf ("\"%s\" found %d times", search_this_word, count);
    if (count) {
        printf (" and deleted every time from the file");
    }
    fclose(f);
    fclose(f1);
    copyFile();

}
else {
    puts ("Cannot open file to read");
}
printf ("\n");
return 0;
}
```

```
/*
================================================
====================
 * In "txt" file
   Lorem Ipsum dolor set amet.
   Lorem Ipsum.
   dolor set amet.
   Ipsumset amet.

 * Output :
   Enter the word you want to search: set
   "set" found 3 times and deleted every time from the file

================================================
==================== */
 /*
================================================
==============

 * Roll No: 30
 *
 * File:      8-remove-blank-lines.c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program to remove all the blank lines
 *          from a given file.
 *
 *
================================================
============ */
```

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

void copyFile()
{
   int delFile;
   char line[1000];
   FILE *f1 = fopen("tempFile.txt", "r");
   FILE *f2 = fopen("readlinebyline.txt", "w");
   if(f1 && f2) {

      while(fgets(line, sizeof line, f1)) {
         fputs (line, f2);
      }

      fclose(f1);
      fclose(f2);

      delFile = remove("tempFile.txt");

      if(delFile) {
         printf ("File not deleted");
      }
   }

   else {
      ferror(f1);
      ferror(f2);
```

```
    }
}

int main()
{
    char prev = '\n', c;
    int count_lines = 0;
    FILE *f = fopen ("readlinebyline.txt", "r+");
    FILE *f1 = fopen ("tempFile.txt", "w");

    if (f) {
        while((c = fgetc(f)) != EOF){
            if(c == '\n' && prev == '\n') {
                count_lines = 1;
            }
            else {
                fprintf (f1, "%c", c);
            }
            prev = c;
        }
        fclose(f);
        fclose(f1);
        copyFile();

        printf ("Unwanted lines deleted");

    }
    else {
        puts ("Cannot open file to read");
```

```
    }
    printf ("\n");
    return 0;
}


/*
==========================================
====================
 in txt file (numbers repersent <line number>)
 * 1.
 * 2.
 * 3. Lorem Ipsum dolor  amet.
 * 4.
 * 5.
 * 6. Lorem Ipsum.
 * 7.
 * 8. dolor  amet.
 * 9.

 * Output :

 * Unwanted lines deleted


==========================================
==================== */
 /*
==========================================
=============
```

```
 * Roll No: 30
 *
 * File:      9-remove-comments     .c
 * Copyright: 18-May-2020 by Ajinkya Rathod(ajinzrathod)
 *
 * Content: Write a program a program to remove all the comments
 *          from a C file.
 *
 *
================================================================ */

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

void copyFile()
{
   int delFile;
   char line[1000];
   FILE *f1 = fopen("tempFile.txt", "r");
   FILE *f2 = fopen("readlinebyline.txt", "w");
   if(f1 && f2) {

      while(fgets(line, sizeof line, f1)) {
         fputs (line, f2);
      }

      fclose(f1);
      fclose(f2);
```

```
        delFile = remove("tempFile.txt");

        if(delFile) {
            printf ("File not deleted");
        }
    }

    else {
        ferror(f1);
        ferror(f2);
    }
}

int main()
{
    char prev = 'A', c;
    int stop = 0, first_occurence = 0, second_occurance = 0,
long_comments = 0;
    FILE *f = fopen ("readlinebyline.txt", "r+");
    FILE *f1 = fopen ("tempFile.txt", "w");

    if (f) {
        while((c = fgetc(f)) != EOF){
            if(long_comments == 1) {
                if (c == '/' && prev == '*') {
                    long_comments = 0;
                }
            }
            if (first_occurence == 1) {
```

```
            if (c == '*') {
                long_comments = 1;
            }
            else if(c != '/') {
                stop = 0;
            }else {
                second_occurance = 1;
            }
            first_occurence = 0;
        }
        if(stop == 1) {
            if (c == '\n') {
                stop = 0;
                second_occurance = 0;
            }
        }

        else if(c == '/') {
            first_occurence = 1;
            stop = 1;
        }

        if(stop == 0 && second_occurance == 0 && long_comments ==
0) {
            if (prev == '/' && c != '\n'){
                fprintf (f1, "/");
            }
            fprintf (f1, "%c", c);
        }
        prev = c;
```

```
        }
        fclose(f);
        fclose(f1);
        copyFile();

        printf("Comments Removed");

    }
    else {
        puts ("Cannot open file to read");
    }
    printf ("\n");
    return 0;
}
```

```
/*
==========================================================
=====================
C File:
#include<stdio.h>

/* This program adds divides by 2 */

/* // ** /* /* /* /* /* / * /* */
/////
///


int main ()
```

```
{
    int a = 10;

    // dividing a by 2
    a /= 2;

    return 0;
}
```

Output :

Comments Removed

```
/*
"New C File"

#include<stdio.h>



int main ()
{
    int a = 10;

    a /= 2;
```

```
   return 0;
}
*/
 //
================================================
==================== */




/*
================================================
==================
 Q.10
 Write a program that will generate a data file containing the
 list of  customers and their corresponding telephone numbers.
 Use a structure  variable to store the name and
 telephone number of each customer.
 Create  a data file using a sample list.

================================================
==================== */

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

struct customers
{
       char name[60],telephone[10];
};
```

```
void getdata(char *fname, struct customers cu[],int n)
{
	int i;
	FILE *fp;
	fp=fopen(fname,"w");
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}
	for(i=0;i<n;i++)
	{
		fflush(stdin);
		printf("\n Enter Name of Customer:");
		gets(cu[i].name);
		fflush(stdin);
		printf(" Enter Telephone No:");
		scanf("%[^\n]",&cu[i].telephone);

		fprintf(fp,"%s \t %s \n",cu[i].name,cu[i].telephone);
	}
		fclose(fp);
}

void displayR(char *fname,struct customers cu[],int n)
{
	int i;
	FILE *fp;
	fp=fopen(fname,"r");
```

```
        if(fp==NULL)
        {
                printf("\n Error in opening an file...");
                exit(0);
        }

        for(int i=0;i<n;i++)
        {
                fscanf(fp,"%s",&cu[i].name);
                fscanf(fp,"%s",&cu[i].telephone);

                printf("\n %s \t %s",cu[i].name,cu[i].telephone);
        }
                fclose(fp);
}


int main()
{
        int n;
        char file_name[80];

        struct customers cu[10];

        printf("\n Enter File name: ");
        scanf("%s",&file_name);

        printf("\n enter number of records: ");
        scanf("%d",&n);
```

```
    getdata(file_name,cu,n);

    printf("\n ====== Display Records ======");

    displayR(file_name,cu,n);

    return 0;

}

/*
==========================================
============
```

Output:

```
    Enter File name: records.txt

    enter number of records: 2

    Enter Name of Customer:Ajinkya
    Enter Telephone No:96387

    Enter Name of Customer:Steffi
    Enter Telephone No:78954

    ====== Display Records ======
    Ajinkya      96387
    Steffi       78954
```

```
==========================================
========== */



//
==========================================
====================



//
==========================================
====================
// Q12

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

struct Employee
{
	int empno;
	char name[20],address[50],phone[20];
	double salary;
};



void getdata(char *fname,struct Employee e1[],int n)
{
```

```
        int i;
        FILE *fp;
        fp=fopen(fname,"a");
        if(fp==NULL)
        {
                printf("\n Error in opening an file...");
                exit(0);
        }
        for(i=0;i<n;i++)
        {
                fflush(stdin);
                printf("\n Enter Employee ID: ");
                scanf("%d", &e1[i].empno);
                printf(" Enter Employee name: ");
                scanf("%s",&e1[i].name);
                fflush(stdin);
                printf(" Enter Employee address: ");
                scanf("%s",&e1[i].address);
                printf(" Enter phone number:");
                fflush(stdin);
                scanf("%s",&e1[i].phone);
                printf(" Enter Salary:");
                fflush(stdin);
                scanf("%lf",&e1[i].salary);

                fwrite(&e1[i], sizeof(e1[i]), 1, fp);

        //      fprintf(fp,"\n %d \t %s \t
%f",e1[i].empno,e1[i].name,e1[i].salary);
        }
```

```
        fclose(fp);
}

void display_namewise(char *fname,struct Employee e1[],int n)
{
    int i,cnt=0;
    char name[20];

    FILE *fp;

    fp=fopen(fname,"r");
    fseek(fp,0L,0);
    if(fp==NULL)
    {
        printf ("\n Error in opening an file...");
        exit(0);
    }

    printf("Enter Name:");
    scanf("%s",name);

    for(int i=0;i<n;i++)
    {

        fread(&e1[i],sizeof(e1[i]),1,fp);
        if(strcmp(name,e1[i].name)==0)
        {
            cnt++;
            printf("\n %d \t %s \t %s \t %s \t
%lf",e1[i].empno,e1[i].name,e1[i].address,e1[i].phone,e1[i].salary);
```

```
                }

        }
        if(cnt==0)
                printf("\n Recored Doesent exist");
        fclose(fp);
}

void display(char *fname,struct Employee e1[],int n)
{
        int i;
        FILE *fp;

        fp=fopen(fname,"r");
        fseek(fp,0L,0);
        if(fp==NULL)
        {
                printf ("\n Error in opening an file...");
                exit(0);
        }

        for(int i=0;i<n;i++)
        {
                fread(&e1[i],sizeof(e1[i]),1,fp);
                printf("\n %d \t %s \t %s \t %s \t
%lf",e1[i].empno,e1[i].name,e1[i].address,e1[i].phone,e1[i].salary);
        }
                fclose(fp);
}
```

```
void modify(char *fname,struct Employee e1[],int n)
{
	int i,empid,cnt=0;
	double sal;
	char name[20];
	FILE *fp,*fptr;

	fp=fopen(fname,"r");

	if(fp==NULL)
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}

	fptr=fopen("temp.txt","w");
	if(fp==NULL)
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}

	printf("\n Enter Employee Id:");
	scanf("%d",&empid);

	for(int i=0;i<n;i++)
	{
		fread(&e1[i],sizeof(e1[i]),1,fp);
```

```
        if(e1[i].empno==empid)
        {
                cnt++;
                fflush(stdin);
                printf("\nEnter Employee ID: ");
                scanf("%d", &e1[i].empno);
                printf(" Enter Employee name: ");
                scanf("%s",&e1[i].name);
                fflush(stdin);
                printf(" Enter Employee address: ");
                scanf("%s",&e1[i].address);
                printf(" Enter phone number:");
                fflush(stdin);
                scanf("%s",&e1[i].phone);
                printf(" Enter Salary:");
                fflush(stdin);
                scanf("%lf",&e1[i].salary);

                fwrite(&e1[i], sizeof(e1[i]), 1, fptr);
        }
        else
        {
                fwrite(&e1[i], sizeof(e1[i]), 1, fptr);
        }
}
if(cnt>0)
        printf("\n Employee Edited Successfully.");
else
        printf("\n Employee Not Exist...!!");
```

```
        fclose(fp);
        fclose(fptr);

        fp=fopen(fname,"w");

        if(fp==NULL)
        {
                printf ("\n Error in opening an file...");
                exit(0);
        }

        fptr=fopen("temp.txt","r");
        if(fp==NULL)
        {
                printf ("\n Error in opening an file...");
                exit(0);
        }
        for(int i=0;i<n;i++)
        {
                fread(&e1[i],sizeof(e1[i]),1,fptr);
                fwrite(&e1[i], sizeof(e1[i]), 1, fp);
        }

                fclose(fp);
                fclose(fptr);

}
```

```
int removeR(char *fname,struct Employee e1[],int n)
{
	int i,empid,cnt=0;
	double sal;
	char name[20];
	FILE *fp,*fptr;

	fp=fopen(fname,"r");

	if(fp==NULL)
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}

	fptr=fopen("temp.txt","w");
	if(fp==NULL)
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}

	printf("\n Enter Employee Id:");
	scanf("%d",&empid);

	for(int i=0;i<n;i++)
	{
		fread(&e1[i],sizeof(e1[i]),1,fp);
```

```
        if(e1[i].empno==empid)
        {
                cnt++;
        }
        else
        {
                fwrite(&e1[i], sizeof(e1[i]), 1, fptr);
        }
}
if(cnt>0)
        printf("\n Employee Deleted Successfully.");
else
        printf("\n Employee Not Exist...!!");


fclose(fp);
fclose(fptr);

fp=fopen(fname,"w");

if(fp==NULL)
{
        printf ("\n Error in opening an file...");
        exit(0);
}

fptr=fopen("temp.txt","r");
if(fp==NULL)
```

```
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}
	for(int i=0;i<n;i++)
	{
		fread(&e1[i],sizeof(e1[i]),1,fptr);
		fwrite(&e1[i], sizeof(e1[i]), 1, fp);
	}

		fclose(fp);
		fclose(fptr);
		if(cnt>0)
			return 1;
		else
			return 0;

}

int main()
{
	int n,ch,res;
	char file_name[80],choice='n',name[20],tele[10];
	int cnt=0;
	struct Employee e1[10];

	printf("\n Enter File name:");
	scanf("%s",&file_name);
	do
```

```
	{
		printf("\n\n 1.Add a new record.");
		printf("\n 2.Delete a record. ");
		printf("\n 3.Modify an existing record.");
		printf("\n 4.Retrieve and display an entire record for a given name.");
		printf("\n 5.Generate a complete list of all names, addresses and telephone numbers.\n");

		printf("\n Enter Your choice:");
		scanf("%d",&ch);

		switch(ch)
		{
			case 1:printf("\n Enter number of records to Add:");
				scanf("%d",&n);
				cnt=cnt+n;
				getdata(file_name,e1,n);
				break;
			case 2:
				res=removeR(file_name,e1,cnt);
				// int remove(char *fname,struct Employee e1[],int n)

				if(res)
					cnt=cnt-1;
				break;
			case 3:
				modify(file_name,e1,cnt);
				break;
```

```
                case 4:
                        display_namewise(file_name,e1,cnt);
                        break;
                case 5:
                        printf("\n Display Records");
                        display(file_name,e1,cnt);
                        break;
            }
            printf("\n Do You want to continue:");
            scanf("%s",&choice);


        }while(choice=='y');

        return 0;
}




//
==========================================
====================


//
==========================================
====================
// Q13
```

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

struct Country
{
	int id;
	char name[60];
	struct capital
	{
		char cp_name[60];
	};
	struct capital cp;
};

void Insert(char *fname,struct Country c1[],int n)
{
	int i;
	FILE *fp;
	fp=fopen(fname,"a");
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}
	for(i=0;i<n;i++)
	{
		printf("\n Enter Country id:");
```

```
            scanf("%d",&c1[i].id);

            fflush(stdin);
            printf(" Enter Country Name:");
            scanf("%s",&c1[i].name);

            fflush(stdin);
            printf(" Enter Capital Name:");
            scanf("%s",&c1[i].cp.cp_name);

            printf("\n %d \t %s \t %s
",c1[i].id,c1[i].name,c1[i].cp.cp_name);
            fwrite(&c1[i], sizeof(c1[i]), 1, fp);
      }
}

void display(char *fname,struct Country c1[],int n)
{
      int i;
      FILE *fp;

      fp=fopen(fname,"r");
      fseek(fp,0L,0);
      if(fp==NULL)
      {
            printf ("\n Error in opening an file...");
            exit(0);
      }

      for(int i=0;i<n;i++)
```

```
	{
		fread(&c1[i],sizeof(c1[i]),1,fp);
		printf("\n %d \t %s \t %s \t
",c1[i].id,c1[i].name,c1[i].cp.cp_name);
	}
		fclose(fp);
}

void main()
{
	int n,ch,res;
	char file_name[80],choice='n',name[20],tele[10];
	static int cnt=0;
	struct Country c1[10];



	printf("\n Enter File name:");
	scanf("%s",&file_name);
	printf("\n --------------------");
	printf("\n 1.Add a new Record.");
	printf("\n 2.Display a Record. ");
	printf("\n --------------------");
	do
	{
		printf("\n Enter Your choice:");
		scanf("%d",&ch);
		switch(ch)
		{
			case 1:printf("\n Enter number of records for Add:");
				scanf("%d",&n);
```

```
                        cnt=cnt+n;
                        Insert(file_name,c1,n);
                        break;
                case 2:
                        printf("\n----------------");
                        printf("\n Display Records");
                        printf("\n-----------------");
                        display(file_name,c1,cnt);
                        break;
            }
            printf("\n Do You want to continue:");
            scanf("%s",&choice);

        }while(choice=='y');

        getch();
}


// ============
// OUTPUT
// ============

//  Enter File name:country.txt

//  --------------------
//  1.Add a new Record.
//  2.Display a Record.
//  --------------------
```

```
//  Enter Your choice:1

//  Enter number of records for Add:3

//  Enter Country id:1
//  Enter Country Name:India
//  Enter Capital Name:Delhi

//  Enter Country id:2
//  Enter Country Name:Spain
//  Enter Capital Name:Madrid

//  Enter Country id:3
//  Enter Country Name:Canada
//  Enter Capital Name:Ottawa

//  Do You want to continue:y

//  Enter Your choice:2

// ----------------
//  Display Records
// -----------------
//  1     India   Delhi
//  2     Spain   Madrid
//  3     Canada Ottawa

//  Do You want to continue:n
```

```
//
==========================================
====================

//
==========================================
====================
// Q14

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>

struct Country
{
	int id;
	char name[60];
	struct capital
	{
		char cp_name[60];
	};
	struct capital cp;
};


void display(char *fname,Country c1[],int n)
{
	int i=0;
	FILE *fp;
```

```
	fp=fopen(fname,"r");

	if(fp==NULL)
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}
	fseek(fp, 0, SEEK_SET);

	while(fread(&c1[i],sizeof(c1[i]),1,fp))
	{
		printf("\n %d \t %s \t %s \t
",c1[i].id,c1[i].name,c1[i].cp.cp_name);
		i++;
	}
		fclose(fp);
}

void find_country(char *fname,Country c1[],int n)
{
	int i,cnt=0;
	char capi[60];
	FILE *fp;

	fp=fopen(fname,"r");
	fseek(fp,0L,0);
	if(fp==NULL)
	{
		printf ("\n Error in opening an file...");
```

```
            exit(0);
        }
        printf("\n Enter Capital:");
        scanf("%s",capi);

        i=0;
        while(fread(&c1[i],sizeof(c1[i]),1,fp))
        {
            if(strcmp(capi,c1[i].cp.cp_name)==0)
            {
                cnt++;
                printf("\n %d \t %s \t %s \t
",c1[i].id,c1[i].name,c1[i].cp.cp_name);
            }
            i++;
        }
        if(cnt==0)
            printf("\n Record Doesnot Exist...!");

            fclose(fp);
}
void find_capital(char *fname,Country c1[],int n)
{
        int i,cnt=0;
        char cou[60];
        FILE *fp;

        fp=fopen(fname,"r");
        fseek(fp,0L,0);
        if(fp==NULL)
```

```
	{
		printf ("\n Error in opening an file...");
		exit(0);
	}
	printf("\n Enter Country:");
	scanf("%s",cou);

	i=0;
	while(fread(&c1[i],sizeof(c1[i]),1,fp))
	{
		if(strcmp(cou,c1[i].name)==0)
		{
			cnt++;
			printf("\n %d \t %s \t %s \t
",c1[i].id,c1[i].name,c1[i].cp.cp_name);
		}
		i++;
	}
	if(cnt==0)
		printf("\n Record Doesnot Exist...!");

		fclose(fp);
}
void main()
{
	int n,ch,res;
	char file_name[80],choice='n',name[20],tele[10];
	static int cnt=0;
	struct Country c1[10];
```

```
printf("\n Enter File name:");
scanf("%s",&file_name);
printf("\n --------------------");
printf("\n 1.Display a Record.");
printf("\n 2.Determine the capital of a specified ountry.");
printf("\n 3.Determine the country whose capital is specified.");
printf("\n --------------------");
do
{
        printf("\n Enter Your choice:");
        scanf("%d",&ch);
        switch(ch)
        {
                case 1:printf("\n-----------------");
                                printf("\n Display Records");
                                printf("\n-----------------");
                                display(file_name,c1,cnt);
                                break;
                case 2:
                                find_capital(file_name,c1,cnt);
                                break;
                case 3:
                                find_country(file_name,c1,cnt);
                                break;
        }
        printf("\n Do You want to continue:");
        scanf("%s",&choice);

}while(choice=='y');
```

```
        getch();
}
/*
================================================
============
```

OUTPUT:

```
 Enter File name:country.txt


 --------------------
 1.Display a Record.
 2.Determine the capital of a specified ountry.
 3.Determine the country whose capital is specified.
 --------------------
 Enter Your choice:1


----------------
 Display Records
-----------------
 111     India    Delhi
 222     NewZealand Wellington
 333     Egypt Cairo
 444     Iran Baghdad
 Do You want to continue:y

 Enter Your choice:2

 Enter Country:NewZealand
```

```
 222     NewZealand    Wellington
 Do You want to continue:n


================================================
============ */



//
================================================
====================
// Q15

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>

void Insert(char *fname,int n)
{
        char str[100];
        int i;
        FILE *fp;
        fp=fopen(fname,"w");
        if(fp==NULL)
        {
                printf("\n Error in opening an file...");
                exit(0);
        }
```

```
	for(i=0;i<=n;i++)
	{
		fgets(str,sizeof(str),stdin);
		fputs(str,fp);
	}
	fclose(fp);
}

void get_line(char *fname,int n,int lno)
{
	char str[100];
	int i,cnt=0;
	FILE *fp;
	fp=fopen(fname,"r");
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}

	for(i=0;i<=n;i++)
	{
		fgets(str,sizeof(str),fp);
		if(i==lno)
		{
			fputs(str,stdout);
			cnt++;
		}
	}
```

```
        fclose(fp);
        //printf("\n cnt is:%d",cnt);
}


void Insert_line_at_k(char *fname,int n,int k)
{
        char str[100],str2[100],ch;
        int i,cnt=0,n2;
        FILE *fp,*fptr;
        fp=fopen(fname,"r");
        if(fp==NULL)
        {
                printf("\n Error in opening an file...");
                exit(0);
        }

        fptr=fopen("temp.txt","w");
        if(fp==NULL)
        {
                printf("\n Error in opening an file...");
                exit(0);
        }

        for(i=0;i<=n;i++)
        {
                fgets(str,sizeof(str),fp);
                fputs(str,fptr);
                if(i==k)
```

```
	{
		printf("\n Enter Number of line:");
		scanf("%d",&n2);

		for(int j=0;j<=n2;j++)
		{
			fgets(str2,sizeof(str2),stdin);
			if(j!=0)
			{
				fputs(str2,fptr);
			}
		}

		cnt++;
	}

}
fclose(fp);
fclose(fptr);


fp=fopen(fname,"w");
if(fp==NULL)
{
	printf("\n Error in opening an file...");
	exit(0);
}

fptr=fopen("temp.txt","r");
```

```
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}

	while((ch=getc(fptr))!=EOF)
	{
		fprintf(fp,"%c",ch);
	}
	fclose(fp);
	fclose(fptr);

}


void Delete_line_at_k(char *fname,int n,int k)
{
	char str[100],str2[100],ch;
	int i,cnt=0,n2;
	FILE *fp,*fptr;
	fp=fopen(fname,"r");
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}

	fptr=fopen("temp.txt","w");
```

```
if(fp==NULL)
{
	printf("\n Error in opening an file...");
	exit(0);
}

for(i=0;i<=n;i++)
{
	fgets(str,sizeof(str),fp);
	fputs(str,stdout);
	fputs(str,fptr);
	if(i==k)
	{
		printf("\n Enter Number of line:");
		scanf("%d",&n2);

		for(int j=0;j<n2;j++)
		{
			fgets(str,sizeof(str),fp);
			i++;
		}

	}

}
fclose(fp);
fclose(fptr);
```

```
	fp=fopen(fname,"w");
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}

	fptr=fopen("temp.txt","r");
	if(fp==NULL)
	{
		printf("\n Error in opening an file...");
		exit(0);
	}

	while((ch=getc(fptr))!=EOF)
	{
		fprintf(fp,"%c",ch);
	}
	fclose(fp);
	fclose(fptr);

}


void display(char *fname,int n)
{
	int i;
	FILE *fp;
	char ch;
```

```
        fp=fopen(fname,"r");
        fseek(fp,0L,0);
        if(fp==NULL)
        {
                printf ("\n Error in opening an file...");
                exit(0);
        }
        ch = fgetc(fp);
        while(ch != EOF)
        {
                        printf("%c", ch);
                        ch = fgetc(fp);
        }
        fclose(fp);
}

void main()
{
        int n,res,lno;
        char file_name[80],choice='n',name[20],tele[10],ch[10],ch2;
        static int cnt=0;

        printf("\n Enter File name:");
        scanf("%s",&file_name);
        printf("\n --------------------");
        printf("\n 1.$E-Enter new text ");
        printf("\n 2.$L-list the entire block of text");
        printf("\n 3.$Fk-find(retrieve)line number k ");
```

```
        printf("\n 4.$In-insert n lines after line number k ");
        printf("\n 5.$Dn-delete n lines after line number k ");
        printf("\n 6.$S-save the edited block of text and end the
computation");
        printf("\n --------------------");
        do
        {
                printf("\n Enter Your choice:");
                scanf("%s",&ch);
                switch(ch[0])
                {
                        case 69:printf("\n Input the number of lines to be
written : ");
                                        scanf("%d", &n);
                                        cnt=cnt+n;
                                        Insert(file_name,cnt);
                                        break;
                        case 76:
                                        printf("\n----------------");
                                        printf("\n Display Files");
                                        printf("\n----------------");
                                        display(file_name,cnt);
                                        break;
                        case 70:
                                        ch2=ch[1]-'0';
                                        lno=ch2;
                                        if(lno>cnt)
                                                printf("\n Invalid line number");
                                        else
                                                get_line(file_name,cnt,lno);
```

```
                        break;
                case 73:
                        ch2=ch[1]-'0';
                        lno=ch2;
                        if(lno>cnt)
                                printf("\n Invalid line number");
                        else
                                Insert_line_at_k(file_name,cnt,lno);
                        break;

                case 68:
                        ch2=ch[1]-'0';
                        lno=ch2;
                        if(lno>cnt)
                                printf("\n Invalid line number");
                        else
                                Delete_line_at_k(file_name,cnt,lno);

                        break;
                case 83:
                        exit(0);
                        break;
                default :
                        printf("\n Invalid Choice");
                        break;
        }
        printf("\n Do You want to continue:");
        scanf("%s",&choice);

}while(choice=='y' || choice=='Y');
```

```
        getch();
}




//
==========================================================
=====================
//
==========================================================
=====================
// Q16
// Command Line Average



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



float avg(int argc, char **argv)
{
    int i;
    float average, total = 0;
    if (argc < 2) {
        printf("Enter atleast 1 number");
        return 0;
    }
```

```
    for (i = 1; i < argc; i++) {
        total = total + atoi(argv[i]);
    }

    average = total / (argc - 1);

    return average;
}



int main(int argc, char **argv)
{
    float avrg;

    avrg = avg(argc, argv);
    printf("Average: %.2f \n", avrg);

    return 0;
}




/*
=============================================
====
 *
 ajinzrathod@ajinz:~/Documents/Assgn$ ./objectFile 78 96 84 45
    Average: 75.75
```

```
==========================================
==== */



//
==========================================
====================
// Q16
// Command Line Check Exectutable Files




#include <stdio.h>
#include <dirent.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>
#include <string.h>

int isExecutable(char fileName[])
{
   struct stat sb;
   if (stat(fileName, &sb) == 0 && sb.st_mode & S_IXUSR) {
      return 1;
   }
   else {
      return 0;
   }
}
```

```
int main(int argc, char **argv)
{
    char folderName[50];
    DIR *d;
    struct dirent *dir;
    int count = 0;

    if (argc == 2) {
        strcpy(folderName, argv[1]);
        d = opendir(folderName);
    }
    else if (argc == 1){
        printf("Checking current directory\n");
        d = opendir(".");
    }
    else {
        printf("Enter only 1 directory");
    }
    if (d) {
        while ((dir = readdir(d)) != NULL)
        {
            if(isExecutable(dir -> d_name)) {
                count++;
                printf("%s" ,dir->d_name);
                printf(" is executable\n");
            }
        }
        if(!count) {
```

```
            printf("No executables Found");
        }
    }
    else{
        printf("Directory Path Invalid\n");
    }
    return 0;
}
```

```
/*
================================================
====
```

```
ajinzrathod@ajinz:~/Documents/Assgn/cla$ ./objFile

Checking current directory
a is executable
.. is executable
. is executable
```

```
================================================
==== */
```

```
//
==========================================================
// Q16
// Command Line Factorial

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



int* factorial(int argc, char **argv, int *arr)
{
    if (argc < 2) {
        printf("Enter atleast 1 number\n");
        exit(0);
    }

    int fact(int num)
    {
        if(num == 0)
            return 1;
        return num * fact(num - 1);
    }

    int i;

    for (i = 1; i < argc; i++) {
        arr[i - 1] = fact(atoi(argv[i]));
    }
```

```
    return arr;
}


int main(int argc, char **argv)
{
    int arr[10], i;
    int *ptr = factorial(argc, argv, arr);

    puts("Factorials are as follows:\n");
    for (i = 1; i < argc; i++)
        printf("%02d: %5d\n", atoi(argv[i]), ptr[i - 1]);

    return 0;
}



/*
================================================
====
 ajinzrathod@ajinz:~/Documents/Assgn$ ./objectFile 3 5 8 2 4

    Factorials are as follows:

    03:     6
    05:   120
    08: 40320
    02:     2
    04:    24
```

```
==========================================
==== */
```

```
//
==========================================
====================
// Q16
// Command Line Find Word In current Directory


#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<dirent.h>

int findWordIn(char search_this_word[], char d_name[])
{
        int i = 0, length = 0, count = 0, finding = 0, first_occurance = 0;
        char c, search;
        FILE *f = fopen (d_name, "r");

    // printf("\nWorking on File: %s \n", d_name);

        length = strlen (search_this_word);

        if (f) {
```

```
if(!length) {
	exit(0);
}

search = search_this_word[0];

while((c = fgetc(f)) != EOF){
	if (count == 0) {
		first_occurance++;
	}

	if (search == c) {
		finding = 1;

		if(length == i + 1) {

			i = finding = 0;
			count++;
			search = search_this_word[i];
		}

		else {
			search = search_this_word[++i];
		}

	}
	else {
		finding = 0;
	}
```

```
            }


            if (count > 0) {
          // printf ("\"%s\" found 1st time at %d position \n",
search_this_word, first_occurance - length + 1);
          // printf ("\"%s\" found %d times\n", search_this_word, count);
                  return first_occurance - length + 1;
            }
            else {
          // printf ("\"%s\" not found\n", search_this_word);
                  return -1;
            }
      }
      else {
      // puts ("Cannot open file to read");
            return -1;
      }
      return -1;
}

int* findWordInDirectory(int argc, char **argv, int *arr)
{
      int count = 0;
      DIR *d;
      char wordToFind[50];
      struct dirent *dir;

      if(argc != 2) {
            printf("Enter one word to find\n");
            exit(1);
```

```
        }
        d = opendir(".");
        strcpy(wordToFind, argv[1]);

        if(d) {
                while((dir = readdir(d)) != NULL) {
                        arr[count] = findWordIn(wordToFind, dir -> d_name);
                        count += 1;
                }
                closedir(d);
        }
        return arr;

}

int main(int argc, char **argv)
{
   // Assuming max 50 files in directory
        int arr[50], count = 0;

        int *ptr = findWordInDirectory(argc, argv, arr);
   // Return Position where Word was found, else returns -1

        DIR *d;
        struct dirent *dir;

        d = opendir(".");

        if (d) {
```

```
            printf("Position    Filename\n");
            while ((dir = readdir(d)) != NULL){

                  if(ptr[count] != -1) {
                        printf("%8d    %s\n", ptr[count], dir->d_name);
                  }
                  count += 1;
            }
            closedir(d);
      }

      return 0;
}

/*
================================================================
====

ajinzrathod@ajinz:~/Documents/Assgn/cla$ ./objectFile Rollwala

Position    Filename
     606    01-alternate.c
     746    02-copy-line-by-line.c
     216    03-find-word.c
     545    04-sort-merge.c
     549    05-repeated-characters.c
     231    06-count-characters-whitespaces.c
    1055    07-remove-occurences.c
     967    08-remove-blank-lines.c
    1020    09-remove-comments.c
```

```
     884    10-telephone.c
     788    16-cla-check-executable.c
    1877    16-cla-find-word-in-directory.c


================================================================ */




//
================================================================
// Q16
// Command Line Rename File

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



int main(int argc, char **argv)
{
    if(argc != 3) {
        printf("Invalid Arguments\n");
        printf("Example\n objFile oldFile.txt newFile.txt\n");
        exit(1);
    }

    if(rename(argv[1], argv[2]) == 0)
```

```
    {
        printf("File renamed successfully.\n");
        exit(0);
    }
    printf("Cannot rename File\n");

    return 0;
}

/*
================================================
=============
 *
 * Output:
 *
 * ajinzrathod@ajinz:~/Documents/Assgn/cla$ ./objFile collegeNemes.c
collegeNames.c
 * File renamed successfully.
 *
 *
================================================
=========== */

//
================================================
====================
// Q16
// Command Line Merge Files
```

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

int copyFile(char sourceFile[], char destinationFile[])
{
    int no_of_lines = 0, success = 1;
    char line[1000];

    FILE *f1 = fopen(sourceFile, "r");
    FILE *f2 = fopen(destinationFile, "a+");

    printf("Working on \"%s\" file:\n", sourceFile);

    if(f1 && f2) {
        while(fgets(line, sizeof line, f1)) {
            no_of_lines++;
            // fputs (line, stdout);
            fputs (line, f2);
        }

        printf("%d lines yanked and pasted", no_of_lines);

        fclose(f1);
        fclose(f2);

    }
```

```
    else {
        success = 0;
        printf("No such File exists");
        // ferror(f1);
        // ferror(f2);
    }
    puts ("\n");
    return success;
}

int copyFiles(int argc, char **argv)
{
    int i, allSuccess = 1, success;

    if (argc < 3) {
        printf("Enter atleast 2 File Names\n");
        exit(1);
    }

    FILE *destinationFile = fopen(argv[argc - 1], "w");
    fclose(destinationFile);
    // // open in write mode to remove previous data


    for (i = 1; i < argc - 1; i++) {
        success = copyFile(argv[i], argv[argc - 1]);
        if(success && allSuccess) {
            allSuccess = 1;
        }
```

```
        else {
            allSuccess = 0;
        }
    }

    return allSuccess;

}

// objectFile srcFile3 srcFile2 srcFile1 ... destinationFile
int main(int argc, char **argv)
{
    int success;

    success = copyFiles(argc, argv);

    if(success) {
        printf("All File are copied Successfully\n");
    }
    else {
        printf("All files are NOT copied\n");
    }

    return 0;
}



/*
==========================================
====
```

ajinzrathod@ajinz:~/Documents/Assgn/cla$ ./a a.txt b.txt c.txt destination.txt

Working on "a.txt" file:

2 lines yanked and pasted

Working on "b.txt" file:

5 lines yanked and pasted

Working on "c.txt" file:

2 lines yanked and pasted

All File are copied Successfully

==========IF WE WRITE WRONG FILE NAME=========

ajinzrathod@ajinz:~/Documents/Assgn/cla$ ./a b.txt e2.txt c.txt d.txt

Working on "b.txt" file:

5 lines yanked and pasted

Working on "e2.txt" file:

No such File exists

Working on "c.txt" file:

3 lines yanked and pasted

All files are NOT copied

```
================================================
==== */
```