**KATHFORD INTERNATIONAL COLLEGE OF ENGINEERING AND MANAGEMENT**

**Affiliated to Tribhuvan University**

**Institute of Science and Technology**

**CONTACT MANAGEMENT SYSTEM**

**A PROJECT REPORT**

**Submitted to**

**Department of Computer Science and Information Technology**

**Kathford International College of Engineering and Management**

***In Partial fulfillment for the first semester course “Fundamental of Computer Programming (CSC-102)” of Bachelor of Science in Computer Science and Information and Technology***

***(B.Sc.CSIT)***

***Under the Supervision of Ms. Deni Shahi***

Submitted by:

Ashish Shrestha (06/072)

Bishal Thapa Shrestha (13/072)

May, 2016

**Kathford International College of Engineering and Management Balkumari, Lalitpur**

# LETTER OF APPROVAL

This is to certify that this project prepared by Ashish Shrestha and Bishal Thapa Shrestha entitled **“CONTACT MANAGEMENT SYSTEM”** in partial fulfillment of the requirements for the first semester course **Fundamental of Computer Programming (CSC 102)** of B.Sc. in Computer Science and Information Technology have been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required subject.

Ms. Deni Shahi Mr. Sushant Poudel

CSIT department HoD, CSIT Department

Balkumari, Lalitpur Balkumari, Lalitpur

External examiner Internal examiner

**Kathford International College of Engineering and Management Balkumari, Lalitpur**

# Supervisor’s Recommendation

I hereby recommend that this project prepared under my supervision by Ashish Shrestha and Bishal Thapa Shrestha entitled **“CONTACT MANAGEMENT SYSTEM”** in partial fulfillment of the requirements for the first semester course **Fundamental of Computer** **Programming (CSC-102)** of B.Sc. in Computer Science and Information Technology be processed for evaluation.

Ms. Deni Shahi

**SUPERVISOR**

CSIT department

Kathford International College of Engineering and Management Balkumari, Lalitpur

**Abstract**

The main purpose of this project is to implement the basics of C programming language and help connect people together in this age of communication. This program, Contact Management System, allows users to save important contact information of personnel, secured in a database. The program uses username and password protection to prevent. Using this program will allow the user to promptly and efficiently get in touch with any of their relatives, employees, etc allowing them to focus on more important tasks.

**Acknowledgement**

We would like to give our special thanks to the people who helped to create this short program. We would like to express our gratitude to our programming teacher, **Mr. Mekh Raj Jaishi** and supervisor **Ms. Deni Shahi** for their contributions in the development and honing of this project.

We would also like to thank **Mr. Sushant Poudel,** HoD, CSIT Department for the encouragement provided. We would also like to thank our friends for helping in making of this project.

* **Ashish Shrestha & Bishal Thapa Shrestha**

**Contents**

**Introduction of project……………………………………………………………1**

**Problem Statement………………………………………………………………...1**

**Objective……………………………………………………………………………1**

**Scope and limitations………………………………………………………………1**

**Implementation…………………………………………………………………….1-9**

**Conclusion and Enhancements……………………………………………………10**

**Appendices………………………………………………………………………….11-24**

1. **Introduction of the Project**

Contact Management System is a program based on basic C programming. This project uses file handling methodologies to store contact informations of people in a local file. This program is console based. The program uses Command Line Interface (CLI) to interact with the user. Even though CLI is not much user friendly, we have made this program as simple as possible so as to improve the user experience.

1. **Problem Statement**

The qualitative management of the contact information was not possible without this program. The main objective of this program is to store contact information in a convenient manner so that it can be retrieved anytime hence solving the problem.

1. **Objectives**

* To allow user(s) to manage contact records conveniently.
* To list saved contacts in a way that is easier to understand and access.
* To help user(s) to easily contact other person(s).

1. **Scope and limitations of Project**

* Text based interface is used.
* Multiple numbers cannot be added in a single contact.
* A touch of online feature is not provided.

1. **Implementation**

This program is a console app hence cannot be used in any devices as an app. This program is completely based on C programming language. The basic concepts of file handling and iteration have been used in this program.

**Algorithm**

1. Main Menu

Step 1: Start

Step 2: Display choices

Step 3: Ask user for choice

Step 4: If choice=0

Exit from program

If choice=1

Goto Add function

If choice=2

Goto list function

If choice=3

Goto Search function

If choice=4

Goto Edit function

If choice=5

Goto Delete function

Else

Display “Enter a valid choice”

Step 5: Stop

1. Add Function

Step 1: Start

Step 2: Open contact.txt in append mode

Step 3: Ask name for new data

If user enters space only, then goto main menu

Else, store data in structure

Step 4: Receive email, address, landline and cellphone number and store in structure

Step 5: Write structure onto contact.txt

Step 6: Close contact.txt

Step 7: Stop

1. List Function

Step 1: Start

Step 2: Open contact.txt in read mode.

Step 3: Display data in ascending order.

Step 4: Wait for user to conform exit.

Step 5: Close contact.txt

Step 6: Stop

1. Search Function

Step 1: Start

Step 2: Open contact.txt in read mode.

Step 3: Receive name from user.

Step 4: display matching results.

Step 5: Wait for user to conform exit.

Step 6: Close contact.txt

Step 7: Stop

1. Edit Function

Step 1: Start

Step 2: Open contact.txt in read mode and temp.txt in write mode.

Step 3: Receive name from user.

Step 4: Check whether given name matches data in contact.txt.

If yes, ask for new data and write it in temp.txt

If no, write data from contact.txt in temp.txt

Step 5: Close both files.

Step 6: Delete contact.txt and rename temp.txt to contact.txt.

Step 7: Stop

1. Delete Function

Step 1: Start

Step 2: Open contact.txt in read mode and temp.txt in write mode.

Step 3: Receive name from user.

Step 4: Check whether given name matches data in contact.txt.

If yes, skip the data

If no, write data from contact.txt in temp.txt

Step 5: Close both files.

Step 6: Delete contact.txt and rename temp.txt to contact.txt.

Step 7: Stop

**Flowchart**

Start

Display Choices

Ask user for choice

Switch   
Choice

Goto List function

Goto Search function

Goto Delete function

Display “Enter a Valid Choice”

Stop

Goto Add function

Goto Edit function

Choice=1

Choice=2

Choice=3

Choice=4

Choice=5

default

Figure: Flowchart for main menu

Start

Open contact.txt

Receive name

If

Name is space only

Goto main menu

Store data in structure

Receive remaining data

Receive

Write structure in file

Close file

Stop

Y

N

Figure: Flowchart for adding contact

Start

Open contact.txt

Display data in ascending order

Write structure in file

Close file

Stop

Figure: Flowchart for listing contacts

Display matched data

Start

Open contact.txt

Receive name from user

Check data in contact.txt

Stop

Figure: Flowchart for searching data

Receive new data

Start

Open contact.txt and temp.txt

Receive name from user

If

Name matches data in contact.txt

Write data from contact.txt into temp.txt

Write structure in file

Close files

Stop

Y

N

Delete contact.txt and rename temp.txt to contact .txt

Figure: Flowchart for editing contact

Start

Open contact.txt and temp.txt

Receive name from user

If

Name matches data in contact.txt

Write data from contact.txt into temp.txt

Write remaining data in file

Close files

Stop

Y

N

Delete contact.txt and rename temp.txt to contact .txt

Figure: Flowchart for deleting contact

**Header files used:**

A header file is generally used to define all of the functions, variables and constants contained in any function library might be used. Some header files used in this program are:

* stdio.h: contains standard input and output functions.
* conio.h: provides console input and output commands
* string.h: defines various functions for manipulating arrays of characters
* windows.h: contains declaration for all function in windows API

**Control Statements used:**

* if : if has been used in the program to check if the value after comparing two strings is zero or not
* switch : switch has been used to switch to multiple user defined functions from the main menu

**Loops used:**

* for loop :
* do while loop
* while loop

**User defined functions:**

* gotoxy(int x, int y) : moves the cursor to (x , y) position.
* draw\_top(int x, int y, int l) : prints top part of a box.
* draw\_bot(int x, int y, int l) : prints bottom part of a box.
* draw\_ver(int x, int y, int l): prints a vertical line.
* add() : adds a contact.
* lister(): lists all contacts.
* edit(): edits a contact.
* del(): deletes a contact.
* search(): searches a contact.

Some arrays have been made such as name[50], add[50], email[256], usr[50], pwd[50], query[50] and n[50]. Basic file handling functions like fopen, fclose, fread, fwrite, fflush etc. have been used in this program.

**6. Conclusion and Enhancement**

Hence, this is a short project for management of contacts which has some limitations. The project can be modified and made bug free. Listed enhancements can be applied to make the project more user-friendly and functional.

* + Graphical User Interface (GUI) can be used for further improvement of the user experience.
  + A feature for adding multiple numbers to a single contact can be added.
  + The contact information is stored in a local file. Further improvement in networking module may be able to provide all those contact information on-line.

**Appendices**

**Screenshots:**

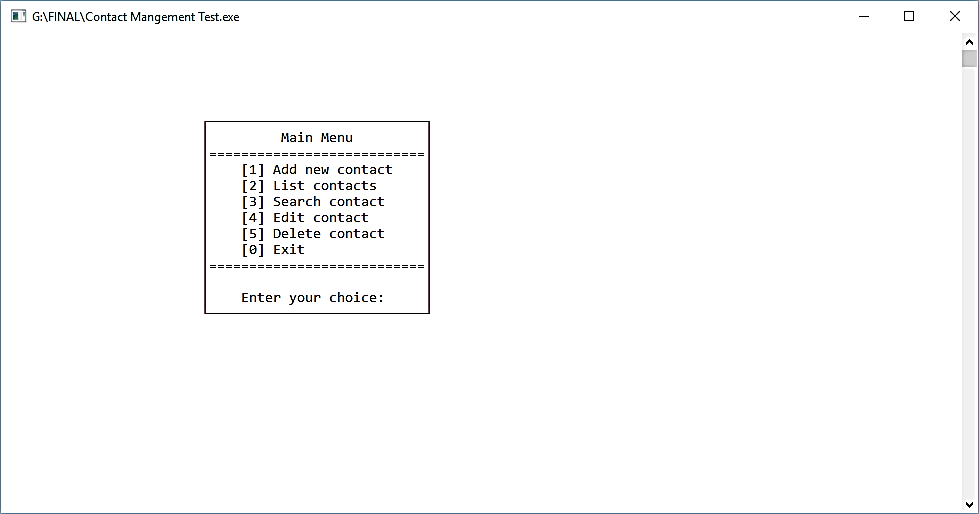
****

Figure: Screenshot of Main menu

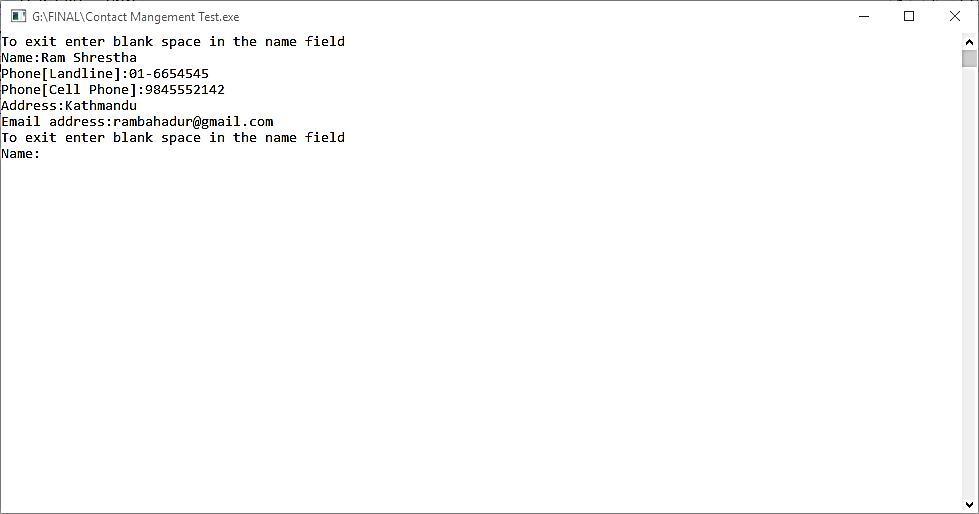
****

Figure: Screenshot of Adding contact

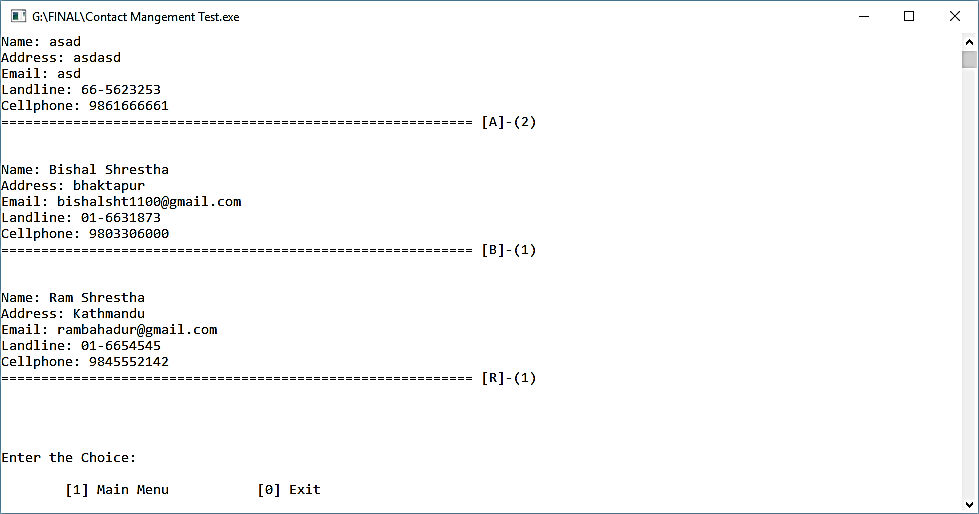


Figure: Screenshot of listing contact



Figure: Screenshot of searching contact

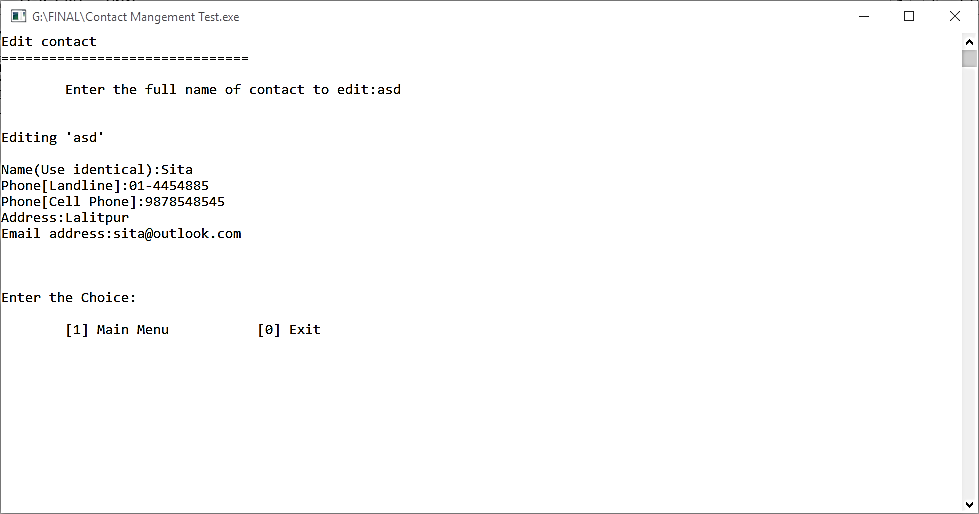


Figure: Screenshot of editing contact

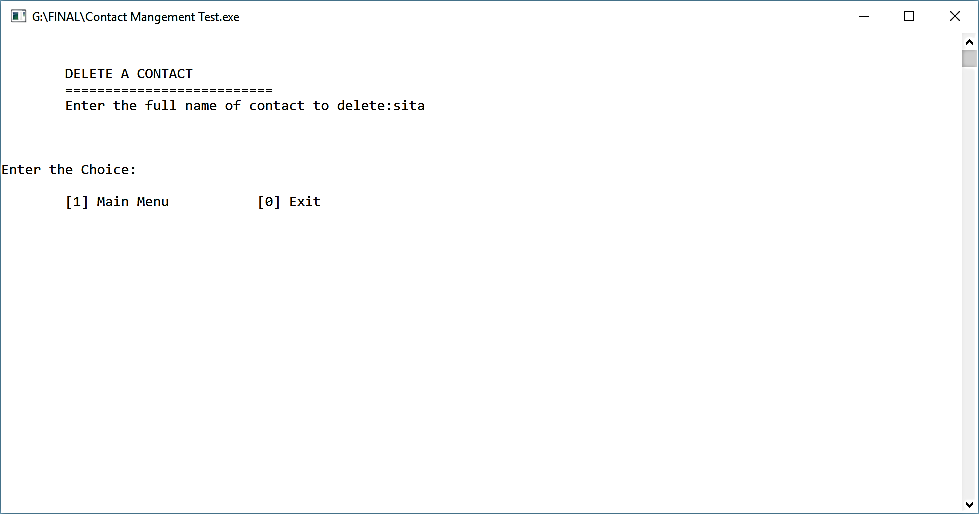


Figure: Screenshot of deleting contact

**Source Code**

**Function for addition of contact:**

void add()

{

system("cls");

fp=fopen("contact.txt","ab");

for (;;)

{

fflush(stdin);

printf("To exit enter blank space in the name field\nName:");

fflush(stdin);

gets(list.name);

if(stricmp(list.name,"")==0 || stricmp(list.name," ")==0)

break;

list.phonel=0;

printf("Phone[Landline]:");

for(i=0;i<9;i++)

{

temp=getch();

if(temp>=48 && temp<=57)

{

if(i==2)

printf("-");

if(i==0 || i>=4)

{list.phonel=(list.phonel\*10)+(temp-48);

printf("%d",temp-48);

}

else if(i==1 && temp>=49 && temp<=57)

{

list.phonel=(list.phonel\*10)+(temp-48);

printf("%d",temp-48);

}

else if(i==2 || i==3)

{

if(temp>=52 && temp<=54)

{

list.phonel=(list.phonel\*10)+(temp-48);

printf("%d",temp-48);

}

else

{

i--;

}

}

else

i--;

}

else if(i==0 && temp == 13)

break;

else

i--;

}

fflush(stdin);

list.phonem=0;

printf("\nPhone[Cell Phone]:9");

for(i=0;i<9;i++)

{

temp=getch();

if(temp>=48 && temp<=57)

{

if(i==0)

{

if(temp==55 || temp==56)

{

list.phonem=(list.phonem\*10)+(temp-48);

printf("%d",temp-48);

}

else

i--;

}

else if(i>=1)

{

list.phonem=(list.phonem\*10)+(temp-48);

printf("%d",temp-48);

}

else

i--;

}

else if(i==0 && temp == 13)

break;

else

i--;

}

fflush(stdin);

printf("\nAddress:");

gets(list.add);

fflush(stdin);

printf("Email address:");

gets(list.email);

fwrite(&list,sizeof(list),1,fp);

}

fclose(fp);

}

**Function for listing of contacts:**

void lister()

{

system("cls");

printf("\n\t\t================================\n\t\t\tLIST OF CONTACTS\n\t\t================================\n\n");

for(i=97; i<=122; i=i+1)

{

fp=fopen("contact.txt","rb");

fflush(stdin);

found=0;

while(fread(&list,sizeof(list),1,fp)==1)

{

if(list.name[0]==i || list.name[0]==i-32)

{

printf("\nName: %s\nAddress: %s\nEmail: %s\n",list.name,list.add,list.email);

printf("Landline: "); l=list.phonel/10000000;

if(l<10 && l>0)

{

printf("0");

printf("%d-",l);

l=list.phonel%10000000;

printf("%d\n",l);

}

else if(l!=0)

{

printf("%d-",l);

l=list.phonel%10000000;

printf("%d\n",l);

}

else

printf("-\n");

printf("Cellphone: ");

if(list.phonem<=90)

printf("-\n");

else

printf("9%ld\n",list.phonem);

found++;

}

}

if(found!=0)

{ printf("=========================================================== [%c]-(%d)\n\n",i-32,found);

}

fclose(fp);

}

}

**Function to search contact:**

void search()

{

system("cls");

do

{

found=0;

printf("\n\n\tCONTACT SEARCH\n\t===========================\n\tName of contact to search: ");

fflush(stdin);

gets(query);

l=strlen(query);

fp=fopen("contact.txt","rb");

system("cls");

printf("\n\nSearch result for '%s' \n===================================================\n",query);

while(fread(&list,sizeof(list),1,fp)==1)

{

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

n[i]=list.name[i];

n[l]='\0';

if(stricmp(n,query)==0)

{

printf("\nName\t: %s\nPhone1\t: %ld\nPhone2\t: %ld\nAddress\t: %s\nEmail\t: %s\n",list.name,list.phonel,list.phonem,list.add,list.email);

found++;

if (found%4==0)

{

printf("Press any key to continue...");

getch();

}

}

}

if(found==0)

printf("\nNo match found!");

else

printf("\n%d match(s) found!",found);

fclose(fp);

printf("\n Try again?\n\n\t[1] Yes\t\t[0] No\n\t");

scanf("%d",&ch);

}while(ch==1);

}

**Function to edit contact:**

void edit()

{

system("cls");

fp=fopen("contact.txt","rb");

ft=fopen("temp.dat","wb");

fflush(stdin);

printf("Edit contact\n===============================\n\n\tEnter the full name of contact to edit:");

gets(n);

while(fread(&list,sizeof(list),1,fp)==1)

{

if(stricmp(n,list.name)!=0)

fwrite(&list,sizeof(list),1,ft);

}

fflush(stdin);

printf("\n\nEditing '%s'\n\n",n);

printf("Name(Use identical):");

gets(list.name);

list.phonel=0;

printf("Phone[Landline]:");

for(i=0;i<9;i++)

{

temp=getch();

if(temp>=48 && temp<=57)

{

if(i==2)

printf("-");

if(i==0 || i>=4)

{list.phonel=(list.phonel\*10)+(temp-48);

printf("%d",temp-48);

}

else if(i==1 && temp>=49 && temp<=57)

{

list.phonel=(list.phonel\*10)+(temp-48);

printf("%d",temp-48);

}

else if(i==2 || i==3)

{

if(temp>=52 && temp<=54)

{list.phonel=(list.phonel\*10)+(temp-48);

printf("%d",temp-48);

}

else

{

i--;

}

}

else

i--;

}

else if(i==0 && temp == 13)

break;

else

i--;

}

fflush(stdin);

list.phonem=0;

printf("\nPhone[Cell Phone]:9");

for(i=0;i<9;i++)

{

temp=getch();

if(temp>=48 && temp<=57)

{

if(i==0)

{

if(temp==55 || temp==56)

{

list.phonem=(list.phonem\*10)+(temp-48);

printf("%d",temp-48);

}

else

i--;

}

else if(i>=1)

{

list.phonem=(list.phonem\*10)+(temp-48);

printf("%d",temp-48);

}

else

i--;

}

else if(i==0 && temp == 13)

break;

else

i--;

}

fflush(stdin);

printf("\nAddress:");

gets(list.add);

fflush(stdin);

printf("Email address:");

gets(list.email);

fwrite(&list,sizeof(list),1,ft);

fclose(fp);

fclose(ft);

remove("contact.txt");

rename("temp.dat","contact.txt");

}

**Function to delete contact:**

void del()

{

system("cls");

fflush(stdin);

printf("\n\n\tDELETE A CONTACT\n\t==========================\n\tEnter the full name of contact to delete:");

gets(n);

fp=fopen("contact.txt","rb");

ft=fopen("temp.dat","w");

while(fread(&list,sizeof(list),1,fp)!=0)

if (stricmp(n,list.name)!=0)

fwrite(&list,sizeof(list),1,ft);

fclose(fp);

fclose(ft);

remove("contact.txt");

rename("temp.dat","contact.txt");

}

**Function to go at certain coordinates:**

void gotoxy(int x, int y)

{

COORD coord;

coord.X = x;

coord.Y = y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);

}

**Function to draw top part of box:**

void draw\_top(int x,int y,int l)

{

gotoxy(x,y);

printf("%c",-38);

for(i=1;i<=(l-2);i++)

{

printf("%c",-60);

}

printf("%c",-65);

}

**Function to draw bottom part of box:**

void draw\_bot(int x,int y,int l)

{

gotoxy(x,y);

printf("%c",-64);

for(i=1;i<=(l-2);i++)

{

printf("%c",-60);

}

printf("%c",-39);

}

**Function to draw vertical lines in box:**

void draw\_ver(int x,int y,int l)

{

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

{

gotoxy(x,(y+i));

printf("%c\n",-77);

}

}

**References**

[1] Balagurusamy E, Programming in ANSI C, Tata McGraw-Hill Publishing Company Limited, Tenth reprint, 2009

**Bibliography**

[www.codewithc.com](http://www.codewithc.com)