VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANASANGAMA, BELAGAVI – 590018

Mini Project Report

On

Number System Converter

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering

In

Computer Science and Engineering

Submitted by

Buragana Pavankumar Dasaratharao - 1RF20CS020

Komala Abhi Keshav Royal – 1RF20CS042

RV Institute of Technology and Management®

(Affiliated to VTU, Belagavi)

JP Nagar, Bengaluru - 560076

Department of Computer Science and Engineering

RV Institute of Technology and Management®

(Affiliated to VTU, Belagavi)

JP Nagar, Bengaluru - 560076

Department of Computer Science and Engineering

CERTIFICATE

Certified that the Mini project entitled "**Number System Converter**"

carried out by

Buragana Pavankumar Dasaratharao – 1RF20CS020
Komala Abhi Keshav Royal – 1RF20CS042

are bonafied students of II Semester B.E, **RV Institute of Technology and Management** in partial fulfilment for the Bachelor of Engineering in COMPUTER SCIENCE AND ENGINEERING, of the **Visvesvaraya Technological University,** Belagavi, during the academic year 2020 - 2021. The Mini project report has been approved as it satisfies the academic requirements in respect of C Programming for Problem Solving.

Dr. Asha S. Manek
Associate Professor
Dept. CSE
RVITM, Bengaluru - 560076

ABSTRACT

A digital computer stores every information in the form of binary numbers and other number systems in various ways. There should be a way which can allow the user to get to know the conversion as well as a system which can help to convert it into different systems. Binary numbers are used to store data in the computer, electronic gates in electronic circuit, ASCII table and IP Address. Octal numbers are used for file protection in UNIX. Hexadecimal numbers for addressing memory. Decimal numbers are used in everyday life. So, the numbers which seem to be a part of everyday life mean much more to all the things around us. There should be an application or a calculator which will help the people to inter convert these numbers which will help them to create new sequences or any useful methods using these number systems and learning the concepts. The application or the calculator will help the user to make the inter conversion of these number system fast and efficient. These can save the time as well as trigger the mind to make useful contributions or new creative things using the different number systems. Therefore, the application will act as a handy assistant to now which the number means in different number system and their equivalent values in different number systems. A small data can be interpreted easily by using this application. Some of the things will become handy if not all. Numbers are a part of our life in every aspect so a person should know what is a radix and a base and get to know the conversions. These can help in innovations and there can a different usage of different number systems for every purpose.

Table of Contents

List of Symbols, Abbreviations & Nomenclature

- Radix or Base – The number of independent digits used in the number system is known as Radix or Base.
- Decimal – A radix-10 number system and has 10 different digits or symbols. These are 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. All the higher numbers after ‘9’ are represented in the terms of these 10 digits only.
- Binary – A radix-2 number system and has only 2 digits. These are 0 and 1. All the higher numbers after ‘1’ are represented in terms of these 2 digits only.
- Hexadecimal – A radix-16 number system and has 16 different digits or symbols. These are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 and A, B, C, D, E, F representing 10, 11, 12, 13, 14, 15.
- Octal – A radix-8 number system and has 8 different digits. These are 0, 1, 2, 3 ,4, 5, 6, 7.
- UNIX – UNiplexed Information Computing System
- IP – Internet Protocol

Chapter 1. Introduction

1.1. Introduction to Number System

- There are number of different numbering system which is in use for the unique ability to represent different numbers. Binary, Octal, Denary and Hexadecimal are number systems that are used in different aspects Denary number is the most commonly used number system which is frequently used in daily life. Nevertheless, each number system has associated benefits which are the reason that different number systems are used in different areas.
- Each of the number system has a fix number of representation of numbers which are used to represent the numbers like, say for example Binary numbers are represented by either one or zero, Octal numbers are represented by numbers from 0, 1, 2, 3, 4, 5, 6, 7 whereas Denary and Hexadecimal numbers are represented by the number of 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 and 0, 1,2. 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, respectively.

1.2. Importance of different Number Systems

- Binary numbers are mostly used when there are only two options available so if one is false then the other is true. Examples of binary system can be their usage to represent bits in a computer which can have only 0 or 1 value, a switch in a electric circuit which can be either on (1) or off (0). Binary system is widely used to represent situations in everyday normal life as well for example for electronic gates in electricity circuits, false or true statements can also be displaced in terms of binary digits where 0 represents false and 1 represents true states.
- Decimal numbers are frequently used in everyday life in accounting, calendar systems, financial systems or daily routine counting. The main benefits of Decimal number system are that they are easier to use as compared to other number systems and have more number to present different situations. Decimal number systems are so frequently used that a person even does not need to have a formal education to know or use them.
- Octal numbers are not that commonly used as compared to other numbers and are mostly used in computing graphics, text and famous operating system such as UNIX also uses octal numbers for their file protection system.
- Hexadecimal number are used where there are more options which needs to be represented off and are mostly commonly used in computing to represent different memory locations. They are positively contributing to the daily life and to the technology and logical world as well and needs to be understood properly if one needs to take advantage of such technology.

1.3. Literature Related to Number System

- It has been reported that in Indian mathematics (and Mayan), a study of large numbers seems to have provided the impetus for the development of a place value number system. Present day students do not have to create a number system, but they do need to understand its structure in order to develop number sense and operations. In order to investigate the value of this approach we considered the use of a combination of historical development of large numbers, number systems and modelling with concrete materials as a way of enhancing students' knowledge and understanding of place value structure.
- It's much easier to work with binary through hex than any other base system. Hexadecimal numbers are being used in representing colours in HTML and CSS. The MAC (Media Access Control) address is a number which uniquely identifies a device on the internet is also written in hexadecimal format.

1.4. Scope of the project

The numbers are used in everyday life at every instance. This project can also be used in different aspects or fields. As we know colour names are represented in hexadecimal numbers so this project can help to curate a list by converting it into a decimal format which is the most basic so that a person a remember more properly the code for each colour. The IP address are also used in binary form so this project can help a person to remember certain websites just by their numbers. UNIX uses file protection system, so the numbers can be easily remembered which are given for different execution tasks. This project can also so helpful in everyday life who is more interested in different number systems.

1.5. Subsequent chapter contents brief explanation

The subsequent chapters will explain the approach and the technology used for this project, algorithm and flowchart, results of different conditions and flows of the program, discussion about the results, and finally the references which helped to make this project a useful one.

Chapter 2. Approach Used/Technology Used

This project is built using C Programming language. Number System Converter used 4 header files and a main function with 12 user-defined functions for a total of 12 conversions which are as follows:

1. Binary to Decimal
2. Binary to Octal
3. Binary to Hexadecimal
4. Decimal to Binary
5. Decimal to Octal
6. Decimal to Hexadecimal
7. Octal to Binary
8. Octal to Decimal
9. Octal to Hexadecimal
10. Hexadecimal to Binary
11. Hexadecimal to Decimal
12. Hexadecimal to Octal

This program is of total 684 lines of code which allows the user to choose the number system conversion whichever the user wants and then take the input and display the equivalent number in another system. It has been built using switch statements, go-to statements, functions and many more entities. It will also check whether the input given is correct or there is an error in the input like if the number given as input is of the valid number system or not.

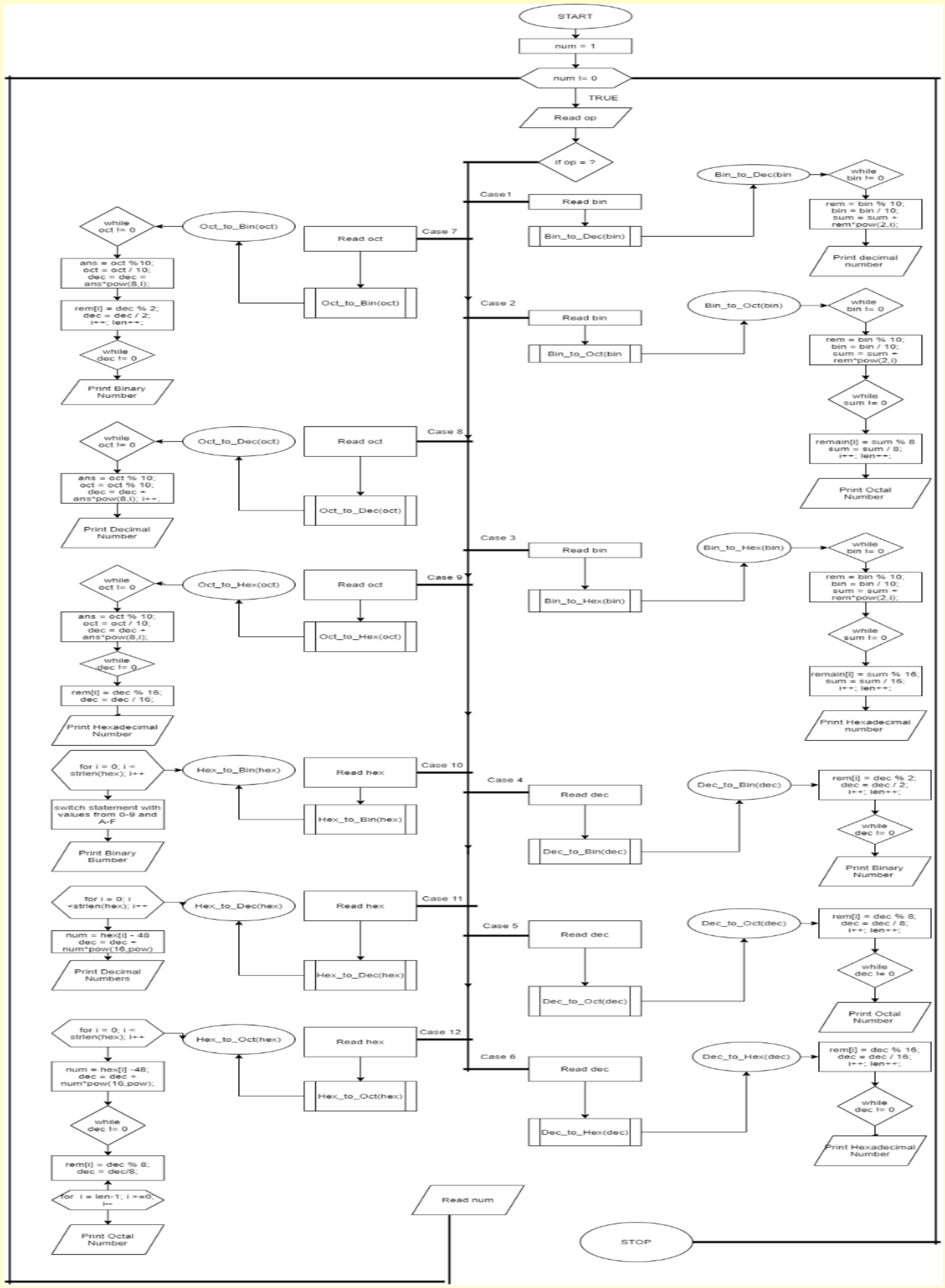
This is a brief about the project and about its code. The following sections will help to know about the program more deeply which will include the flowchart, algorithm and the code implementation.

Flowchart is the step-by-step representation of the program using different shapes and arrows for the process in progression and each shape will represent a certain process.

Algorithm is the pseudo-code of the program which will explain the code sequence of the program in an easy way from start to end of the program.

Code Implementation will represent the code of the whole program in that section which will help the user to learn the concepts in a better way and can also update it with new features in the future.

2.1 Flowchart



2.2 Algorithm

Step 1: Start

Step 2: Include header files (stdio.h, math.h, conio.h, string.h)

- Stdio.h - The **stdio.h** header defines three variable types, several macros, and various functions for performing input and output.
- Math.h - The **math.h** header defines various mathematical functions and one macro. All the functions available in this library take **double** as an argument and return **double** as the result.
- Conio.h - The conio.h header file used in C programming language contains functions for console input/output. Some of its most commonly used functions are clrscr, getch, getche, etc.
- String.h - The **string.h** header defines one variable type, one macro, and various functions for manipulating arrays of characters.

```
Step 3: main() function
        num = 1;
Step 4: while (num != 0)
        [Read op] choosing the type of conversion from 1-12
Step 5: switch case
        To perform binary to decimal if op = 1
        [Read bin] taking input in binary number system and checking if it is valid
        Call function Bin_to_Dec(bin)
Step 6: To perform binary to octal if op = 2
        [Read bin] taking input in binary number system and checking if it is valid
        Call function Bin_to_Oct(bin)
Step 7: To perform binary to hexadecimal if op = 3
        [Read bin] taking input in binary number system and checking if it is valid
        Call function Bin_to_Hex(bin)
Step 8: To perform decimal to binary if op = 4
        [Read dec] taking input in decimal number system and checking if it is valid
        Call function Dec_to_Bin(dec)
Step 9: To perform decimal to octal if op = 5
        [Read dec] taking input in decimal number system and checking if it is valid
        Call function Dec_to_Oct(dec)
Step 10: To perform decimal to hexadecimal if op = 6
        [Read bin] taking input in decimal number system and checking if it is valid
        Call function Dec_to_Hex(dec)
Step 11: To perform octal to binary if op = 7
        [Read oct] taking input in octal number system and checking if it is valid
        Call function Oct_to_Bin(oct)
Step 12: To perform octal to decimal if op = 8
        [Read oct] taking input in octal number system and checking if it is valid
        Call function Oct_to_Dec(oct)
Step 13: To perform octal to hexadecimal if op = 9
        [Read oct] taking input in octal number system and checking if it is valid
        Call function **Oct_to_Hex(oct)**
Step 14: To perform hexadecimal to binary if op = 10
        [Read hex] taking input in hexadecimal number system and checking if it is valid
        Call function **Hex_to_Bin(hex)**
Step 15: To perform hexadecimal to decimal if op = 11
        [Read hex] taking input in hexadecimal number system and checking if it is valid
        Call function **Hex_to_Dec(hex)**
Step 16: To perform hexadecimal to octal if op = 12
        [Read hex] taking input in hexadecimal number system and checking if it is valid
        Call function **Hex_to_Oct(hex)**
Step 17: [Read num] to ask if the user continue to or not
Step 18: Stop
```

Bin_to_Dec(bin)

```
Step 1: Start
Step 2: rem, sum = 0, i = 0;
        while(bin != 0)
                rem = bin % 10;
                bin = bin / 10;
                sum = sum + rem*pow(2,i);
                i++;
        Step 3: Print sum which is the decimal equivalent of entered binary number
```

Bin_to_Oct(bin)

```
Step 1: Start
Step 2: i = 0, rem, sum = 0, rem[100], len=0
        while (bin != 0)
                rem = bin % 10;
                bin = bin / 10;
                sum = sum + rem*pow(2,i);
                i++;
Step 3: i = 0;
        while (sum != 0)
                rem[i] = sum%8;
                sum = sum / 8;
                i++; len++;
Step 4: for (i = len – 1; i >= 0; i--)
                Print rem[i] which is the octal equivalent of entered binary number
```

Bin_to_Hex(bin)

```
Step 1: Start
Step 2: i = 0, rem, sum = 0, rem[100], len = 0
        While (bin != 0)
                rem = bin % 10;
                bin = bin / 10;
                sum = sum + rem*pow(2,i);
                i++;
Step 3: i = 0;
        while (sum != 0)
                rem[i] = sum%16;
                sum = sum / 16;
                i++; len++;
Step 4: for (i = len – 1; i >= 0; i--)
                Switch(rem[i])
                Case 10: print A;
                Case 11: print B;
                Case 12: print C;
                Case 13: print D;
                Case 14: print E;
                Case 15: print F;
                Default: print rem[i];
                Print rem[i] which is the hexadecimal equivalent of entered binary number
```

Dec_to_Bin(dec)

```
Step 1: Start
Step 2: i, rem[50], len = 0;
        do rem[i] = dec % 2;
           dec = dec / 2;
           i++; len++;
        while (dec != 0)
Step 3: for (i = len – 1; i >= 0; i--)
                Print rem[i] which is the binary equivalent of entered decimal number
```

Dec_to_Oct(dec)

```
Step 1: Start
Step 2: : i, rem[50], len = 0;
        do rem[i] = dec % 8;
           dec = dec / 8;
           i++; len++;
        while (dec != 0)
Step 3: for (i = len – 1; i >= 0; i--)
```

```
                    Print rem[i] which is the octal equivalent of entered decimal number
```

Dec_to_Hex(dec)

```
Step 1: Start
Step 2: : i, rem[50], len = 0;
          do rem[i] = dec % 16;
             dec = dec / 16;
              i++; len++;
          while (dec != 0)
Step 3: for (i = len – 1; i >= 0; i--)
                    Switch(rem[i])
                    Case 10: print A;
                    Case 11: print B;
                    Case 12: print C;
                    Case 13: print D;
                    Case 14: print E;
                    Case 15: print F;
                    Default: print rem[i];
                    Print rem[i] which is the hexadecimal equivalent of entered decimal number
```

Oct_to_Bin(oct)

```
Step 1: Start
Step 2: i = 0, ans, dec = 0, rem[100], len=0
          while (oct != 0)
                    ans = oct % 10;
                    oct = oct / 10;
                    dec = dec + ans*pow(8,i);
                    i++;
Step 3: i = 0;
          while (dec != 0)
                    ans[i] = dec%2;
                    dec = dec / 2;
                    i++; len++;
Step 4: for (i = len – 1; i >= 0; i--)
                    Print rem[i] which is the binary equivalent of entered octal number
```

Oct_to_Dec(oct)

```
Step 1: Start
Step 2: dec = 0, ans, rem, i = 0;
          While (oct != 0)
```

```
                ans = oct % 10;
                oct = oct / 10;
                dec = dec + ans*pow(8,i);
                i++;
Step 3: Print dec which is the decimal equivalent of entered octal number
```

Oct_to_Hex(oct)

```
Step 1: Start
Step 2: i = 0, ans, dec = 0, rem[100], len=0
        while (oct != 0)
                ans = oct % 10;
                oct = oct / 10;
                dec = dec + ans*pow(8,i);
                i++;
Step 3: i = 0;
        while (dec != 0)
                ans[i] = dec%16;
                dec = dec / 16;
                i++; len++;
Step 4: for (i = len – 1; i >= 0; i--)
                Switch(rem[i])
                Case 10: print A;
                Case 11: print B;
                Case 12: print C;
                Case 13: print D;
                Case 14: print E;
                Case 15: print F;
                Default: print rem[i];
                Print rem[i] which is the hexadecimal equivalent of entered octal number
```

Hex_to_Bin(hex)

```
Step 1: Start
Step 2: i = 0;
        Print equivalent binary number
        for (i = 0; i < strlen(hex); i++)
        Switch (hex[i])
        Case 0: print 0000;
        Case 1: print 0001;
        Case 2: print 0010;
        Case 3: print 0011;
        Case 4: print 0100;
        Case 5: print 0101;
        Case 6: print 0110;
        Case 7: print 0111;
        Case 8: print 1000;
        Case 9: print 1001;
        Case A or a: print 1010;
        Case B or b: print 1011;
        Case C or c: print 1100;
        Case D or d: print 1101;
        Case E or e: print 1110;
        Case F or f: print 1111;
        Default: print invalid number;
```

Hex_to_Dec(hex)

```
Step 1: Start
Step 2: i, num = 0, power = 0, dec = 0;
        for ( i = strlen(hex)-1; i >=0; i++)
                if (hex[i] = A or a) then num = 10;
                if (hex[i] = B or b) then num = 11;
                if (hex[i] = C or c) then num = 12;
                if (hex[i] = D or d) then num = 13;
                if (hex[i] = E or e) then num = 14;
                if (hex[i] = F or f) then num = 15;
                else num = hex[i] – 48;
        dec = dec + num*pow(16, power);
        power++;
Step 3: print dec which is the decimal equivalent of entered hexadecimal number
```

Hex_to_Oct(hex)

```
Step 1: Start
Step 2: i, len, num = 0, power = 0, dec = 0; rem[100];
        for ( i = strlen(hex)-1; i >=0; i++)
                if (hex[i] = A or a) then num = 10;
                if (hex[i] = B or b) then num = 11;
                if (hex[i] = C or c) then num = 12;
                if (hex[i] = D or d) then num = 13;
                if (hex[i] = E or e) then num = 14;
                if (hex[i] = F or f) then num = 15;
                else num = hex[i] – 48;
```

```
            dec = dec + num*pow(16, power);
            power++;
Step 3: i = 0, len = 0;
            While (dec != 0)
                        rem[i] = dec % 8;
                        dec = dec / 8;
                        i++; len++;
Step 4: for (i = len – 1; i >= 0; i--)
                        Print rem[i] which is the octal equivalent of entered hexadecimal number
```

2.3. Code Implementation

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

long int Bin_to_Dec(long int);    //1:BINARY TO DECIMAL
long int Bin_to_Oct(long int);    //2:BINARY TO OCTAL
long int Bin_to_Hex(long int);   //3:BINARY TO HEXA-DECIMAL
long int Dec_to_Bin(long int);   //4:DECIMAL TO BINARY
long int Dec_to_Oct(long int);   //5:DECIMAL TO OCTAL
long int Dec_to_Hex(long int);  //6:DECIMAL TO HEXA-DECIMAL
long int Oct_to_Bin(long int);   //7:OCTAL TO BINARY
long int Oct_to_Dec(long int);  //8:OCTAL TO DECIMAL
long int Oct_to_Hex(long int);  //9:OCTAL TO HEXA-DECIMAL
void Hex_to_Bin(char []);         //10:HEXA-DECIMAL TO BINARY
void Hex_to_Dec(char []);        //11:HEXA-DECIMAL TO DECIMAL
void Hex_to_Oct(char []);        //12:HEXA-DECIMAL TO OCTAL

int main()
{
   int op,num=1,check; //local declarations
   long int bin,oct,dec; // local declarations
   char hex[100];
   int i,j,space;  //  FOR PATTERN

   printf("\t\tWELCOME  TO  NUMBER  SYSTEM  CONVERSION\n\n");  //  Greeting
   message
```

```
while(num!=0)
{
        printf("\t\t>>>>>> CHOOSE THE CONVERSION <<<<<<\n\n"); // Designing the opening interface with choices

    printf("=> BINARY <=\n"); // For all conversions from binary to decimal, octal and hexadecimal

    printf("1: Binary to Decimal.\n2: Binary to Octal.\n3: Binary to Hexa-Decimal.\n");

    printf("\n=> DECIMAL <=\n"); // For all conversions from decimal to binary, octal, hexadecimal

    printf("4: Decimal to Binary.\n5: Decimal to Octal.\n6: Decimal to Hexa-Decimal.\n");

    printf("\n=> OCTAL <=\n"); // For all conversions from octal to binary, decimal and hexadecimal

    printf("7: Octal to Binary.\n8: Octal to Decimal.\n9: Octal to Hexa-Decimal.\n");

    printf("\n=> HEXA-DECIMAL <=\n"); // For all conversions from hexadecimal to binary, decimal and octal

    printf("10: Hexa-Decimal to Binary.\n11: Hexa-Decimal to Decimal.\n12: Hexa-Decimal to Octal.\n");

    printf("\nENTER YOUR CHOICE: "); // Taking input from the user for the conversion he/she wants
    scanf("%d",&op);

    switch(op)
    {
      case 1:
        printf("\n***BINARY TO DECIMAL***\n"); // Binary to decimal conversion
        D:
        printf("\nEnter the Number in Binary form (0s & 1s): "); // Taking inputs in only form of 0s and 1s
        scanf("%ld",&bin);
        // CHECKING INPUT IS IN BINARY FORM
        check=bin;

        while(check!=0)
        {
          num=check%10;
          if(num>1)
          {
            printf("\n%d IS NOT BINARY NUMBER.\n",bin); // If any numbers other
than 0 & 1 is typed then error
            printf("***TRY AGAIN****\n");
            goto D; // goto statement takes to D label
          }
          else
          check=check/10;
        }

        Bin_to_Dec(bin);
        break;

      case 2:
        printf("\n***BINARY TO OCTAL***\n"); // Binary to Octal conversion
        E:
        printf("\nEnter the Number in Binary form (0s & 1s): "); // Taking inputs in only
0s and 1s
        scanf("%ld",&bin);
        // CHECKING INPUT IS IN BINARY FORM
        check=bin;

        while(check!=0)
        {
          num=check%10;
          if(num>1)
          {
            printf("\n%d IS NOT BINARY NUMBER.\n",bin); // Any number other
than 0 and 1 is found then error
            printf("***TRY AGAIN****\n");
            goto E; // goto statement takes to E label
          }
          else
          check=check/10;
        }

        Bin_to_Oct(bin);
        break;

      case 3:
        printf("\n***BINARY TO HEXA-DECIMAL***\n"); // Binary to Hexadecimal
        F:
        printf("\nEnter the Number in Binary form (0s & 1s): "); // Taking inputs in only
0s and 1s
        scanf("%ld",&bin);
        // CHECKING INPUT IS IN BINARY FORM
        check=bin;

        while(check!=0)
        {
          num=check%10;
          if(num>1)
          {
            printf("\n%d IS NOT BINARY NUMBER.\n",bin); // Any number other
than 0 and 1 is found then error
            printf("***TRY AGAIN****\n");
            goto F;
          }
          else
          check=check/10;
        }

        Bin_to_Hex(bin);
        break;

      case 4:
        printf("\n***DECIMAL TO BINARY***\n"); // Decimal to Binary
        printf("\nEnter the Number in Decimal form (0 to 9): "); // Taking input
        scanf("%ld",&dec);
        Dec_to_Bin(dec); break;

      case 5:
        printf("\n***DECIMAL TO OCTAL***\n"); // Decimal to Octal
        printf("\nEnter the Number in Decimal form (0 to 9): "); // Taking input
        scanf("%ld",&dec);
        Dec_to_Oct(dec); break;

    case 6:
        printf("\n***DECIMAL TO HEXA-DECIMAL***\n"); // Decimal to
Hexadecimal
        printf("\nEnter the Number in Decimal form (0 to 9): "); // Taking input
        scanf("%ld",&dec);
        Dec_to_Hex(dec); break;

    case 7:
        printf("\n***OCTAL TO BINARY***\n"); // Octal to Binary
        A:
        printf("\nEnter the Number in Octal form (0 to 7): "); // Taking input number
containing digits from 0-7
        scanf("%ld",&oct);
        // CHECKING INPUT IS IN OCTAL FORM
        check=oct;

        while(check!=0)
        {
            num=check%10;
            if(num>7)
            {
                printf("\n%d IS NOT OCTAL NUMBER.\n",num); // Error if number
contains digits other than 0-7
                goto A;
            }
            else
            {
            check=check/10;
            i++;
            }
        }
        Oct_to_Bin(oct); break;

    case 8:
        printf("\n***OCTAL TO DECIMAL***\n"); // Octal to Decimal
        B:
        printf("\nEnter the Number in Octal form (0 to 7): "); // Taking input number
containing digits from 0-7
```

```
        scanf("%ld",&oct);
        // CHECKING INPUT IS IN OCTAL FORM
        check=oct;

        while(check!=0)
        {
          num=check%10;
          if(num>7)
          {
            printf("\n%d IS NOT OCTAL NUMBER.\n",num); // Error if number
contains digits other than 0-7
            goto B;
          }
          else
          {
          check=check/10;
          i++;
          }
        }
        Oct_to_Dec(oct); break;

      case 9:
        printf("\n***OCTAL TO HEXA-DECIMAL***\n"); // Octal to Hexadecimal
        C:
        printf("\nEnter the Number in Octal form (0 to 7): "); // Taking input number
containing digits from 0-7
        scanf("%ld",&oct);
        // CHECKING INPUT IS IN OCTAL FORM
        check=oct;

        while(check!=0)
        {
          num=check%10;
          if(num>7)
          {
            printf("\n%d IS NOT OCTAL NUMBER.\n",num); // Error if number
contains digits other than 0-7
            goto C;
          }
          else
          {
          check=check/10;
          i++;
          }
        }
        Oct_to_Hex(oct); break;

      case 10:
        printf("\n***HEXA-DECIMAL TO BINARY***\n"); // Hexadecimal to Binary
        X:
        printf("\nEnter the Number in Hexa-Decimal form: "); // Taking input
        scanf("%s",&hex);
        //check
        for(i=strlen(hex)-1;i>=0;i--)
        {
          if(hex[i]>'f' && hex[i]<='z' || hex[i]>'F'&& hex[i]<='Z')
          {
            printf("\nYou have to Enter Hexa-Decimal Number.\n");
            printf("'%c' IS NOT Hexa-Decimal Number.\n",hex[i]); // If the entered
number is not a hexa-decimal number then it is a error displayed
            goto X;
          }
        }
        Hex_to_Bin(hex); break;

      case 11:
        printf("\n***HEXA-DECIMAL TO DECIMAL***\n"); // Hexadecimal to
Decimal
        Y:
        printf("\nEnter the Number in Hexa-Decimal form: "); // Taking input
        scanf("%s",&hex);
        //check
        for(i=strlen(hex)-1;i>=0;i--)
        {
          if(hex[i]>'f' && hex[i]<='z' || hex[i]>'F'&& hex[i]<='Z')
          {
            printf("\nYou have to Enter Hexa-Decimal Number.\n");
            printf("'%c' IS NOT Hexa-Decimal Number.\n",hex[i]); // If the entered
number is not a hexa-decimal number then it is a error displayed
            goto Y;
```

```
                    }
                }
                Hex_to_Dec(hex); break;

            case 12:
                printf("\n***HEXA-DECIMAL TO OCTAL***\n"); // Hexadecimal to Octal
                Z:
                printf("\nEnter the Number in Hexa-Decimal form: "); // Taking input
                scanf("%s",&hex);
                //check
                for(i=strlen(hex)-1;i>=0;i--)
                {
                    if(hex[i]>'f' && hex[i]<='z' || hex[i]>'F'&& hex[i]<='Z')
                    {
                        printf("\nYou have to Enter Hexa-Decimal Number.\n");
                        printf("'%c' IS NOT Hexa-Decimal Number.\n",hex[i]); // If the entered
number is not a hexa-decimal number then it is a error displayed
                        goto Z;
                    }
                }
                Hex_to_Oct(hex); break;

            default:
                printf("\n***INVALID NUMBER***\n");
                break;
        }
        printf("\n\nDO YOU WANT TO CONTINUE = (1/0) :\n"); // Asking the user if want
to continue or not
        scanf("%d",&num);

    }

}

long int Bin_to_Dec(long int bin)   // function for conversion of binary to decimal
{
    int rem,sum=0,i=0; // declarations
    while(bin!=0)
    {
        rem=bin%10;
        bin=bin/10;
        sum=sum+rem*pow(2,i); // calculating the binary number
        i++;
    }

    printf("\nEquivalent Decimal Number : %d",sum); // printing equivalent decimal
number for entered binary number
}

long int Bin_to_Oct(long int bin)   // function for conversion of binary to octal
{
    int i=0,rem,sum=0,remain[100],len=0; // declarations

    while(bin!=0)
    {
        rem=bin%10;
        bin=bin/10;
        sum=sum+rem*pow(2,i); // calculating binary number
        i++;
    }
    i=0;
    while(sum!=0)
    {
        remain[i]=sum%8; // remainder
        sum=sum/8; // reducing number for next iteration
        i++;
        len++;
    }
    printf("\nEquivalent Octal Number : "); // printing equivalent octal number for entered
binary number
    for(i=len-1;i>=0;i--)
    {
        printf("%d",remain[i]);
    }
}

long int Bin_to_Hex(long int bin)   // function for conversion of binary to hexadecimal
{
    int rem,i=0,sum=0,remain[100],len=0; // declarations
    while(bin!=0)
    {
        rem=bin%10;
        bin=bin/10;
        sum=sum+rem*pow(2,i); // calculating the binary number
        i++;
    }
    i=0;
    while(sum!=0)
    {
        remain[i]=sum%16; // remainder
        sum=sum/16; // reducing number for next iteration
        i++;
        len++;
    }
    printf("\nEquivalent Hexa-Decimal Number : "); // printing equivalent hexadecimal
number for entered binary number
    for(i=len-1;i>=0;i--)
    {
        switch(remain[i])
        {
            case 10:
                printf("A"); break;

            case 11:
                printf("B"); break;

            case 12:
                printf("C"); break;

            case 13:
                printf("D"); break;

            case 14:
                printf("E"); break;

            case 15:
                printf("F"); break;

            default:
```

```
            printf("%d",remain[i]); // default statement
        }

    }
}

long int Dec_to_Bin(long int dec)   // function for conversion of decimal to binary
{
    int rem[50],i,len=0;
    do
    {
        rem[i]=dec%2; // remainder
        dec=dec/2; // reducing number for next iteration
        i++;
        len++;
    }
    while(dec!=0);

    printf("\nEquivalent Binary Number : "); // printing equivalent binary number for
entered decimal number
    for(i=len-1;i>=0;i--)
    {
        printf("%d",rem[i]);
    }
}

long int Dec_to_Oct(long int dec)   // function for conversion of decimal to octal
{
    int rem[50],i,len=0; //declarations
    do
    {
        rem[i]=dec%8; // remainder
        dec=dec/8;
        i++;
        len++;
    }
    while(dec!=0);

    printf("\nEquivalent Octal Number : "); // printing equivalent octal number for entered
decimal number
```

```
    for(i=len-1;i>=0;i--)
    {
        printf("%d",rem[i]);
    }
}

long int Dec_to_Hex(long int dec)   // function for conversion of decimal to hexadecimal
{
    int rem[50],i,len=0;
    do
    {
        rem[i]=dec%16; //remainder
        dec=dec/16;
        i++;
        len++;
    }
    while(dec!=0);

    printf("\nEquivalent Hexa-Decimal Number : "); // printing equivalent hexadecimal
number for  entered decimal number
    for(i=len-1;i>=0;i--)
    {
        switch(rem[i])
        {
            case 10:
                printf("A"); break;

            case 11:
                printf("B"); break;

            case 12:
                printf("C"); break;

            case 13:
                printf("D"); break;

            case 14:
                printf("E"); break;

            case 15:
                printf("F"); break;

            default:
                printf("%d",rem[i]); //default statement
        }

    }
}

long int Oct_to_Bin(long int oct)   // function for conversion of octal to binary
{
    int rem[50],len=0,decimal=0,i=0,num,ans; // declarations

    while(oct!=0)
    {
        ans=oct % 10;
        decimal = decimal + ans * pow(8,i); // calculating decimal
        i++;
        oct = oct/10;
    }

    i=0;
    do
    {
        rem[i]=decimal%2; //remainder
        decimal=decimal/2;
        i++;
        len++;
    }
    while(decimal!=0);

    printf("\nEquivalent Binary Number : "); // printing equivalent binary number for
entered octal number
    for(i=len-1;i>=0;i--)
    {
        printf("%d",rem[i]);
    }
}

long int Oct_to_Dec(long int oct)   // function for conversion of octal to decimal
```

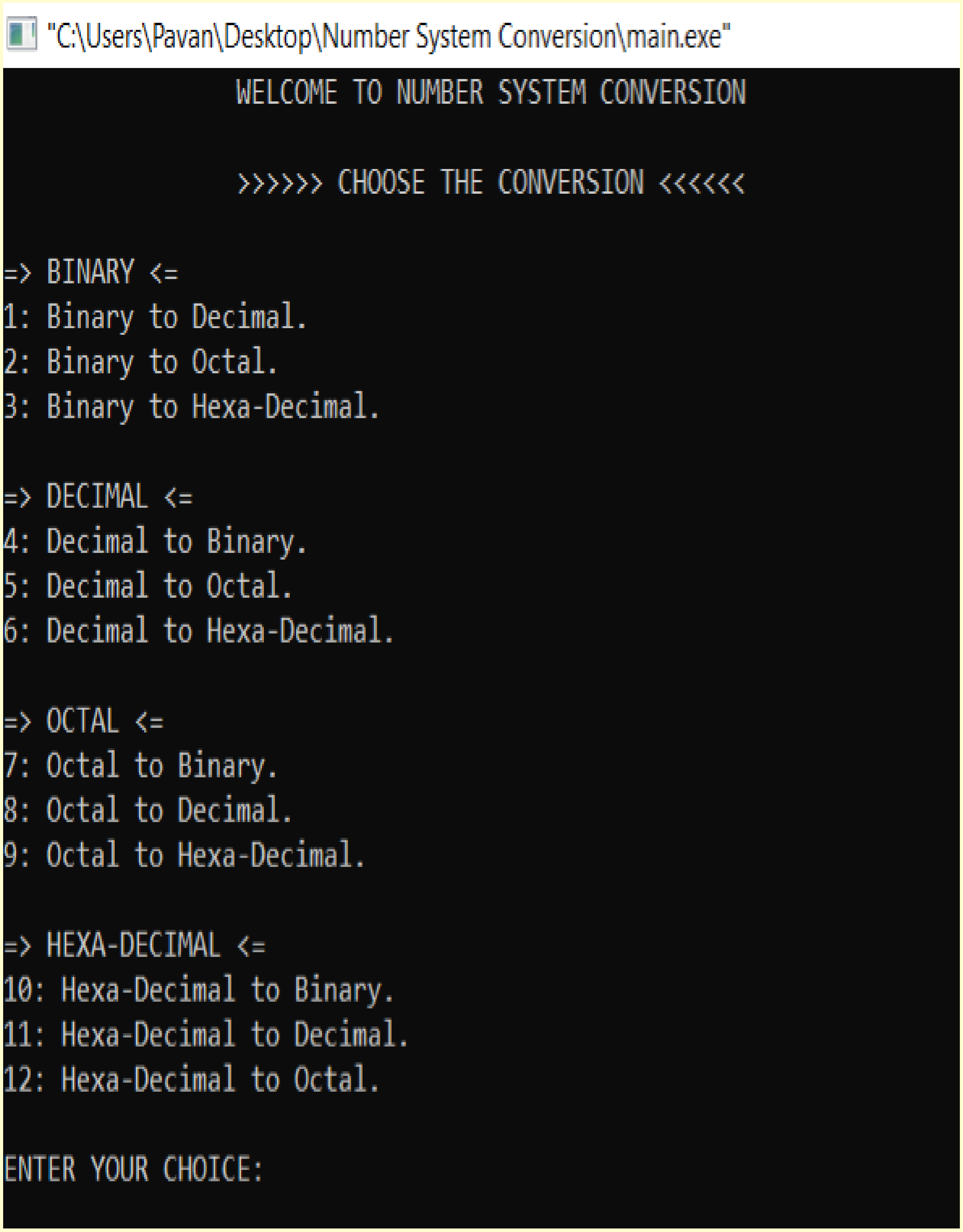
```
{
    int decimal=0,i=0,num,ans; //declarations

    while(oct!=0)
    {
        ans=oct % 10; //remainder
        decimal = decimal + ans * pow(8,i); // calculating the decimal number
        i++;
        oct = oct/10; // reducing the number for next iteration
    }
    printf("\nEquivalent Decimal Number : %d",decimal); // printing equivalent decimal
number of entered octal number
}

long int Oct_to_Hex(long int oct)   // function for conversion of octal to hexadecimal
{
    int rem[50],len=0,decimal=0,i=0,num,ans=0; // declarations
    while(oct!=0)
    {
        ans=oct % 10;
        decimal = decimal + ans * pow(8,i); // calculating decimal number
        i++;
        oct = oct/10;
    }
    i=0;
    while(decimal!=0)
    {
        rem[i]=decimal%16; // remainder
        decimal=decimal/16; // reducing the number by dividing it by 16 for next iteration
        i++;
        len++;
    }
    printf("\nEquivalent Hexa-Decimal Number : "); // printing equivalent hexadecimal
number of entered octal number
    for(i=len-1;i>=0;i--)
    {
        switch(rem[i])
        {
            case 10:
                printf("A"); break;
            case 11:
                printf("B"); break;

            case 12:
                printf("C"); break;

            case 13:
                printf("D"); break;

            case 14:
                printf("E"); break;

            case 15:
                printf("F"); break;

            default:
                printf("%d",rem[i]); // default statement
        }

    }
}

void Hex_to_Bin(char hex[])   // function for conversion of hexadecimal to binary
{
    int i=0;
    printf("\nEquivalent Binary Number : "); // printing equivalent binary number for
entered hexadecimal number
    for(i=0;i<strlen(hex);i++)
    {
        switch (hex[i])
        {
        case '0':
            printf("0000"); break;
        case '1':
            printf("0001"); break;
        case '2':
            printf("0010"); break;
        case '3':
            printf("0011"); break;
        case '4':
            printf("0100"); break;
        case '5':
            printf("0101"); break;
        case '6':
            printf("0110"); break;
        case '7':
            printf("0111"); break;
        case '8':
            printf("1000"); break;
        case '9':
            printf("1001"); break;
        case 'A':
        case 'a':
            printf("1010"); break;
        case 'B':
        case 'b':
            printf("1011"); break;
        case 'C':
        case 'c':
            printf("1100"); break;
        case 'D':
        case 'd':
            printf("1101"); break;
        case 'E':
        case 'e':
            printf("1110"); break;
        case 'F':
        case 'f':
            printf("1111"); break;

        default:
            printf("\n Invalid hexa digit %c ", hex[i]);  // Default statement
        }
    }

}

void Hex_to_Dec(char hex[])   // function for conversion of hexadecimal to decimal
{
    int i,num=0,power=0,decimal=0;  // declarations

    for(i=strlen(hex)-1;i>=0;i--)
    {
        if(hex[i]=='A'||hex[i]=='a')
        {
            num=10;
        }
        else if(hex[i]=='B'||hex[i]=='b')
        {
            num=11;
        }
        else if(hex[i]=='C'||hex[i]=='c')
        {
            num=12;
        }
        else if(hex[i]=='D'||hex[i]=='d')
        {
            num=13;
        }
        else if(hex[i]=='E'||hex[i]=='e')
        {
            num=14;
        }
        else if(hex[i]=='F'||hex[i]=='f')
        {
            num=15;
        }
        else
        //(a[i]>=0 || a[i]<=9)
        {
            num=hex[i]-48;
        }

        decimal=decimal+num*pow(16,power);  // calculating the decimal number
        power++;
    }
    printf("\nEquivalent Decimal Number : %d",decimal); // Printing equivalent decimal
number of entered hexadecimal number
```

```
}

void Hex_to_Oct(char hex[])   // function for conversion of hexadecimal to octal
{
   int i,len,num=0,power=0,decimal=0,rem[100];  // declarations

   for(i=strlen(hex)-1;i>=0;i--)
   {
      if(hex[i]=='A'||hex[i]=='a')
      {
         num=10;
      }
      else if(hex[i]=='B'||hex[i]=='b')
      {
         num=11;
      }
      else if(hex[i]=='C'||hex[i]=='c')
      {
         num=12;
      }
      else if(hex[i]=='D'||hex[i]=='d')
      {
         num=13;
      }
      else if(hex[i]=='E'||hex[i]=='e')
      {
         num=14;
      }
      else if(hex[i]=='F'||hex[i]=='f')
      {
         num=15;
      }
      else
      //(a[i]>=0 || a[i]<=9)
      {
         num=hex[i]-48;
      }

      decimal=decimal+num*pow(16,power); // calculating the decimal number
      power++;
```

```
    }

    i=0,len=0;
    while(decimal!=0)
    {
        rem[i]=decimal%8;  // remainder
        decimal=decimal/8;   // reducing the decimal number by dividing it by 8 for next
iteration
        i++;
        len++;
    }
    printf("\nEquivalent Octal Number : ");    // Printing the equivalent octal number of
entered hexadecimal number
    for(i=len-1;i>=0;i--)
    {
        printf("%d",rem[i]);
    }

}
```

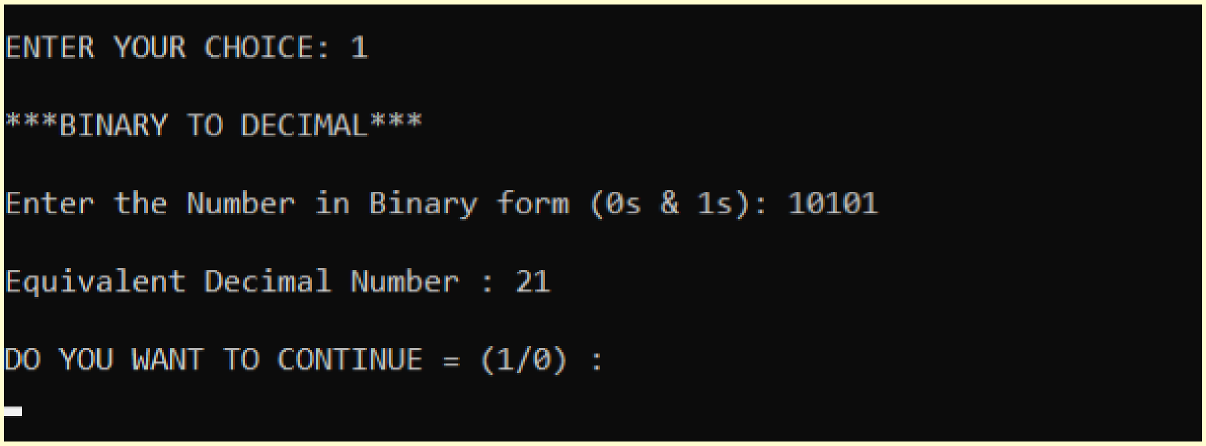
Chapter 3. Results and Discussions

3.1. Results



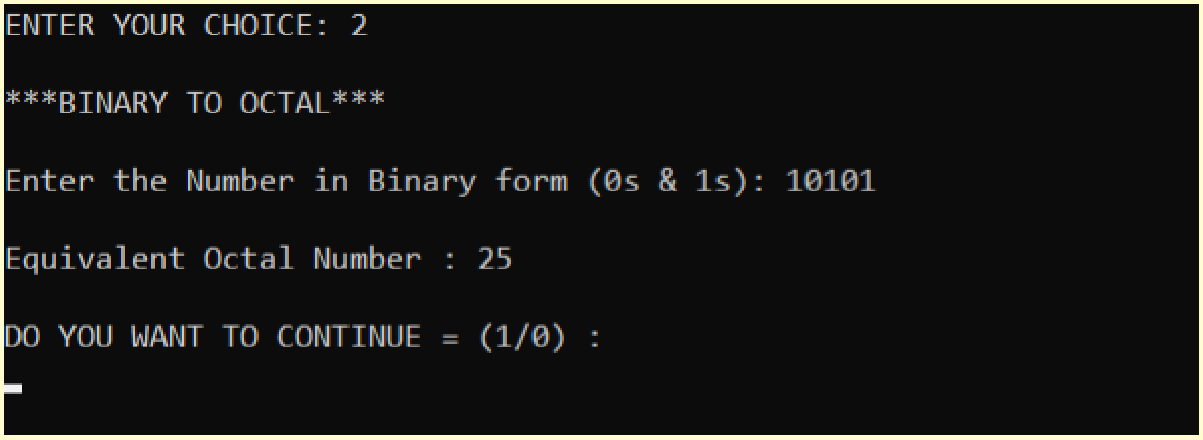
```
"C:\Users\Pavan\Desktop\Number System Conversion\main.exe"

                WELCOME TO NUMBER SYSTEM CONVERSION

                >>>>>> CHOOSE THE CONVERSION <<<<<<

=> BINARY <=
1: Binary to Decimal.
2: Binary to Octal.
3: Binary to Hexa-Decimal.

=> DECIMAL <=
4: Decimal to Binary.
5: Decimal to Octal.
6: Decimal to Hexa-Decimal.

=> OCTAL <=
7: Octal to Binary.
8: Octal to Decimal.
9: Octal to Hexa-Decimal.

=> HEXA-DECIMAL <=
10: Hexa-Decimal to Binary.
11: Hexa-Decimal to Decimal.
12: Hexa-Decimal to Octal.

ENTER YOUR CHOICE:
```

Figure 3.1.1. Application Launch Screen



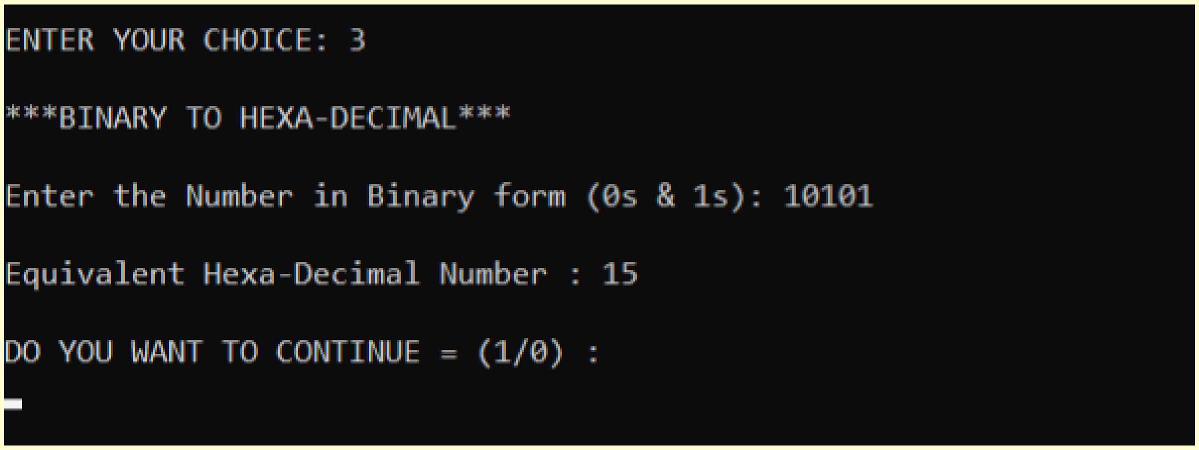
```
ENTER YOUR CHOICE: 1

***BINARY TO DECIMAL***

Enter the Number in Binary form (0s & 1s): 10101

Equivalent Decimal Number : 21

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.2. Choice (1) Binary to Decimal



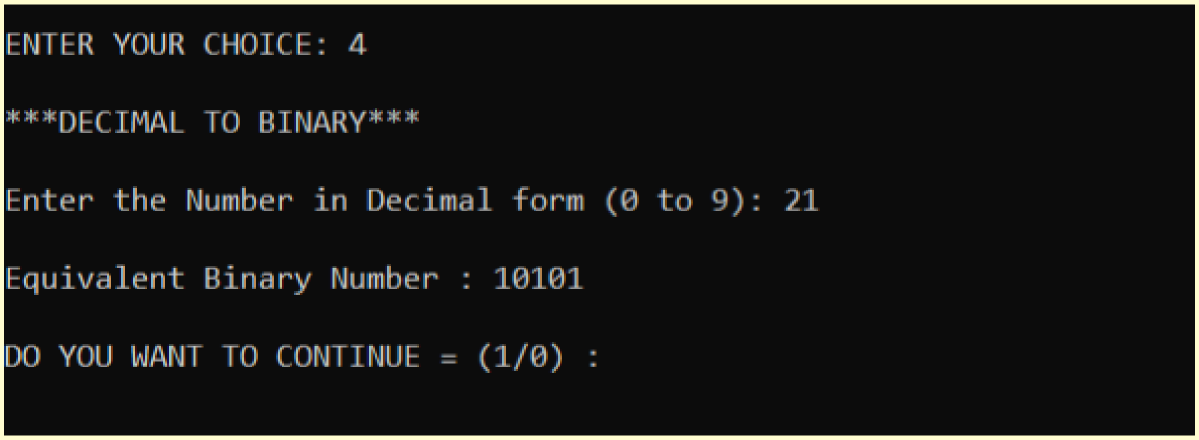
```
ENTER YOUR CHOICE: 2

***BINARY TO OCTAL***

Enter the Number in Binary form (0s & 1s): 10101

Equivalent Octal Number : 25

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.3. Choice (2) Binary to Octal



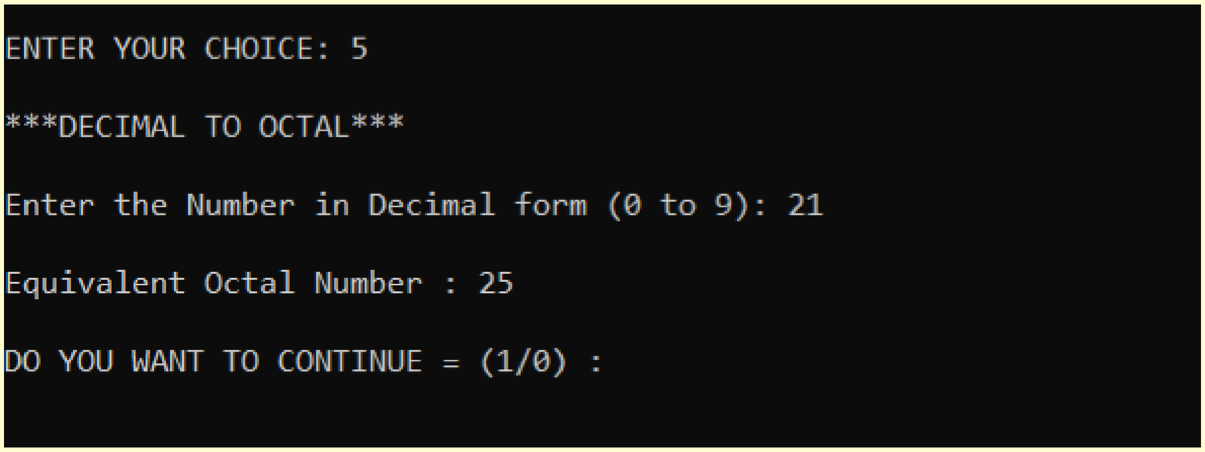
```
ENTER YOUR CHOICE: 3

***BINARY TO HEXA-DECIMAL***

Enter the Number in Binary form (0s & 1s): 10101

Equivalent Hexa-Decimal Number : 15

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.4. Choice (3) Binary to Hexadecimal



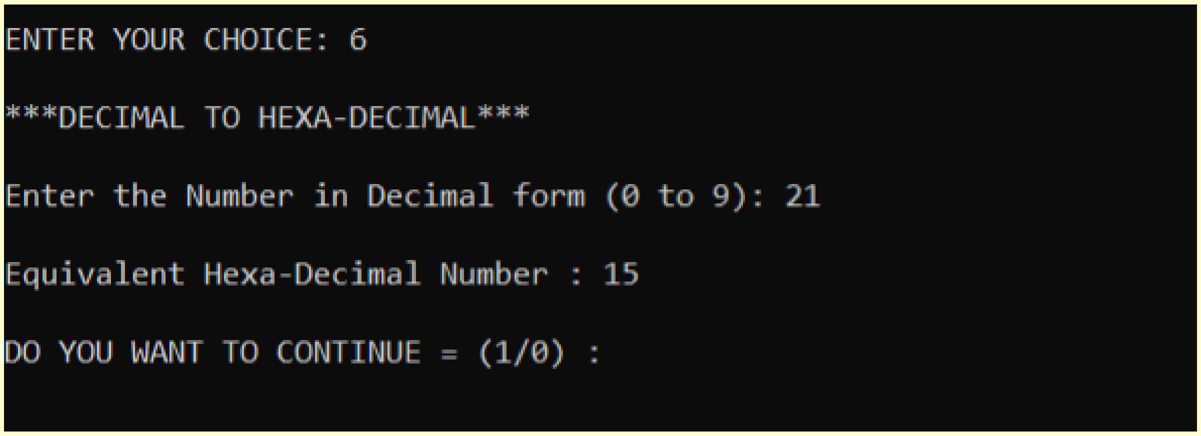
```
ENTER YOUR CHOICE: 4

***DECIMAL TO BINARY***

Enter the Number in Decimal form (0 to 9): 21

Equivalent Binary Number : 10101

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.5. Choice (4) Decimal to Binary



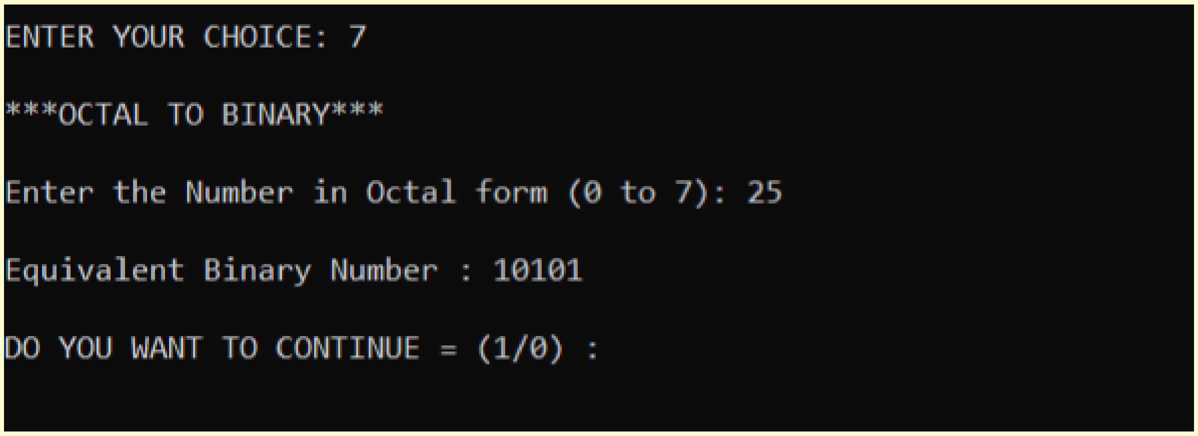
```
ENTER YOUR CHOICE: 5

***DECIMAL TO OCTAL***

Enter the Number in Decimal form (0 to 9): 21

Equivalent Octal Number : 25

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.6. Choice (5) Decimal to Octal



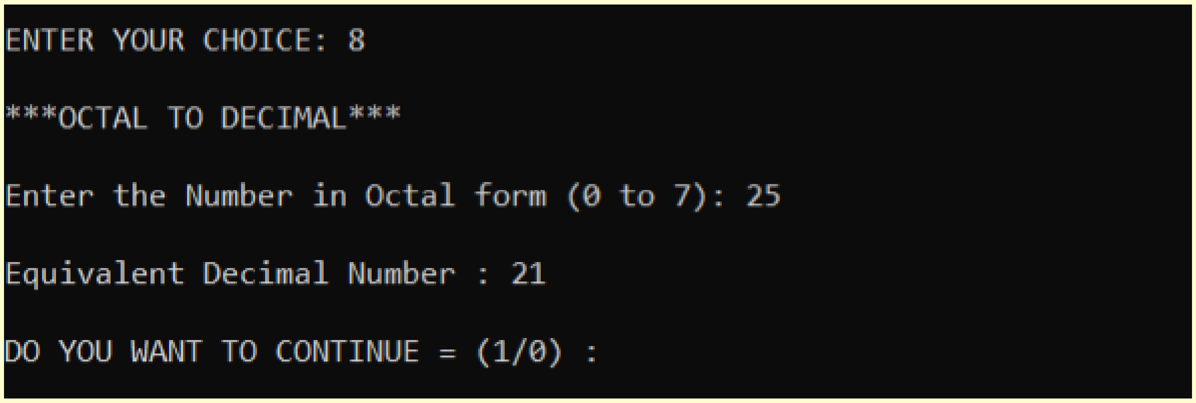
```
ENTER YOUR CHOICE: 6

***DECIMAL TO HEXA-DECIMAL***

Enter the Number in Decimal form (0 to 9): 21

Equivalent Hexa-Decimal Number : 15

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.7. Choice (6) Decimal to Hexadecimal



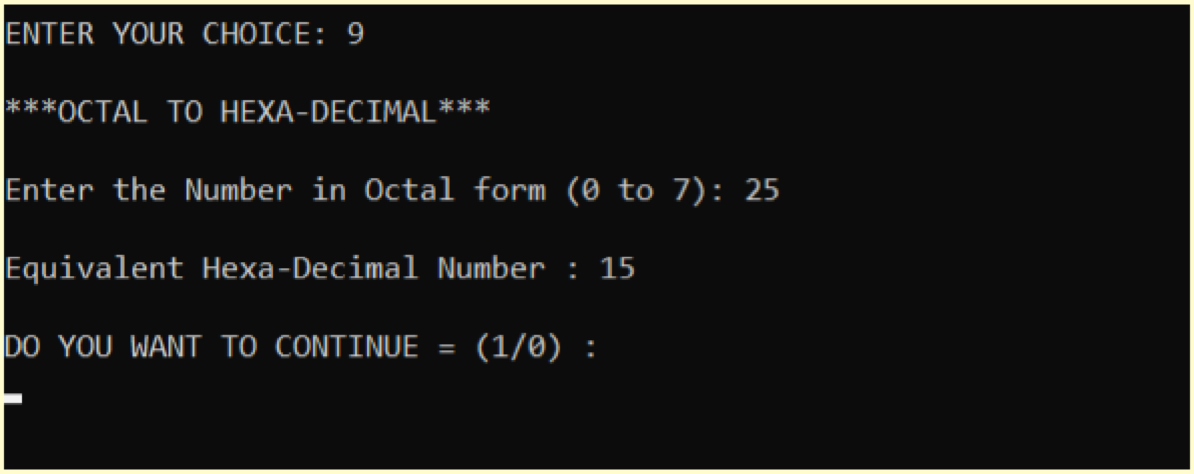
```
ENTER YOUR CHOICE: 7

***OCTAL TO BINARY***

Enter the Number in Octal form (0 to 7): 25

Equivalent Binary Number : 10101

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.8. Choice (7) Octal to Binary



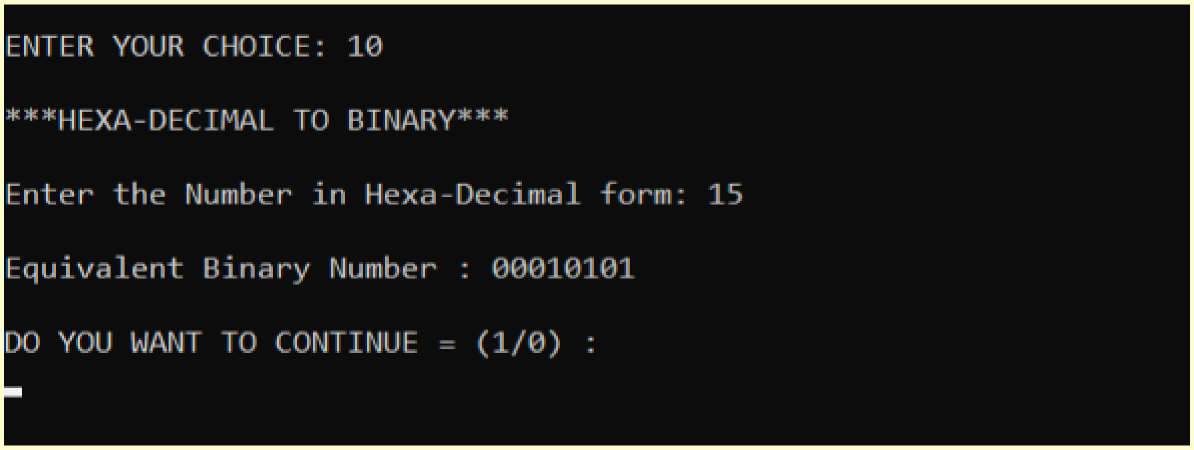
```
ENTER YOUR CHOICE: 8

***OCTAL TO DECIMAL***

Enter the Number in Octal form (0 to 7): 25

Equivalent Decimal Number : 21

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.9. Choice (8) Octal to Decimal



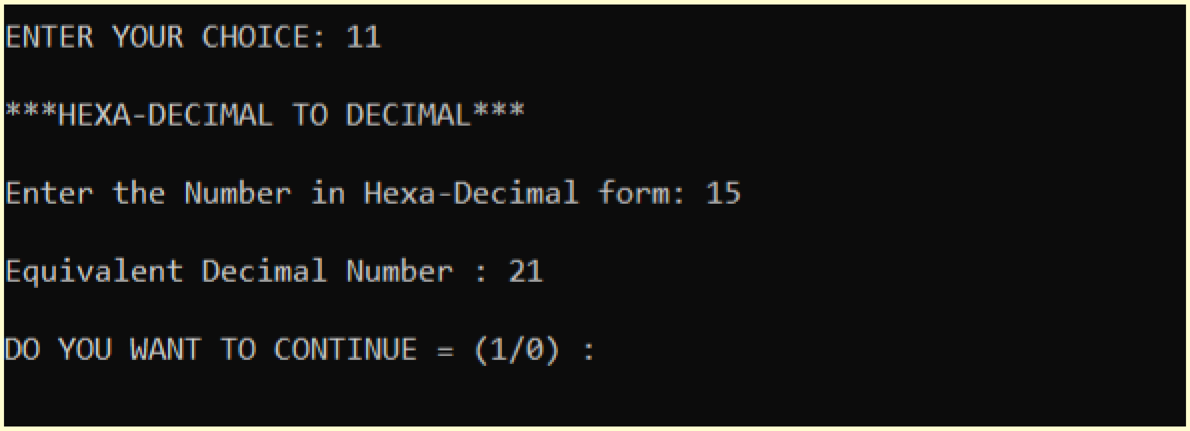
```
ENTER YOUR CHOICE: 9

***OCTAL TO HEXA-DECIMAL***

Enter the Number in Octal form (0 to 7): 25

Equivalent Hexa-Decimal Number : 15

DO YOU WANT TO CONTINUE = (1/0) :
_
```

Figure 3.1.10. Choice (9) Octal to Hexadecimal



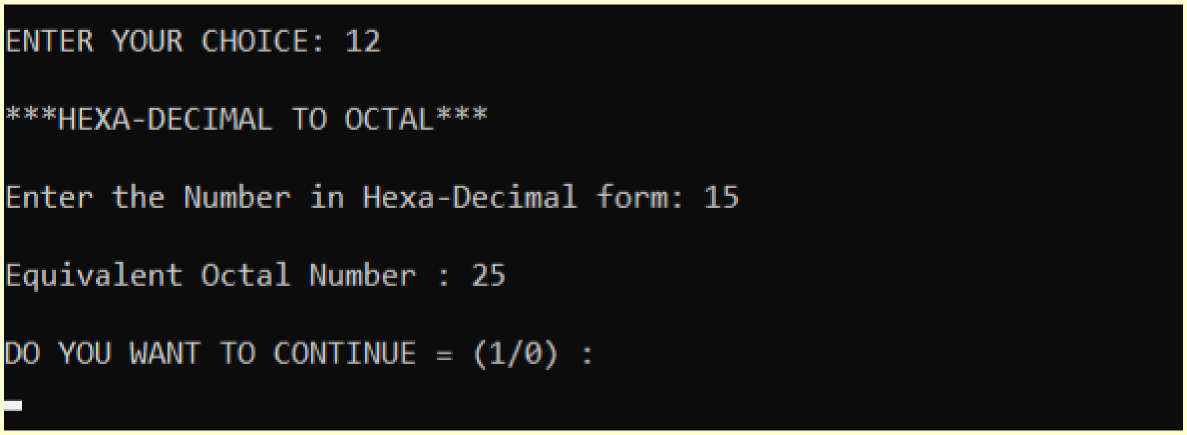
```
ENTER YOUR CHOICE: 10

***HEXA-DECIMAL TO BINARY***

Enter the Number in Hexa-Decimal form: 15

Equivalent Binary Number : 00010101

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.11. Choice (10) Hexadecimal to Binary



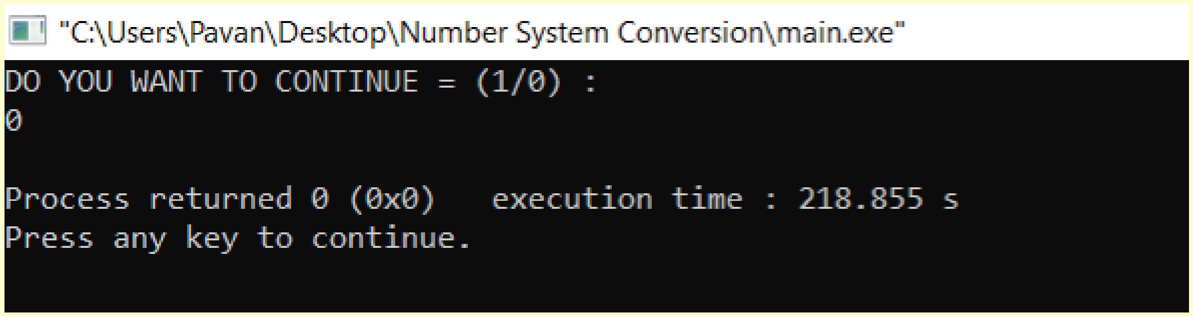
```
ENTER YOUR CHOICE: 11

***HEXA-DECIMAL TO DECIMAL***

Enter the Number in Hexa-Decimal form: 15

Equivalent Decimal Number : 21

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.12. Choice (11) Hexadecimal to Decimal



```
ENTER YOUR CHOICE: 12

***HEXA-DECIMAL TO OCTAL***

Enter the Number in Hexa-Decimal form: 15

Equivalent Octal Number : 25

DO YOU WANT TO CONTINUE = (1/0) :
```

Figure 3.1.13. Choice (12) Hexadecimal to Octal



```
"C:\Users\Pavan\Desktop\Number System Conversion\main.exe"
DO YOU WANT TO CONTINUE = (1/0) :
0

Process returned 0 (0x0)   execution time : 218.855 s
Press any key to continue.
```

Figure 3.1.14. Exiting the application

2.2. Discussion

- The program basically greets the user and then gives the user total 12 choices for conversion which includes 1. Binary to Decimal 2. Binary to Octal 3. Binary to Hexadecimal 4. Decimal to Binary 5. Decimal to Octal 6. Decimal to Hexadecimal 7. Octal to Binary 8. Octal to Decimal 9. Octal to Hexadecimal 10. Hexadecimal to Binary 11. Hexadecimal to Decimal 12. Hexadecimal to Octal
- The user will be asked about their choice. If any numbers from 1 to 12 is typed then the user would be asked about the input of a certain number system which would be checked for correctness. If the number which is given as input has error then it would again ask the user for choices.
- If the entered number is correct then the equivalent number in other number system which is chosen would be printed.
- After the conversion and the number is printed the user would be asked if he/she would like to continue further or quit the program by typing 0 for quit and 1 for continue.
- If 0 is entered then the program will be quitted and if 1 is entered then the program will be continued further and the user will be again asked about the choice which he/she wants the conversion to be made.
- Hence, the program allows the interconversion of numbers among binary, decimal, octal and hexadecimal as well as it checks for the correctness and also the continuity of the program.

Chapter 4. Conclusions

- This project is able to interconvert all the numbers among number systems from binary, decimal, octal and hexadecimal.
- This program gives the user all the 12 choices which was discussed earlier.
- This program is also fast and efficient.
- It also checks the correctness of the number entered by the user and provides the feedback if there is an error or will carry forward if the number is correct.
- It gives the user a sense of continuity by allowing to use the program again by entering 0 or 1 and this process will reduce the time to open the project again and again for the conversions and will an efficient way.
- This program can be improved by adding graphics to it and by improving the interface.
- It can also be improved by adding the functionality to convert decimal such as 12.1 into the different numbers as this program can only convert whole numbers or integers.
- It can be improving the algorithm of the program for more effectiveness and by using more functions which are repeated more times inside another function as there are some of the conversions which uses same blocks of code.
- The project can be implemented in calculators specially for the made for the conversions also which will be helpful in some ways.
- This project is an overall effort to get the concepts clear about different number systems as well as it would be helpful in IT fields as well as for students who have the concepts of number system in their curriculum as a practical way and also as a help to identify the answers.
- There are more improvements like interface, use of reusable blocks of code as function which can be incorporated in the future versions.
- It is a good learning for newbies who wants to brush the skills in programming as this project also uses many entities of C programming language and it is also a vast project which will help to understand how to work on large scale programs also.

Chapter 5: Appendices

Appendix 1

Table 5.1. Equivalent numbers in different number systems

Binary	Decimal	Octal	Hexadecimal
0	0	0	00
1	1	1	01
10	2	2	02
11	3	3	03
100	4	4	04
101	5	5	05
110	6	6	06
111	7	7	07
1000	8	10	08
1001	9	11	09
1010	10	12	0A
1011	11	13	0B
1100	12	14	0C
1101	13	15	0D
1110	14	16	0D
1111	15	17	0E
10000	16	20	0F
10001	17	21	10
10010	18	22	11
10011	19	23	12
10100	20	24	13

Appendix 2

Table 5.2. Abbreviations which are used in Computer Science

AI	Artificial Intelligence
ALU	Arithmetic Logic Unit
ASCII	American Standard Code for Information Interchange
CAD	Computer Aided Design
DBMS	Data Base Management System
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
JPEG	Joint Photographic Experts Group
MPEG	Moving Pictures Experts Group
NIC	Network Interface Card
OCR	Optical Character Recognition
OSS	Open Source Software
PDF	Portable Document Format
QoS	Quality of Service
SDLC	Software Development Life Cycle
SQL	Structured Query Language
TCP	Transport Control Protocol
URL	Uniform Resource Locator
UTF	Unicode Transformation Format

REFERENCES

[1]. Yashavant Kanetkar, Let Us C – 5th Edition, BPB Publications, 2004, p 49-354.

[2]. Brian W. Kernighan and Dennis Ritchie, The C Programming Language, Prentice Hall Software Series, USA, p 5-88.

[3]. Mala Saraswathy Nataraj & Michael O.J. Thomas, “Developing Understanding of Number System Structure from the History of Mathematics”, Mathematics Education Research Journal, 2009, p 96-115.

[4]. D.P. Kothari and I.J. Nagrath, Basic Electronics – 2nd Edition, McGraw Hill Education(India) Private Limited, 10.2.