

GOMENDRA MULTIPLE COLLEGE

Birtamode-04, Jhapa

Affiliated to Purbanchal University



Program: BCA- IT

Semester: 1st

Faculty: Science and Technology

Project report on

NUMBERING SYSTEM CALCULATOR

Submitted by:

- Kushal Bhandari
- Bishal Magar
- Ramjit Soren

Under the supervision of

Assistant Lecturer

Mr. Nabin Prasain

ACKNOWLEDGEMENT

I would like to sincerely express my gratitude to all those who played a pivotal role in the successful realization of the "Numbering System Calculator" project. This project would not have been possible without the support, guidance, and cooperation of several individuals and resources.

Foremost, I extend my heartfelt appreciation to Mr. Nabin Prasain(Assistant Lecturer) and Dr. Rupak Khanal (College Chief) for their invaluable mentorship throughout this project. Their expertise, constant encouragement, and constructive feedback were instrumental in shaping the development of this project.

I would also like to acknowledge the support and resources provided by Gomendra Multiple Campus. The access to facilities, libraries, and the academic environment was crucial to the project's execution. The guidance provided by college have been invaluable in shaping my understanding of programming concepts and enhancing my skills in C programming. I am genuinely thankful for the opportunities and encouragement that the college has provided.

Lastly, I extend my gratitude to my dedicated fellow team members, Bishal Magar and Ramjit Soren, who worked tirelessly as a cohesive team, sharing ideas, troubleshooting issues, and collectively contributing to the project's success. Each member's distinct skills and unwavering commitment greatly enhanced the quality of our project.

In conclusion, I want to express my deepest appreciation to everyone involved in this project, as it would not have been possible without your support and contributions.

Contents

Topics	Page No.
1. Introduction of Project	
➤ Key Features	
2. Objectives	
➤ General Objectives	
➤ Specific Objectives	
3. System Flowchart	
4. Importance of Number System Calculator	
5. System Requirements	
6. Source Code	
7. Output	
8. Abstract	
9. Future Use and Implementation	
10. Conclusion	
11. Bibliography	

Introduction of Project

A numbering system calculator is a powerful tool that enables users to convert numbers between different numerical systems, such as binary, decimal, octal, and hexadecimal. These numerical systems play a crucial role in various fields, including computer science, digital electronics, and mathematics.

This program is a Number System Converter implemented in the C programming language. It provides a user-friendly menu-driven interface for converting numbers between various number systems, including Binary, Decimal, Octal, and Hexadecimal.

It presents a menu-driven interface and includes functions for each type of conversion. Users can select the conversion they want to perform from the menu, input the required values, and obtain the converted result. This tool is particularly useful for programmers, students, or anyone working with different numeral systems in their projects or studies.

Key Features:

Here are the key features of the code:

- **User-Friendly Menu:** The program presents a user-friendly menu that allows the user to select various conversion options. Users can choose from converting between binary, decimal, octal, and hexadecimal numeral systems.
- **Clear Console:** The code includes a '**clearConsole**' function to clear the console screen, providing a cleaner interface for the user.
- **User Interaction:** After each conversion, the program asks the user if they want to continue with another conversion or quit. It also validates the user's input.
- **Clean Code Structure:** The code is organized with functions for each conversion, improving readability and maintainability.
- **Looping:** The code uses loops to allow the user to perform multiple conversions in a single run of the program.
- **Graceful Exit:** The program provides a way for users to exit gracefully with a "Thank you for opening calculator!" message.

Objectives

A numbering system calculator is a tool or software application designed to perform various operations on different numbering systems, such as decimal, binary, octal, and hexadecimal. This system aims to facilitate conversions between different numbering systems, perform mathematical operations in various bases, and provide a user-friendly and efficient tool for working with numbers represented in different bases. In digital design and programming, numbering system calculators can be used for debugging and verifying calculations and representations in different numbering systems to ensure the correctness of algorithms and hardware designs.

The objectives of this system can be divided into two categories, which are being explained as follow:

General Objectives:

1. To create a software for converting numbers between different numeral systems

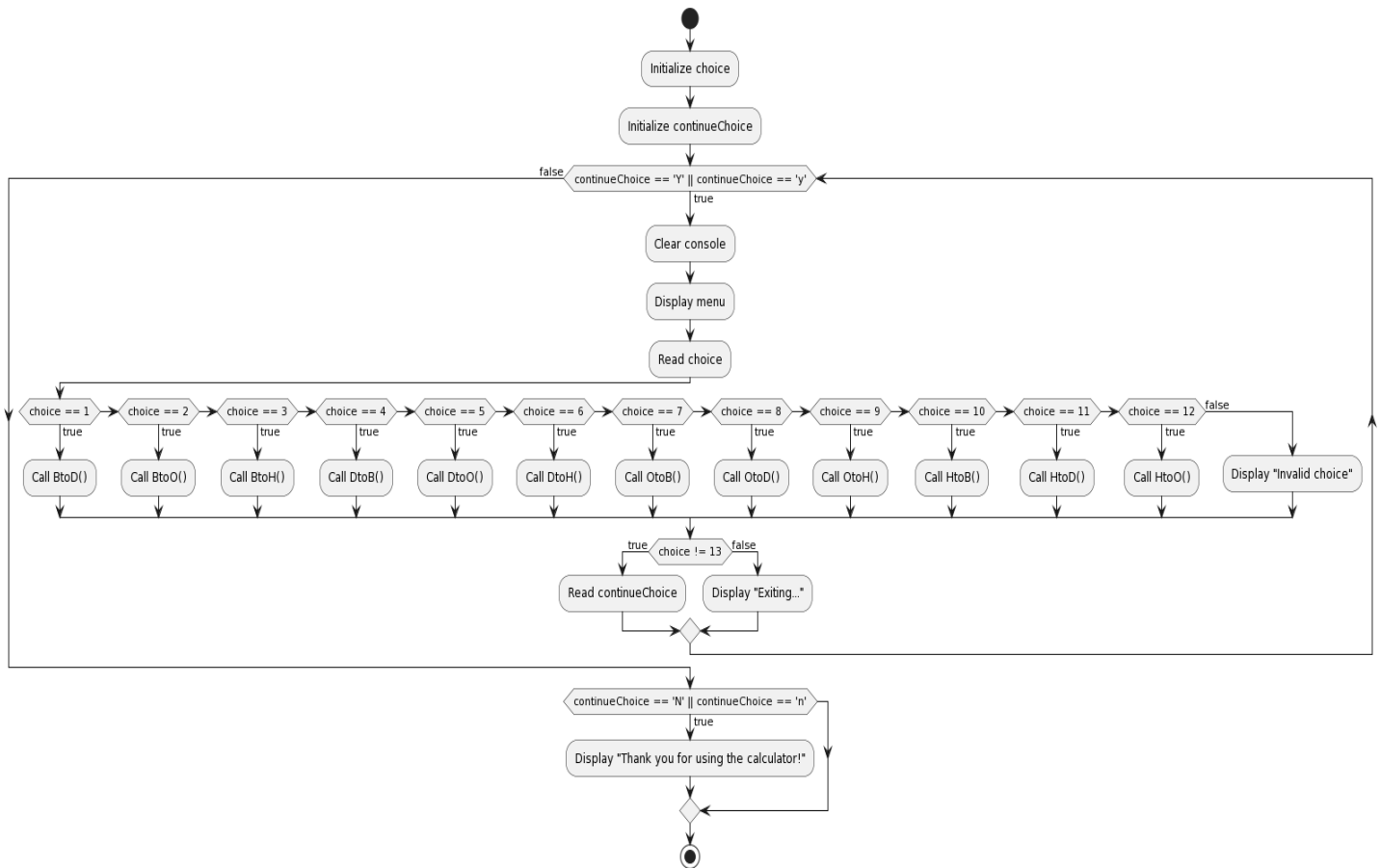
Specific Objectives:

1. To handle integer numbers in various numeral systems.
2. To develop a versatile number system software.
3. To provides accurate and precise results.
4. To serve as an educational tools.
5. To offer a user-friendly interface with intuitive input and output option's.
6. To allow continuous use of the system as long as the user requires it.

Importance of Number System Calculator

The numbering system calculator can convert numbers between different numbering systems, including binary, decimal, octal, and hexadecimal. A numbering system calculator is a valuable tool for both learning and practical purposes, especially in fields related to computer science and engineering. It simplifies complex conversions and facilitates a better understanding of numbering systems. For individuals learning or practicing programming, especially in low-level languages like C or assembly language, understanding and manipulating different numbering systems is fundamental. This calculator can help them practice and understand these concepts better.

Flow Chart



Source Code

```
#include <stdio.h>

#include <math.h>

#include <string.h>

#include <stdlib.h> // Added for system() function


#define MAX_HISTORY_SIZE 3


// Function prototypes
void BtoD();
void BtoO();
void BtoH();
void DtoB();
void DtoO();
void DtoH();
void OtoB();
void OtoD();
void OtoH();
void HtoB();
void HtoD();
void HtoO();
void options();


void clearConsole()
{
    // Clear console screen (system-specific commands)
    #ifdef _WIN32
```



```
system("cls");
#else
system("clear");
#endif
}
void options()
{
    int choice;
    char continueChoice;
    do
    {
        clearConsole(); // Clear the console before displaying the menu
        printf("\t\t*****NUMBERING SYSTEM CALCULATOR*****\n");

        printf("\n\nYour options are:\n\n");
        printf("1. Binary to Decimal\n");
        printf("2. Binary to Octal\n");
        printf("3. Binary to Hexadecimal\n");
        printf("4. Decimal to Binary\n");
        printf("5. Decimal to Octal\n");
        printf("6. Decimal to Hexadecimal\n");
        printf("7. Octal to Binary\n");
        printf("8. Octal to Decimal\n");
        printf("9. Octal to Hexadecimal\n");
        printf("10. Hexadecimal to Binary\n");
        printf("11. Hexadecimal to Decimal\n");
        printf("12. Hexadecimal to Octal\n");
        printf("13. Quit\n"); // Added option to quit the program
        printf("\nEnter your choice: ");
        scanf("%d", &choice);
```

```
switch (choice)
{
case 1:
    printf("\nSelected choice: Binary to Decimal\n");
    BtoD();
    break;
case 2:
    printf("\nSelected choice: Binary to Octal\n");
    BtoO();
    break;
case 3:
    printf("\nSelected choice: Binary to Hexadecimal\n");
    BtoH();
    break;
case 4:
    printf("\nSelected choice: Decimal to Binary\n");
    DtoB();
    break;
case 5:
    printf("\nSelected choice: Decimal to Octal\n");
    DtoO();
    break;
case 6:
    printf("\nSelected choice: Decimal to Hexadecimal\n");
    DtoH();
    break;
case 7:
    printf("\nSelected choice: Octal to Binary\n");
    OtoB();
```

```

        break;
case 8:
    printf("\nSelected choice: Octal to Decimal\n");
    OtoD();
    break;
case 9:
    printf("\nSelected choice: Octal to Hexadecimal\n");
    OtoH();
    break;
case 10:
    printf("\nSelected choice: Hexadecimal to Binary\n");
    HtoB();
    break;
case 11:
    printf("\nSelected choice: Hexadecimal to Decimal\n");
    HtoD();
    break;
case 12:
    printf("\nSelected choice: Hexadecimal to Octal\n");
    HtoO();
    break;
case 13:
    printf("\n\t\t*****Exiting.....*****");
    break;
default:
    printf("Invalid choice\n");
    break;
}

if (choice != 13 && choice != 14) {

```

```

        printf("\n\nDo you want to continue? (Y/N): ");
        scanf(" %c", &continueChoice);
    }

}

while (choice != 14 && (continueChoice == 'Y' || continueChoice == 'y'));

if(continueChoice == 'N' || continueChoice == 'n')
    printf("\nThank you for using the calculator!!!!\n");
}

void BtoD()
{
    int binary, decimal = 0, base = 1, remainder;

    printf("\nEnter valid binary number: ");
    scanf("%d", &binary);

    while (binary > 0) {
        remainder = binary % 10;
        decimal += remainder * base;
        binary /= 10;
        base *= 2;
    }

    printf("Decimal number is %d\n", decimal);
}

void BtoO()
{
    int binary, octal = 0, remainder, decimal = 0, base = 1;

    printf("\nEnter valid binary number: ");

```

```
scanf("%d", &binary);
```

```
// Check if the input is a valid binary number
```

```
while (binary > 0)
```

```
{
```

```
    remainder = binary % 10;
```

```
    decimal += remainder * base;
```

```
    binary /= 10;
```

```
    base = base * 2;
```

```
}
```

```
base = 1;
```

```
while (decimal > 0)
```

```
{
```

```
    remainder = decimal % 8;
```

```
    octal = octal + (remainder * base);
```

```
    decimal = decimal / 8;
```

```
    base = base * 10;
```

```
}
```

```
printf("Octal number is %d\n", octal);
```

```
}
```

```
void BtoH()
```

```
{
```

```
int decimal=0, remainder, i=0, j=0, temp, binary;
```

```
char hexa[100];
```

```
printf("\nEnter valid binary number: ");
```

```
scanf("%d", &binary);
```

```
// first convert to decimal and decimal to hexadecimal
```

```

temp=binary;
while (temp>0)
{
    remainder=temp % 10;
    decimal=decimal+remainder * pow(2,i);
    i++;
    temp=temp/10;

}
printf("Decimal value is %d\n", decimal);
i=0;
remainder=0;
while (decimal > 0)
{
    remainder=decimal%16;
    if(remainder<10)
    {
        hexa[i++]=remainder+48;
    }
    else
    {
        hexa[i++]=remainder + 55;
    }
    decimal=decimal/16;
}
printf("Hexadecimal value: ");
for(j=i-1; j>=0; j--)
{
    printf("%c",hexa[j]);
}

```

```
}
```

```
void DtoB()
```

```
{
```

```
    int Decnum, rem, Binnum = 0, Base = 1;
```

```
    printf("\nEnter valid decimal number:");
```

```
    scanf("%d", &Decnum);
```

```
    while (Decnum > 0)
```

```
    {
```

```
        rem = Decnum % 2;
```

```
        Binnum = Binnum + (rem * Base);
```

```
        Decnum = Decnum / 2;
```

```
        Base = Base * 10;
```

```
    }
```

```
    printf("Binary value is %d\n", Binnum);
```

```
}
```

```
void DtoO()
```

```
{
```

```
    int decimal, octal=0, reminder, base=1;
```

```
    printf("\nEnter valid decimal number: ");
```

```
    scanf("%d", &decimal);
```

```
    while(decimal > 0)
```

```
    {
```

```
        reminder = decimal % 8;
```

```
        octal = octal + reminder * base;
```

```
        base = base * 10;
```

```
        decimal = decimal / 10;
```

```
    }
```

```

    printf("Octal Value is: %d\n", octal);
}

void DtoH()
{
    int decimal, reminder, i=0;
    char hexadecimal[100];
    printf("\nEnter valid decimal number: ");
    scanf("%d", &decimal);
    while(decimal > 0)
    {
        reminder = decimal % 16;
        if(reminder<10)
        {
            hexadecimal[i++]=48 + reminder;
        }
        else
        {
            hexadecimal[i++]=55 + reminder;
        }
        decimal = decimal / 16;
    }
    for(i=i-1; i>=0; i--)
        printf("Hexadecimal value is %s\n",hexadecimal);
}

```

```

void OtoB()
{
    long binary=0;
    int octal, decimal=0, remainder=0, base=0;

```



```

printf("\nEnter valid octal number: ");
scanf("%d", &octal);
int tempOctal = octal;
while (tempOctal != 0) {
    if (tempOctal % 10 < 0 || tempOctal % 10 > 7) {
        printf("Not an octal number\n");
        return ; // Exit with an error code
    }
    tempOctal /= 10;
}
while(octal != 0)
{
    remainder = octal % 10;
    decimal += remainder * pow(8 , base);
    octal /= 10;
    base++;
}
base=1;
remainder=0;

while(decimal)
{
    remainder = decimal % 2;
    binary += remainder * base;
    decimal = decimal / 2;
    base = base * 10;
}
printf("Binary value is %ld\n", binary);

```

```
}
```

```
void OtoD()
```

```
{
```

```
    int octal, decimal=0, reminder=0, base=0;
```

```
    printf("\nEnter valid octal number: ");
```

```
    scanf("%d", &octal);
```

```
    while(octal > 0)
```

```
    {
```

```
        reminder = octal % 10;
```

```
        decimal = decimal + reminder * pow(8 , base);
```

```
        octal = octal / 10;
```

```
        base++;
```

```
    }
```

```
    printf("Decimal value is %d\n", decimal);
```

```
}
```

```
void OtoH()
```

```
{
```

```
    int octal;
```

```
    printf("\nEnter valid octal number: ");
```

```
    scanf("%o", &octal);
```

```
    printf("Hexadecimal value is %x\n", octal);
```

```
}
```

```
void HtoB()
```

```
{
```

```
char hexa[1000];
char binary[1000];
int base = 0;
printf("\nEnter valid hexadecimal number: ");
scanf("%s", hexa);
while(hexa[base])
{
    switch(hexa[base])
    {
        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("Invalid hexadecimal value");
        }
        base++;
    }
}

```

```

void HtoD()
{
    char hex[100];
    int decimal = 0, base = 1, len, i;
    printf("\nEnter valid hexadecimal number: ");
    scanf("%s", hex);
    len=strlen(hex);
    for(i=len-1; i>=0; i--)
    {
        if(hex[i] >= '0' && hex[i] <= '9')
        {
            decimal += (hex[i] - 48) * base;
            base *= 16;
        }
        else if(hex[i] >= 'A' && hex[i] <= 'F')
        {
            decimal += (hex[i] - 55) * base;
            base *= 16;
        }
        else if(hex[i] >= 'a' && hex[i] <= 'f')

```

```
{
    decimal += (hex[i] - 87) * base;
    base *= 16;
}
}

printf("Decimal value is %d\n", decimal);
}
```

void HtoO()

```
{
    int hexa;
    printf("\nEnter valid hexadecimal number: ");
    scanf("%x",&hexa);
    printf("Octal equivalent of number is %o\n", hexa);
}
```

int main()

```
{
    options();
    return 0;
}
```

Output

1. List of Options:

```
Your options are:  
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  
13. Quit
```

2. Insert User Choice:

```
Enter your choice: 1
```

3. Selected Choice:

```
Selected choice: Binary to Decimal
```

4. Insert Input:

```
Enter valid binary number: 1010
```

5. Display Output:

```
Decimal number is 10
```

6. Permission for Another Calculation (If Yes):

```
Do you want to continue? (Y/N): Y_
```

7. Insert Another Choice:

```
Enter your choice: 13
```

8. Exiting the program:

```
*****Exiting.....*****
```

9. Permission for Further Operation (If No):

```
Do you want to continue? (Y/N): N
```

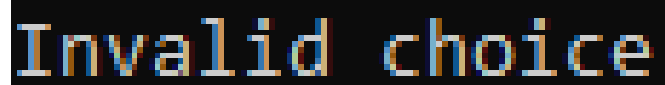
10. Display Output:

```
Thank you for using the calculator!!!!
```

11. Invalid Choice:

```
Enter your choice: 15
```


12. Display Output for Invalid Input Choice:



```
Invalid choice
```

System Requirements

Minimum System Requirements

1. Operating System: Windows, Linux, or MacOS.
2. Processor: Dual-core AMD or Intel processor with a clock speed of 3 GHz or higher.
3. Memory (RAM): 2GB of RAM or more.
4. Storage: At least 64 GB of available HDD/SSD space.
5. Input Devices: PC with a compatible monitor, mouse, and standard keyboard.
6. Software Dependencies: Compilers such as Code Blocks, Turbo C++, and Dev C++.

The Numbering System Calculator project should run without major problems if user system meets these minimum requirements. However, for the best performance and to handle larger datasets effectively, it's advisable to have a more powerful system that goes beyond these minimum specifications, particularly in terms of RAM and processor capabilities.