Conversion of number systems essay sample

ASSIGN BUSTER

The number is a symbol or a word used to represent a numeral, while a system is a functionally related groups of elements, so as whole, a number is set or group of symbols to represent numbers or numerals. In other words, any system that is used for naming or representing numbers is a number system. We are quite familiar with decimal number system using ten digits. However digital devices and computers use binary number system instead of decimal number system, having only two digits i. e 0 and 1. Binary number system is based on the same fundamental concept of decimal number system. Various other number systems also use the same fundamental concept of decimal number system e. g. octal number system(using 8 digits) and hexadecimal number systems(using 16 digits).

The knowledge of number systems, their limitations, data formats, inter-conversion and other related terms is essential for understanding of computers and successful programming for digital devices. Understanding all these number systems and particularly their inter conversion of number systems requires a lot of time and a large number of techniques to expertise. The well known number systems and their inter conversions to be discussed are binary, octal, decimal and hexadecimal.

INTRODUCTION

The study of number systems is useful to the student of computing due to the fact that number systems other than the familiar decimal (base 10) number system are used in the computer field. Digital computers internally use the binary (base 2) number system to represent data and perform arithmetic calculations. The binary number system is very efficient for computers, but not for humans. Representing even relatively small numbers

with the binary system requires working with long strings of ones and zeroes. The hexadecimal (base 16) number system (often called “ hex” for short) provides us with a shorthand method of working with binary numbers. One digit in hex corresponds to four binary digits (bits), so the internal representation of one byte can be represented either by eight binary digits or two hexadecimal digits. Less commonly used is the octal (base 8) number system, where one digit in octal corresponds to three binary digits (bits). In the event that a computer user (programmer, operator, end user, etc.) needs to examine a display of the internal representation of computer data (such a display is called a “ dump”), viewing the data in a “ shorthand” representation (such as hex or octal) is less tedious than viewing the data in binary representation.

The binary, hexadecimal , and octal number systems will be looked at in the following pages. The decimal number system that we are all familiar with is a positional number system. The actual number of symbols used in a positional number system depends on its base. The highest numerical symbol always has a value of one less than the base. The decimal number system has a base of 10, so the numeral with the highest value is 9; the octal number system has a base of 8, so the numeral with the highest value is 7, the binary number system has a base of 2, so the numeral with the highest value is 1, etc. Any number can be represented by arranging symbols in specific positions. You know that in the decimal number system, the successive positions to the left of the decimal point represent units (ones), tens, hundreds, thousands, etc. Put another way, each position represents a specific power of base 10. As a computer programmer, we need to know

about the different numeral systems. In our computer works, there will be lots of times when we will be using numeral systems like the binary, hexadecimal and sometimes octal as well. Thus, it would be a great idea to know how to convert numbers from one numeral system to the other.

The digits that all 4 numeral systems use are shown below Decimal| Binary| Hexadecimal| Octal|

0| 0| 0| 0|

1| 1| 1| 1|

2| 10| 2| 2|

3| 11| 3| 3|

4| 100| 4| 4|

5| 101| 5| 5|

6| 110| 6| 6|

7| 111| 7| 7|

8| 1000| 8| 10|

9| 1001| 9| 11|

10| 1010| A| 12|

11| 1011| B| 13|

12| 1100| C| 14|

13| 1101| D| 15|

14| 1110| E| 16|

15| 1111| F| 17|

16| 10000| 10| 20|

Decimal system

The decimal number system, so familiar to us, is the oldest positional number system. In a positional system, a number is represented by a set of symbols. Each symbol represents a particular value depending on its position. The actual number of symbols used in a positional system depends on its base.

The decimal system uses a base of 10 and thus it uses 10 symbols, 0 to 9. Any number can be represented by arranging symbols in various positions. In the decimal systems, each position represents a specific power of 10. Each successive position to the left of the decimal point represents a value ten times greater than the position to its immediate right as shown below:

Position 6 5 4 3 2 1 0 Place value 106 105 104 103 102 101 100

For example, the decimal number 5704 represents:

3 2 1 0 Position

103 102 101 100 Value

5 7 0 4 Decimal point

4 X 100 = 4

0 X101 = 0

7 X 102 = 700

5 X 103 = 5000

We can express this in general form as:

Dm(10m)+dm-1(10m-1)+.....+d0= di

Where di are the decimal symbols, 0 to 9 and m-1 are the number of symbols. This is called the expanded notation for the integer.

Similarly, a fractional part of a decimal number can be represented as: di 10-

i

Where n is the number of symbols in the fractional part.

Binary system

The binary system is the positional number system to the base 2. It uses two symbols 0 and 1. Again, each position in a binary number represents a power of the base as shown below:

Position 6 5 4 3 2 1 0

Place value 26 25 24 23 22 21 20

(64) (32) (16) (8) (4) (2) (1)

Note that each successive position in the integer part of the binary number has a value two times greater than the position to its right.

For example, the binary number 1101 represents the decimal values as shown below:

3 2 1 0

23 22 21 20

1 1 0 1

1 X 1 = 1 0 X 2 = 0 1 X 4 = 4

1 X 8 = 8

Sum 13

Thus, (1101)2 = (13)10

The subscript 2 denotes a number in binary system and 10denotes a number in the decimal system. In general form, it can be written as:

Dm(2m)+dm-1(2m-1)+…….+d0= di 2i

Where di are the binary symbols, 0 or 1. We can further generalize the notation to any base b as di bi

Note that the base b is usually an integer greater than one, and digits di are between 0 and b -1.

The base is sometimes called radix and the fractional point is called radix point.

Hexadecimal system

The hexadecimal system is a number system that uses 16 as its base. This system requires 16 one digit symbols. The first ten symbols are represented by digits 0 through 9 and the remaining six by the letters A through F. The letter A denotes 10, B denotes 11 and so on.

In the hexadecimal system, each position represents a value 16 times greater than the position to its immediate right. The place value of hexadecimal system are shown below:

Position 4 3 2 1 0

Place value 164 163 162 161 160

The following example illustrates the decimal value represented by a hexadecimal number.

3 2 1 0 Position

163 162 161 160 Value

1 2 A F Hexadecimal point 15 X 1 = 15 10 X 16 = 160 2 X 256 = 512

1 X 4096 = 4096

Sum 4783

That is, (12AF)16=(4783)10

To convert a binary number to hexadecimal, we need only to group the binary digits in sets of four and convert each group to its equivalent

hexadecimal digit. Thus, the binary number 0111 1010 0001 0010 0001 becomes 7A121 in hexadecimal. This is illustrated below:

Binary quadruplets 0111 1010 0001 0010 0001

Hexadecimal point 7 A 1 2 1

This example clearly illustrates the advantage of hexadecimal system over binary system. For all large binary numbers, the hexadecimal representation is much more compact and, therefore, easier to write and manipulate than its binary equivalent.

Octal system

The octal number system is a system having base b as 8. The eight octal symbols are 0 through 7. The place values in the octal system are powers of 8 as shown below:

Position 4 3 2 1 0

Place value 84 83 82 81 80

The position values increase by a factor of 8 from right to left. The example below shows an octal number and its equivalent decimal value:

Position 3 2 1 0

Value 83 82 81 80

Octal point 2 0 5 6

6 X 1 = 6 5 X 8 = 40 0 X 64 = 0 2 X 512 = 1024 Sum 1070 Thus, 20568 = 107010

Since, 8= 2^3, each octal digit has a unique 3 bit binary representation. Just as in the hexadecimal system, to convert a binary number to octal, it is only necessary to group the binary digits in sets of three and convert each

set to its octal equivalent.

For example, the binary number 1011010 can be represented in octal as follows:

Binary triplets 001 011 010

Octal equivalent 1 3 2

ALGORITHMS:

1. Conversion of binary to decimal number system

* Integer part:

i. Multiply the leftmost digit by the base 2.

ii. Add the next digit to the right of the product.

iii. Multiply the sum by base 2 and add the next digit.

iv. Continue this process till the last rightmost digit is added. * Fractional part:

i. Multiply the leftmost digit by the reciprocal of the base i. e 0. 5 (1/2). ii. Add the next digit to the left of the product.

iii. Multiply the sum by the reciprocal of the base i. e 0. 5 and add the next digit. iv. Continue this process till the last leftmost digit in the fractional part is added. v. Multiply the last sum by 0. 5.

2. Conversion of octal to decimal number system

* Integer part:

i. Multiply the leftmost digit by the base 8.

ii. Add the next digit to the right of the product.

iii. Multiply the sum by base 8 and add the next digit.

iv. Continue this process till the last rightmost digit is added. * Fractional part:

i. Multiply the leftmost digit by the reciprocal of the base (1/8). ii. Add the next digit to the left of the product.

iii. Multiply the sum by the reciprocal of the base (1/8) and add the next digit. iv. Continue this process till the last leftmost digit in the fractional part is added. v. Multiply the last sum by 1/8.

3. Conversion of hexadecimal to decimal number system

* Integer part:

i. Multiply the leftmost digit by the base 16.

ii. Add the next digit to the right of the product.

iii. Multiply the sum by base 16 and add the next digit.

iv. Continue this process till the last rightmost digit is added. * Fractional part:

i. Multiply the leftmost digit by the reciprocal of the base (1/16). ii. Add the next digit to the left of the product.

iii. Multiply the sum by the reciprocal of the base (1/16) and add the next digit. iv. Continue this process till the last leftmost digit in the fractional part is added. v. Multiply the last sum by 1/16.

4. Conversion of decimal to binary number system

* Integer part:

i. Divide the integer part of the decimal number by the base 2. The remainder would constitute the rightmost digit of the integer part of the new number. ii. Divide the quotient again by base 2. The remainder is the second digit from the right. iii. Continue this process until a zero quotient is obtained. The last remainder is the leftmost digit of the new number. * Fractional part:

i. Multiply the fractional part by the base2 i. e base of the new system. The integral part of the product constitutes the leftmost digit of fractional part of the new number. ii. Multiply the fractional part of the product by base 2. The integral part of the resultant product is the second digit from left. iii. Continue the process until a zero fractional part or a duplicate fractional part occurs. The integer part of the last product will be the rightmost digit of the fractional part. 5. Conversion of decimal to octal number system

* Integer part:

i. Divide the integer part of the decimal number by the base 2. The remainder would constitute the rightmost digit of the integer part of the new number. ii. Divide the quotient again by base 2. The remainder is the second digit from the right. iii. Continue this process until a zero quotient is obtained. The last remainder is the leftmost digit of the new number. * Fractional part:

i. Multiply the fractional part by the base2 i. e base of the new system. The integral part of the product constitutes the leftmost digit of fractional part of the new number. ii. Multiply the fractional part of the product by base 2. The integral part of the resultant product is the second digit from left. iii. Continue the process until a zero fractional part or a duplicate fractional part occurs. The integer part of the last product will be the rightmost digit of the fractional part. 6. Conversion of decimal to hexadecimal number system

* Integer part:

i. Divide the integer part of the decimal number by the base 2. The remainder would constitute the rightmost digit of the integer part of the new

number. ii. Divide the quotient again by base 2. The remainder is the second digit from the right. iii. Continue this process until a zero quotient is obtained. The last remainder is the leftmost digit of the new number. * Fractional part:

i. Multiply the fractional part by the base2 i. e base of the new system. The integral part of the product constitutes the leftmost digit of fractional part of the new number. ii. Multiply the fractional part of the product by base 2. The integral part of the resultant product is the second digit from left. iii. Continue the process until a zero fractional part or a duplicate fractional part occurs. The integer part of the last product will be the rightmost digit of the fractional part. 7. Conversion of octal to hexadecimal number system

* Integer part:

i. Write the octal number.

ii. Place the binary equivalent of each digit below the number i. e in triplets.

iii. Regroup them as binary quadruplets starting from the rightmost digit of the integral part, with zero’s added if necessary. iv. Convert each group into its equivalent hexadecimal digit. * Fractional part:

i. Write the octal number.

ii. Place the binary triplets equivalent of each digit below the number. iii. Regroup them as binary quadruplets starting from the leftmost digit of the fractional part, with zero’s added if necessary. iv. Convert each group into its equivalent hexadecimal digit. 8. Conversion of octal to binary number system

i. Write the octal number.

ii. Then write the equivalent binary quadruplets of each digit of the octal number. 9. Conversion of hexadecimal to octal number system

* Integer part:

i. Write the hexadecimal number.

ii. Place the equivalent binary quadruplets below the number. iii. Regroup them as binary triplets starting from the rightmost digit of the integral part, with zero’s added if necessary. iv. Convert each group into its equivalent octal digit.

* Fractional part:

i. Write the hexadecimal number.

ii. Place the equivalent binary quadruplets below the number. iii. Regroup them as binary triplets starting from the leftmost digit of the fractional part, with zero’s added if necessary. iv. Convert each group into its equivalent octal digit.

10. Conversion of hexadecimal to binary number system

iii. Write the hexadecimal number.

iv. Then write the equivalent binary quadruplets of each digit of the hexadecimal number.

11. Conversion of binary to hexadecimal number system

i. Group the binary number in sets of 4 and write the corresponding equivalent hexadecimal digit. 12. Conversion of binary to octal number system

i. Group the binary number in sets of 3 and write the corresponding equivalent octal digit

EXAMPLES

1. Binary to decimal conversion

Since binary is a base-2 system, each digit represents an increasing power of 2, with the rightmost digit representing 20, the next representing 21, then 22, and so on. To determine the decimal representation of a binary number simply take the sum of the products of the binary digits and the powers of 2 which they represent. For example

1010 is a binary value and here we converted into decimal

[(1) × 23] + [(0) × 22] + [(1) × 21] + [(0) × 20] = [1 × 8] + [0 × 4] + [1 × 2] + [0 × 1] = 10 Fractional Decimal Number Conversion

For example, the fractional binary number 010101 is converted to decimal form by 0 × 2-1 + 1 × 2-2 + 0 × 2-3 + 1 × 2-4 = 0. 312

2. Octal to decimal conversion

Convert 547 octal to decimal value,

(5 × 82) + (4 × 81) + (7 × 80) = 359.

3. Hexadecimal to decimal conversion

Here we convert the hexadecimal value of 4B5A to decimal

(4 × 163) + (11 × 162) + (5 × 161) + (10 × 160) = 19290. 4. Decimal to binary conversion

Step1: If the number is odd , write 1 on the right corner (or) it is even means write 0. Step2: Then divide the number by 2. If the answer is in decimal point (Eg; 4. 5) ignore the fraction value (ie. 4) Step3: Continue the step 1 & 2 until end up with 1.

Step4: Note the final binary value from bottom to up.

Example is given below,

Here we convert 75 to binary value,

75/2 = 37. 5 ———- 1

37/2 = 18. 5 ———- 1

18/2 = 9 ———- 0

9/2 = 4. 5 ———- 1

4/2 = 2 ———- 0

2/2 = 1 ———- 0

1 ———- 1

That is 10010112 is the binary equivalent

5. Decimal to octal conversion

Convert (4321)10 to equivalent octal value

4321/8 = 540 ———- 1

540/8 = 67 ———- 4

67/8 = 8 ———- 3

8/8 = 1 ———- 0

1 ———- 1

The equivalent octal value is (10341)8

6. Decimal to hexadecimal conversion

The steps followed to convert decimal to binary are

Step1: Divide the number by 16. Write the remainder on the right corner. If the remainder is more than 9 it is replaced by corresponding symbol. Step2: Note the final Hexadecimal. value from bottom to up. Example is given below,

Here we convert 4321 to Octal value,

4321/16 = 270 ——–1

279/16 = 16 ——–E

16/16 = 1 ———0

1 ———1

Hence the Hexadecimal equivalent is 10E116

7. Octal to hexadecimal conversion

For easy conversion we have to follow two steps to convert octal to Hexadecimal Step1: convert Octal to binary value.

Step2: Then convert the binary value into hexadecimal.

For Example:

Convert 547 to Hexadecimal,

5 4 7

Step1: 101 100 111

Step2: Arrange the above value in digits,

0001 0110 0111

Convert binary to Hexadecimal ,

1 6 7

Hence the required Hexadecimal equivalent is 167.

8. Octal to binary conversion

It is very easy calculate the equivalent binary number for a given octal number, each digit of a hex number is individually converted to its binary equivalent. For Example:

Convert 547 to binary ,

5 ——— 101

4 ——— 100

7 ——— 111

Hence the required binary equivalent is 101100111.

9. Hexadecimal to octal conversion

For easy conversion we have to follow two steps to convert Hexadecimal to octal. Step1: convert hexadecimal to binary

Step2: Then convert the binary value into octal.

For Example:

Convert 4B5 to octal,

4B5 = 0100 1011 0101

Then convert the binary to octal,

010 010 110 101

2 2 6 5

2265 is the required Octal value.

10. Hexadecimal to binary conversion

It is very easy calculate the equivalent binary number for a given Hexadecimal number, each digit of a hex number is individually converted to its binary equivalent. For Example: covert 4B5 to binary value

4 ————- 0100

B ————- 1011

5 ————- 0101

Hence binary equivalent value is 010010110101.

11. Binary to hexadecimal conversion

To convert a binary number into its hexadecimal equivalent, divide it into

groups of four bits. If the number of bits isn’t a multiple of four, simply insert extra 0 bits at the left are called padding. For example: 10100102 = 0101 0010 grouped with padding = 5216

110111012 = 1101 1101 grouped = DD16

12. Binary to octal conversion

Binary is also easily converted to the octal numeral system, since octal uses a radix of 8, which is a power of two. Converting from octal to binary proceeds in the same fashion as it does for hexadecimal. For example:

110 1012 = 658

001 1112 = 178

PROGRAMS:

```
#include
#include
#include
#include
#include
#include
#define MAX 1000
//bin2dec
void bin2dec()
{
int a[20], b[20];
int i, j, k, cntint, cntfra, flag, rem;
float rem1;
char s[20];
```

```
cntint= cntfra= flag= rem= 0;

rem1= 0. 0;

clrscr();

printf(“ Enter the binary no : “);

scanf(“%s”, s);

for(i= 0, j= 0, k= 0; i {

if(s[i]==’.’)

{

flag= 1;

}

else if(flag== 0)

a[j++]= s[i]-48;

else if(flag== 1)

b[k++]= s[i]-48;

}

cntint= j;

cntfra= k;

for(j= 0, i= cntint-1; j {

rem = rem + (a[j] * pow(2, i));

}

for(k= 0, i= 1; k {

rem1 = rem1 + (b[k] / pow(2, i));

}

rem1 = rem + rem1;

printf(“ The decimal value of the given binary is : %f”, rem1); }
```

```
//oct2dec

void oct2dec()
{
int a[20], b[20];
int i, j, k, c, fra, flag, rem;
float rem1;
char s[20];
c= fra= flag= rem= 0;
rem1= 0. 0;
clrscr();
printf(“ ENTER THE OCTAL NUMBER : “);
scanf(“%s”, s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
a[j++]= s[i]-48;
else if(flag== 1)
b[k++]= s[i]-48;
}
c= j;
fra= k;
for(j= 0, i= c-1; j {
```

```
rem = rem +(a[j] * pow(8, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 +(b[k] / pow(8, i));
}
rem1= rem+rem1;
printf(“ THE DECIMAL VALUE OF GIVEN OCTAL NO IS : %f”, rem1); }

//hex2dec
void hex2dec()
{
int a[20], b[20];
int i, j, k, c, fra, flag, rem;
float rem1;
char s[40];
c= fra= flag= rem= 0;
rem1= 0. 0;
clrscr();
printf(“ Enter the hexadecimal no : “);
scanf(“%s”, s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
```

```
{ if(s[i]==’A’)
{
a[j++]= 10;
}
else if(s[i]==’B’)
{
a[j++]= 11;
}
else if(s[i]==’C’)
{
a[j++]= 12;
}
else if(s[i]==’D’)
{
a[j++]= 13;
}
else if(s[i]==’E’)
{
a[j++]= 14;
}
else if(s[i]==’F’)
{
a[j++]= 15;
}
else
```

```
{
a[j++]= s[i]-48;
}
}
else if(flag== 1)
{
if(s[i]==’A’)
{
b[k++]= 10;
}
else if(s[i]==’B’)
{
b[k++]= 11;
}
else if(s[i]==’C’)
{
b[k++]= 12;
}
else if(s[i]==’D’)
{
b[k++]= 13;
}
else if(s[i]==’E’)
{
b[k++]= 14;
}
```

```
else if(s[i]==’F’)
{
b[k++]= 15;
}
else

b[k++]= s[i]-48;
}
}
c= j;
fra= k;
for(j= 0, i= c-1; j {
rem = rem + (a[j] * pow(16, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 + (b[k] / pow(16, i));
}
rem1 = rem + rem1;
printf(“ The decimal value of the given hexadecimal number is : %f”, rem1);

}

void main()
{
int choice;
clrscr();
do
```

```
{
printf(“ Enter your choice:”);
printf(“ 1. Bin2dec”);
printf(“ 2. Oct2dec”);
printf(“ 3. Hex2dec”);
scanf(“%d”,&choice);
switch(choice)
{
case 1:
bin2dec(); break;
case 2:
oct2dec();
break;
case 3:
hex2dec();
break;
default:
exit(0);
}
}
while(choice <= 3);
getch();
}

//bin2hex
#include
```

```
#include
#include
#include
void main()
{
int rev[20], a[20], b[20], p[20];
int i, j, k, l, flag, c, fra;
int rem, num1, num3;
float rem1, dno, num2, num4;
char s[20];
clrscr();
c= fra= 0;
flag= l= 0;
rem= 0;
printf(“ ENTER THE BINARY NO : “);
scanf(“%s”,&s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
{
if(s[i]==’A’)
{
a[k++]= 10;
```

```
}
else if(s[i]==’B’)
{
a[k++]= 11;
}
else if(s[i]==’C’)
{
a[k++]= 12;
}
else if(s[i]==’D’)
{
a[k++]= 13;
}
else if(s[i]==’E’)
{
a[k++]= 14;
}
else if(s[i]==’F’)
{
a[k++]= 15;
}
else
{
a[k++]= s[i]-48;
}
}
```

```
else if(flag== 1)
{
if(s[i]==’A’)
{
b[j++]= 10;
}
else if(s[i]==’B’)
{
b[j++]= 11;
}
else if(s[i]==’C’)
{
b[j++]= 12;
}
else if(s[i]==’D’)
{
b[j++]= 13;
}
else if(s[i]==’E’)
{
b[j++]= 14;
}
else if(s[i]==’F’)
{
b[j++]= 15;
}
```

```
else
{
b[j++]= s[i]-48;
}
}
}
c= k;
fra= j;
for(j= 0, i= c-1; j {
rem = rem + (a[j] * pow(2, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 + (b[k] / pow(2, i));
}
rem1 = rem + rem1;
dno= rem1;
num1 = (int)dno;
num2 = dno – num1;
i= 0;
while(num1!= 0)
{
rem = num1 % 16;
rev[i] = rem;
num1 = num1 / 16;
i++;
}
```

```
j= 0;
while(num2!= 0. 0)
{
num2 = num2 * 16;
num3 = (int) num2;
num4 = num2 – num3;
num2 = num4;
p[j] = num3;
j++;
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE HEXADECIMAL VALUE OF GIVEN BINARY NO IS : “); for(i= l-1;
i>= 0; i–)
{
if(rev[i]== 15)
{
printf(“ F”);
}
else if(rev[i]== 14)
{
printf(“ E”);
}
```

```
else if(rev[i]== 13)
{
printf(“ D”);
}
else if(rev[i]== 12)
{
printf(“ C”);
}
else if(rev[i]== 11)
{
printf(“ B”);
}
else if(rev[i]== 10)
{
printf(“ A”);
}
else
{

printf(“%d”, rev[i]);
}
}
printf(“.”);
for( k= 0; k {
if(p[k]== 15)
{
```

```
printf(“ F”);
}
else if(p[k]== 14)
{
printf(“ E”);
}
else if(p[k]== 13)
{
printf(“ D”);
}
else if(p[k]== 12)
{
printf(“ C”);
}
else if(p[k]== 11)
{
printf(“ B”);
}
else if(p[k]== 10)
{
printf(“ A”);
}

else
{
printf(“%d”, p[k]);
```

```
}
}
getch();
}

//oct2hex
#include
#include
#include
#include
void main()
{
int a[20], b[20], c[20], rev[20];
int h, i, j, k, l, x, fra, flag, rem, num1, num3;
float rem1, num2, num4, dno;
char s[20];
x= fra= flag= rem= 0;
rem1= 0. 0;
clrscr();
printf(“ ENTER THE OCTAL NUMBER : “);
scanf(“%s”, s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
```

```
else if(flag== 0)
a[j++]= s[i]-48;
else if(flag== 1)
b[k++]= s[i]-48;
}
x= j;
fra= k;
for(j= 0, i= x-1; j {
rem = rem +(a[j] * pow(8, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 +(b[k] / pow(8, i));
}
rem1= rem+rem1;
dno= rem1;
num1=(int)dno;
num2= dno-num1;

i= 0;
while(num1!= 0)
{
rem= num1 % 16;
rev[i] = rem;
num1= num1 /16;
i++;
}
```

```
j= 0;
while(num2!= 0. 0)
{
num2= num2 * 16;
num3=(int)num2;
num4= num2-num3;
num2= num4;
a[j]= num3;
j++;
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE HEXADECIMAL VALUE OF GIVEN OCTAL NO IS : “); for(i= l-1; i>=
0; i–)
{
if(rev[i]== 10)
printf(“ A”);
else if(rev[i]== 11)
printf(“ B”);
else if(rev[i]== 12)
printf(“ C”);
else if(rev[i]== 13)
printf(“ D”);
```

```
else if(rev[i]== 14)
printf(“ E”);
else if(rev[i]== 15)
printf(“ F”);
else
printf(“%d”, rev[i]);
}
h= j;
printf(“.”);
for(k= 0; k {
if(a[k]== 10)
printf(“ A”);
else if( a[k]== 11)
printf(“ B”);
else if(a[k]== 12)
printf(“ C”);
else if(a[k]== 13)
printf(“ D”);
else if(a[k]== 14)
printf(“ E”);
else if(a[k]== 15)
printf(“ F”);
else
printf(“%d”, a[k]);
}
```

```
getch();
}

#include
#include
#include
#include
#include
void main()
{
int a[20], b[20], c[20], rev[20];
int h, i, j, k, l, x, fra, flag, rem, num1, num3;
float rem1, num2, num4, dno;
char s[20];
x= fra= flag= rem= 0;
rem1= 0. 0;
clrscr();
printf(“ ENTER THE DECIMAL NUMBER : “);
scanf(“%s”, s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
a[j++]= s[i]-48;
else if(flag== 1)
```

```
b[k++]= s[i]-48;
}
x= j;
fra= k;
for(j= 0, i= x-1; j {
rem = rem +(a[j] * pow(10, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 +(b[k] / pow(10, i));
}
rem1= rem+rem1;
dno= rem1;
num1=(int)dno;
num2= dno-num1;

i= 0;
while(num1!= 0)
{
rem= num1 % 16;
rev[i] = rem;
num1= num1 /16;
i++;
}
j= 0;
while(num2!= 0. 0)
{
```

```
num2= num2 * 16;
num3=(int)num2;
num4= num2-num3;
num2= num4;
a[j]= num3;
j++;
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE HEXADECIMAL VALUE OF GIVEN DECIMAL NO IS : “); for(i= l-1;
i>= 0; i–)
{
if(rev[i]== 10)
printf(“ A”);
else if(rev[i]== 11)
printf(“ B”);
else if(rev[i]== 12)
printf(“ C”);
else if(rev[i]== 13)
printf(“ D”);
else if(rev[i]== 14)
printf(“ E”);
else if(rev[i]== 15)
```

```
printf(“ F”);
else
printf(“%d”, rev[i]);
}
h= j;
if(h!= 0)
{
printf(“.”);
}
for(k= 0; k {
if(a[k]== 10)
printf(“ A”);
else if( a[k]== 11)
printf(“ B”);
else if(a[k]== 12)
printf(“ C”);
else if(a[k]== 13)
printf(“ D”);
else if(a[k]== 14)
printf(“ E”);
else if(a[k]== 15)
printf(“ F”);
else
printf(“%d”, a[k]);
}
getch();
```

```
}
#include
#include
#include
#include
void main()
{
int a[20], b[20], c[20], rev[20];
int h, i, j, k, l, x, fra, flag, rem, num1, num3;
float rem1, num2, num4, dno;
char s[20];
x= fra= flag= rem= 0;
rem1= 0. 0;
clrscr();
printf(“ ENTER THE DECIMAL NUMBER : “);
scanf(“%s”, s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
a[j++]= s[i]-48;
else if(flag== 1)
b[k++]= s[i]-48;
}
```

```
x= j;
fra= k;
for(j= 0, i= x-1; j {
rem = rem +(a[j] * pow(10, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 +(b[k] / pow(10, i));
}
rem1= rem+rem1;
dno= rem1;
num1=(int)dno;
num2= dno-num1;

i= 0;
while(num1!= 0)
{
rem= num1 % 16;
rev[i] = rem;
num1= num1 /16;
i++;
}
j= 0;
while(num2!= 0. 0)
{
num2= num2 * 16;
num3=(int)num2;
```

```
num4= num2-num3;
num2= num4;
a[j]= num3;
j++;
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE HEXADECIMAL VALUE OF GIVEN DECIMAL NO IS : “); for(i= l-1;
i>= 0; i–)
{
if(rev[i]== 10)
printf(“ A”);
else if(rev[i]== 11)
printf(“ B”);
else if(rev[i]== 12)
printf(“ C”);
else if(rev[i]== 13)
printf(“ D”);
else if(rev[i]== 14)
printf(“ E”);
else if(rev[i]== 15)
printf(“ F”);
else
```

```
printf(“%d”, rev[i]);
}
h= j;
if(h!= 0)
{
printf(“.”);
}
for(k= 0; k {
if(a[k]== 10)
printf(“ A”);
else if( a[k]== 11)
printf(“ B”);
else if(a[k]== 12)
printf(“ C”);
else if(a[k]== 13)
printf(“ D”);
else if(a[k]== 14)
printf(“ E”);
else if(a[k]== 15)
printf(“ F”);
else
printf(“%d”, a[k]);
}
getch();
}
```

```
//hex2oct
#include
#include
#include
#include

void main()
{
int rev[20], a[20], b[20], p[20];
int i, j, k, l, flag, c, fra;
int rem, num1, num3;
float rem1, dno, num2, num4;
char s[20];
clrscr();
c= fra= 0;
flag= l= 0;
rem= 0;
printf(“ ENTER THE HEXADECIMAL NO : “);
scanf(“%s”,&s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
{
```

```
if(s[i]==’A’)
{
a[k++]= 10;
}
else if(s[i]==’B’)
{
a[k++]= 11;
}
else if(s[i]==’C’)
{
a[k++]= 12;
}
else if(s[i]==’D’)
{
a[k++]= 13;
}
else if(s[i]==’E’)
{
a[k++]= 14;
}
else if(s[i]==’F’)
{
a[k++]= 15;
}
else
{
```

```
a[k++]= s[i]-48;
}
}
else if(flag== 1)
{
if(s[i]==’A’)
{
b[j++]= 10;
}
else if(s[i]==’B’)
{
b[j++]= 11;
}
else if(s[i]==’C’)
{
b[j++]= 12;
}
else if(s[i]==’D’)
{
b[j++]= 13;
}
else if(s[i]==’E’)
{
b[j++]= 14;
}
else if(s[i]==’F’)
```

```
{
b[j++]= 15;
}
else
{
b[j++]= s[i]-48;
}
}
}
c= k;
fra= j;
for(j= 0, i= c-1; j {
rem = rem + (a[j] * pow(16, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 + (b[k] / pow(16, i));
}
rem1 = rem + rem1;
dno= rem1;
num1 = (int)dno;
num2 = dno – num1;
i= 0;
while(num1!= 0)
{
rem = num1 % 8;
rev[i] = rem;
```

```
num1 = num1 / 8;
i++;
}
j= 0;
while(num2!= 0. 0)
{
num2 = num2 * 8;
num3 = (int) num2;
num4 = num2 – num3;
num2 = num4;
p[j] = num3;
j++;
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE OCTAL VALUE OF GIVEN HEXADECIMAL NO IS : “); for(i= l-1; i>=
0; i–)
{

//oct2bin
#include
#include
#define MAX 1000
```

```
int main()
{
char octalNumber[MAX];
long int i= 0;
clrscr();
printf(“ Enter any octal number: “);
scanf(“%s”,&octalNumber);
printf(“ Equivalent binary value: “);
while(octalNumber[i])
{
switch(octalNumber[i])
{
case ‘ 0’: printf(“ 000”); break;
case ‘ 1’: printf(“ 001”); break;
case ‘ 2’: printf(“ 010”); break;
case ‘ 3’: printf(“ 011”); break;
case ‘ 4’: printf(“ 100”); break;
case ‘ 5’: printf(“ 101”); break;
case ‘ 6’: printf(“ 110”); break;
case ‘ 7’: printf(“ 111”); break;
case ‘.’: printf(“.”); break;
default: printf(“ Invalid octal digit %c “, octalNumber[i]); return 0; }
i++;
}
getch();
```

```
return 0;
}

//hex to bin
#include
#include
#define MAX 1000
int main()
{
char binaryNumber[MAX], hexaDecimal[MAX];
long int i= 0;
clrscr();
printf(“ Enter any hexadecimal number: “);
scanf(“%s”,&hexaDecimal);

printf(“ Equivalent binary value: “);
while(hexaDecimal[i]){
switch(hexaDecimal[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’: printf(“ 1010”); break;
case ‘ B’: printf(“ 1011”); break;
case ‘ C’: printf(“ 1100”); break;
case ‘ D’: printf(“ 1101”); break;
case ‘ E’: printf(“ 1110”); break;
case ‘ F’: printf(“ 1111”); break;
case ‘ a’: printf(“ 1010”); break;
case ‘ b’: printf(“ 1011”); break;
case ‘ c’: printf(“ 1100”); break;
case ‘ d’: printf(“ 1101”); break;
case ‘ e’: printf(“ 1110”); break;
case ‘ f’: printf(“ 1111”); break;
case ‘.’: printf(“.”); break;
default : printf(“ Invalid hexadecimal digit %c “, hexaDecimal[i]); return 0;
}
i++;
}

getch();
return 0;
}

//bin2oct
#include
#include
```

```
#include

#include

void main()

{

int rev[20], a[20], b[20], p[20];

int i, j, k, l, flag, c, fra;

int rem, num1, num3;

float rem1, dno, num2, num4;

char s[20];

clrscr();

c= fra= 0;

flag= l= 0;

rem= 0;

printf(“ ENTER THE BINARY NO : “);

scanf(“%s”,&s);

for(i= 0, j= 0, k= 0; i {

if(s[i]==’.’)

{

flag= 1;

}

else if(flag== 0)

a[k++]= s[i]-48;

else if(flag== 1)

b[j++]= s[i]-48;

}
```

```
c= k;
fra= j;
for(j= 0, i= c-1; j {
rem = rem + (a[j] * pow(2, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 + (b[k] / pow(2, i));
}
rem1 = rem + rem1;
dno= rem1;
num1 = (int)dno;
num2 = dno – num1;
i= 0;
while(num1!= 0)
{
rem = num1 % 8;
rev[i] = rem;
num1 = num1 / 8;
i++;
}
j= 0;
while(num2!= 0. 0)
{
num2 = num2 * 8;
num3 = (int) num2;
num4 = num2 – num3;
```

```
num2 = num4;
p[j] = num3;
j++;
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE OCTAL VALUE OF GIVEN BINARY NO IS : “);
for(i= l-1; i>= 0; i–)
{
printf(“%d”, rev[i]);
}
printf(“.”);
for( k= 0; k {
printf(“%d”, p[k]);
}
getch();
}

//bin2oct
#include
#include
#include
#include
```

```
void main()
{
int rev[20], a[20], b[20], p[20];
int i, j, k, l, flag, c, fra;
int rem, num1, num3;
float rem1, dno, num2, num4;
char s[20];
clrscr();
c= fra= 0;
flag= l= 0;
rem= 0;
printf(“ ENTER THE BINARY NO : “);
scanf(“%s”,&s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
a[k++]= s[i]-48;
else if(flag== 1)
b[j++]= s[i]-48;
}
c= k;
fra= j;
for(j= 0, i= c-1; j {
```

```
rem = rem + (a[j] * pow(2, i));
}
for(k= 0, i= 1; k {
rem1 = rem1 + (b[k] / pow(2, i));
}
rem1 = rem + rem1;
dno= rem1;
num1 = (int)dno;
num2 = dno – num1;
i= 0;
while(num1!= 0)
{
rem = num1 % 8;
rev[i] = rem;
num1 = num1 / 8;
i++;
}
j= 0;
while(num2!= 0. 0)
{
num2 = num2 * 8;
num3 = (int) num2;
num4 = num2 – num3;
num2 = num4;
p[j] = num3;
j++;
```

```
if(j== 4)
{
break;
}
}
l= i;
printf(“ THE OCTAL VALUE OF GIVEN BINARY NO IS : “);
for(i= l-1; i>= 0; i–)
{
printf(“%d”, rev[i]);
}
printf(“.”);
for( k= 0; k {
printf(“%d”, p[k]);
}
getch();
}

//dec2oct
#include
#include
#include
#include

void main()
{
int rev[20], a[20], b[20], p[20];
```

```
int i, j, k, l, flag, c, fra;
int rem, num1, num3;
float rem1, dno, num2, num4;
char s[20];
clrscr();
c= fra= 0;
flag= l= 0;
rem= 0;
printf(“ ENTER THE DECIMAL NO : “);
scanf(“%s”,&s);
for(i= 0, j= 0, k= 0; i {
if(s[i]==’.’)
{
flag= 1;
}
else if(flag== 0)
a[k++]= s[i]-48;
else if(flag== 1)
b[j++]= s[i]-48;
}
c= k;
fra= j;
for(j= 0, i= c-1; j {
rem = rem + (a[j] * pow(10, i));
}
for(k= 0, i= 1; k {
```

```
rem1 = rem1 + (b[k] / pow(10, i));
}
rem1 = rem + rem1;
dno= rem1;
num1 = (int)dno;
num2 = dno – num1;
i= 0;
while(num1!= 0)
{
rem = num1 % 8;
rev[i] = rem;
num1 = num1 / 8;
i++;
}
j= 0;
while(num2!= 0. 0)
{
num2 = num2 * 8;
num3 = (int) num2;
num4 = num2 – num3;
num2 = num4;
p[j] = num3;
j++;
if(j== 4)
{
break;
```

```
}
}
l= i;
printf(“ THE OCTAL VALUE OF GIVEN DECIMAL NO IS : “);
for(i= l-1; i>= 0; i–)
{
printf(“%d”, rev[i]);
}
printf(“.”);
for( k= 0; k {
printf(“%d”, p[k]);
}
getch();
}

//dec2bin
#include
#include
void main()
{
int i, n, j, b[100], m;
clrscr();
printf(“ Enter a Number:”);
scanf(“%d.%d”,&n,&m);
for(i= 0; i <= n; i++)
{
```

```
b[i]= n%2;
n= n/2;
}
printf(“ Binary Equivalent:”);
for(i= i+0; i>= 0; i–)
printf(“%d”, b[i]);
i= 0;
m1= m;
while(m1!= 0)
{
m1= m1/10;
i++;
}
while(i!= 0)
{
m= m*0. 1;
i–;
}
getch();
}
```

ADVANTAGES:

1. The advantage of the binary system is its simplicity. A computing device can be created out of anything that has a series of switches, each of which can alternate between an “ on” position and an “ off” position. These switches can be electronic, biological, or mechanical, as long as they can be

moved on command from one position to the other. 2. Most computers have electronic switches. When a switch is “ on” it represents the value of one, and when the switch is “ off” it represents the value of zero. Digital devices perform mathematical operations by turning binary switches on and off. The faster the computer can turn the switches on and off, the faster it can perform its

calculations. 3. Hexadecimal (base 16) converts easily to binary (base 2), which is used by computers at the fundamental level. DISADVANTAGES

1. A binary number has more digits than its decimal equivalent i. e it will be longer. Binary is an effective number system for computers because it is easy to implement with digital electronics. It is inefficient for humans to use binary, however, because it requires so many digits to represent a number. The number 76, for example, takes only two digits to write in decimal, yet takes seven digits to write in binary (1001100). This is not a problem for the computer but it makes it harder for us to read and work with. 2. Binary is more difficult than decimal for us to read as we are more used to decimal.

APPLICATIONS

1. It is particularly useful for human communications with a computer.
2. It is used in memory systems.
3. It is useful to convert hexadecimal to binary and vice-versa easily.
4. It is used in digital computing.
5. Also used in encoding.

REFERENCES

1. Number-system-conversion. html

2. Numerical Methods: By E. Balagurusamy

3. Digital Logic Design: By R. P. Jain