Date :   Page No :

Assignment Number

Problem Statement

Program in C to convert a number between two bases.

Theory

The base of a number system is defined as the number of individual digits present in the system. For example, in decimal system, there are total 10 digits, from 0-9. Hence the base of decimal number system is 10. Similarly, there are only 0 and 1 in binary number system, hence the base is 2. Defining base is important to evaluate the value of a number in an weighted number system. For example, consider the following number in base B :

$$(d_2d_1d_0)_B$$

The equivalent decimal value for this number will be :

$$(d_2d_1d_0)_B = (B^2 \times d_2 + B^1 \times d_1 + B^0 \times d_0)_{10}$$

Hence to convert between any to arbitrary bases, we first need to convert it to decimal, and then convert the decimal value to the desired base. To convert any number from decimal to a desired base B, we divide the number by B, until a quotient of 0 is obtained. Then we collect the remainder of each step reversely, to get the final output.

Examples :

1. **Binary to octal :** $(1111)_2 = (15)_{10} = (17)_8$
2. **Hexadecimal to binary :** $(ffff)_{16} = (65535)_{10} = (1111111111111111)_2$
3. **Octal to hexadecimal :** $(77)_8 = (63)_{10} = (3f)_{16}$

Algorithms

1. Algorithm for Get_Cval(character, inputBase)

```
// returns the equivalent decimal value of a character in given inputBase
    Step 1:  If(ascii(c) >= ascii(0) and ascii(c) <= ascii(9))
        // ascii() is a function which returns the ascii value of a character
             Then
        a)  Return ascii(c) - ascii(0)
    Step 2:  Else If(ascii(c) >= ascii(a) and ascii(c) <= (ascii(a) + inputBase - 11))
             Then
        a)  Return ascii(c) - ascii(a) + 10
    Step 3:  Else If(ascii(c) >= ascii(A) and ascii(c) <= (ascii(A) + inputBase - 11))
             Then
        a)  Return ascii(c) - ascii(A) + 10
    Step 4:  Else
        a)  Return -1 // the character should not be present in given base
             [End of if structure]
```

2. Algorithm for Con_Lt10(number, inputBase)

```
// to convert a number from a base less than 10 to decimal
    Step 1:  Set i = 0
    Step 2:  Repeat through step 2.a to 2.c while(n >= 1)
             Begin
        a)  Set digit[i] = number % 10
        b)  Set number = number / 10 // '/' denotes integer division operation
        c)  Set i = i + 1
        [End of While loop]
    Step 3:  Set bak = i - 1
    Step 4:  Set i = 0
    Step 5:  Set sum = 0
    Step 6:  Repeat through step 6.a to 6.b while(i <= bak)
             Begin
        a) Set sum = sum + (digit[i] * $inputBase^{i}$)
        b) Set i = i + 1
             [End of While loop]
    Step 7:  Return sum
```

3. Algorithm for Con_Gt10(hex, inputBase)

```
// to convert a number from a base greater than 10 to decimal
Step 1:  Set len = length_of(hex) - 1 // length_of is a function which returns
    the length
                                        // of an array
Step 2:  Set dec = 0
Step 3:  Set i = 0
Step 4:  Set val = 0
Step 5:  Repeat through step 5.a to 5.d while (hex[i] != Null)
         Begin
    a) Set val = Get_Cval(hex[i], inputBase)
    b) If(val = -1) // The character should not be present in this inputBase
       Then
       1. Print "Invalid character" hex[i] "for base" inputBase
       2. Set hasBadChar = 1 // Flag to denote a bad character in input
       [End of if structure]
    c) Set dec = dec + (val * $inputBase^{len}$)
    d) Set len = len-1
         [End of while loop]
Step 6:  Return dec
```

4. Algorithm for Main()

```
Step 1:  Set choice = y
Step 2:  Repeat through step 2.a to 2.o while (choice = y)
         Begin
   a) Print "Enter the input base : "
   b) Read inputBase
   c) Print "Enter the number : "
   d) If(inputBase < 2 or inputBase > 16)
      Then
      1. Print "Input base must fall within the range [2,16]
      2. Exit
      [End of if structure]
   e) If(inputBase = 16)
      Then
      1. Read hex
   f) Else
      1. Read number
      [End of if structure]
   g) Print "Enter the output base : "
   h) Read outputBase
   i) If(outputBase < 2 or outputBase > 16)
      Then
      1. Print "Output base must fall within the range [2,16]"
      2. Exit
      [End of if structure]
   j) If(inputBase = 10)
      Then
      1. Set toDecimal = number
   k) Else
      1. If(inputBase < 10)
         Then
         i.  Set toDecimal = Con_Lt10(number, inputBase)
```

```
    2.  Else
        i.   Set toDecimal = Con_Gt10(hex, inputBase)
        [End of if structure]
    3.  If(hasBadChar = 1) // Bad character in source
        Then
        i.   Exit
        [End of if structure]
    [End of if structure]
l)  If(outputBase = 10)
    Then
    1.  Print "Converted number : " toDecimal
m)  Else
    1.  Set i = 0
    2.  Repeat through step 2.i to step 2.iii while(toDecimal >= 1)
        i.   Set final[i] = toDecimal % outputBase
        ii.  Set number = number / outputBase
        iii. Set i = i + 1
        [End of while loop]
    3.  Set i = i - 1
    4.  Print "Converted number : "
    5.  Repeat through step 5.i to step 5.iii while (i > 0)
        Begin
        i.   If(final[i] > 9) // we need to print some characters
             Then
             1.  Set character = final[i] - 10 + ascii(A)
             2.  Print character
        ii.  Else
             1.  Print final[i]
             [End of if structure]
        iii.  Set i = i - 1
        [End of while loop]
    [End of if structure]
n)  Print "Do you want to continue (y/n) ? "
o)  Read choice
```

```
    [End of while loop]
Step 3:   End
```

Source Code

```
#include <stdio.h> // printf(), scanf()
#include <math.h> // pow()
#include <string.h> // strlen()

// ====================================================
//                              Bookkeeping methods
// ====================================================

// Returns the decimal value of a character
int get_cval(char c, int ib){
    if(c >= '0' && c <= '9') // It is a numeric digit
        return c - '0';
    if(c >= 'a' && c <= 'a' + (ib - 11)) // It is something between a - f
        return c - 'a' + 10;
    if(c >= 'A' && c <= 'A' + (ib - 11)) // It is something between A - F
        return c - 'A' + 10;
    return -1; // It is impossible to have a digit like this in present base!
}

// =======================================================
//                              Conversion methods
// =======================================================

static short hasBadChar = 0; // Flag to denote if received string has an invalid
character

// Convert a number in a base > 10 to base 10
long int con_gt10(char hex[20], long int ib){
    long int len, val, dec = 0, i = 0;
    len = strlen(hex) - 1;
    for(i=0; hex[i]!='\0'; i++){
        val = get_cval(hex[i], ib);
        if(val == -1){
            printf("\n[Error] Bad character %c for input base %ld!", hex[i], ib);
            hasBadChar = 1;
        }
        dec += val * pow(ib, len);
        len--;
    }
    return dec;
}

// Convert a number in a base < 10 to base 10
long int con_lt10(long int n, long int ib){
    long int b[20], i=0, s=0, p;
    while(n >= 1){
        b[i] = n%10;
        n = n/10;
        p = i;
        i++;
    }
    for (i=0; i<=p; i++){
        s += (b[i] * pow(ib,i));
    }
    return s;
}

// ============================================================
//                          Driver
// ============================================================

int main(){
    char ch = 'Y';
    do{
        long int n, a[20], ib, ob, i = 1, p, c;
        char hex[20]; // String to store if the input base is > 10
```

```
printf("\n[Input] Enter the input base : ");
scanf("%ld",&ib);
if(ib < 2 || ib > 16){ // Hey man, behave yourself!
   printf("\n[Error] Input base must be fall in the range [2,16]!");
   return 1;
}

if (ib < 11){ // Thank god! The input is going to be pure decimal
   printf("\n[Input] Enter the number : ");
   scanf("%ld",&n);
}
else { // There is going to be some nasty chars in it
   printf("[Input] Enter the number : ");
   scanf("%s", hex);
}

printf("\n[Input] Enter the output base : ");
scanf("%ld",&ob);

if(ob < 2 || ob > 16){ // Whoops!
   printf("\n[Error] Output base must fall in the range [2,16]!");
}

if(ib != 10){ // We can't really convert a decimal to decimal, can we?
   if(ib < 10)
      c = con_lt10(n, ib);// A function to convert all < 10 bases to decimal
   else{
      c = con_gt10(hex, ib); // A function to convert all > 10 bases to
                                    // decimal

      if(hasBadChar == 1) // Don't play dumb with me!
         return 1;
   }
```

```
        }
        else
            c = n;

        if(ob == 10){ // We have the converted decimal by now
            printf("\n[Output] Converted number : %ld",c);
        }
        else { // Converting to the output base, when will this day end?

            // You know this, right?
            while(c >= 1){
                a[i] = c%ob;
                c = c/ob;
                p = i;
                i++;
            }

            printf("\n[Output] Converted number : ");

            for (i=p;i>=1;i--){ // Let's desugar it
                if(a[i] > 9) // Duh! I can handle it too!
                    printf("%c", (char)(a[i] + 87));
                else
                    printf("%ld",a[i]);
            }

        }

        printf("\n[Input] Do you want to continue (y/n) ? ");
        scanf(" %c",&ch);

    } while(ch=='y' || ch=='Y'); // I can keep it up, can you?
    return 0;
}
```

Input and Output

Set 1

```
[Input] Enter the input base : 2
[Input] Enter the number : 101011
[Input] Enter the output base : 10
[Output] Converted number : 43
[Input] Do you want to continue (y/n) ? n
```

Set 2

```
[Input] Enter the input base : 16
[Input] Enter the number : qwe12
[Input] Enter the output base : 10
[Error] Bad character q for input base 16!
[Error] Bad character w for input base 16!
```

Set 3

```
[Input] Enter the input base : -2
[Error] Input base must be fall in the range [2,16]!
```

Set 4

```
[Input] Enter the input base : 16
[Input] Enter the number : ffff
[Input] Enter the output base : 10
[Output] Converted number : 65535
[Input] Do you want to continue (y/n) ? n
```

Discussion

1. To convert a number from one base to another one must enter two different bases.
2. If the entered bases are same then the output will be same as like the input. So for better programming one must check at first whether the bases are same or not.
3. If one enters the negative number for bases and also for number of that corresponding base, then user cannot get the desired output. So to get the desired output one must enter the positive number for bases and for numbers.