```
#include<iostream>
#include <string.h>

using namespace std;

unsigned int ValueIn = 0;
char EntryStr[100];
char Output[200];
char HexStr[17] = {'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};

void ConvertToHex();
void ConvertToDec();
void ConvertToOct();
void ConvertToBin();
void ConvertEntry();

int main ( int argc, char **argv )
{

    cout << "Input Unsigned 32 bit Integer: ";
    cin >> EntryStr;                                                    //Enter Number
String

    ConvertEntry();                                                     //Convert to a Int
    cout << "String Conversion Value = " << ValueIn << endl;

    ConvertToHex();                                                     //Convert to Hex
String
    cout << "Hex = " << Output<<endl;
    ConvertToOct();                                                     //Convert to
Octal String
    cout << "Octal = " << Output << endl;
    ConvertToBin();                                                     //Convert to
Binary String
    cout << "Binary = " << Output << endl;
    ConvertToDec();                                                     //Conver to Dec
String
    cout << "Decimal = " << Output << endl;

    return 0;
}


void ConvertEntry()
{
    __asm__ (
        "movl $EntryStr,%esi;"
        "mov $0, %eax;"             // initialize the accumulator
        "mov $10, %ecx;"
        "ce:;"
        "mov $0, %ebx;"             // clear all the bits in EBX
        "mov (%esi), %bl;"          // load next character in BL
        "inc %esi;"                 // and advance source index

        "cmp $'0', %bl;"            // does character preceed '0'?
        "jb  ce1;"                  // yes, it's not a numeral jb:jump below
        "cmp $'9', %bl;"            // does character follow '9'?
        "ja  ce1;"                  // yes, it's not a numeral ja:jump above

        "sub $'0', %bl;"            // else convert numeral to int
        "mull %ecx;"                // multiply accumulator by ten. %eax * 10
        "add %ebx, %eax;"           // and then add the new integer
```

```
        "jmp ce;"                   // go back for another numeral
        "ce1:;"
        "movl %eax, ValueIn;"       //Move Value to ValueIn
        );

}


void ConvertToOct()
{
    __asm__ (
        "movl $Output+11, %ecx;"    //point to end of output string needed
        "movl $0,(%ecx);"           //null term
        "subl $1,%ecx;"

        "movl ValueIn,%ebx;"        //Load EBX with value
        "co: ;"                     //loop start
        "movl %ebx,%eax;"           //move working value to EAX
        "andl $7,%eax;"             //Xor to get lowest byte value...
        "add $48,%eax;"             //ascii adjust
        "mov %al,(%ecx);"           //store number...

        "shr $3,%ebx;"              //shift working value right for next byte

        "dec %ecx;"                 //dec str pointer
        "cmpl $Output, %ecx;"       //at beggining?
        "jge co;"                   //loop if not
    );

}



void ConvertToHex()
{
    __asm__ (
        "movl $Output+8, %ecx;"     //point to end of output string needed
        "movl $0,(%ecx);"           //null term
        "subl $1,%ecx;"

        "movl ValueIn,%ebx;"        //Load EBX with value
        "ch: ;"                     //loop start
        "movl %ebx,%eax;"           //move working value to EAX
        "andl $0xF,%eax;"           //xor to get lowest byte value...
        "mov HexStr(%eax), %al;"
        "mov %al,(%ecx);"           //store number...

        "shr $4,%ebx;"              //shift working value right for next byte

        "dec %ecx;"                 //dec str pointer
        "cmpl $Output, %ecx;"       //at beggining?
        "jge ch;"                   //loop if not
    );
}



void ConvertToDec()
{
    __asm__ (
```

```
        "movl $1000000000,%ebx;"    //load divisor...
        "movl $Output, %ecx;"       //point to output string
        "movl ValueIn,%eax;"        //Load EAX with value
        "cd: ;"                     //loop start
        "xorl %edx,%edx;"           //clear things
        "divl %ebx;"                //eax = quotient, edx = remainder
        "add $48,%eax;"             //ascii adjust
        "mov %al,(%ecx);"           //store number...
        "inc %ecx;"                 //inc string ptr
        "cd2:;"
        "movl %edx,%edi;"           //store remainer
        "movl %ebx,%eax;"           //mov divisor for divide
        "movl $10,%ebx;"            //setup divide
        "xorl %edx,%edx;"           //clear things
        "divl %ebx;"                //reduce divisor
        "movl %eax,%ebx;"           //get divsor
        "movl %edi,%eax;"           //restore remainer

        "cmpl $0,%ebx;"             //have we gone thru whole thing?
        "jg cd;"                    //loop if not

        "movl $0,(%ecx);"           //null term
    );
}



void ConvertToBin()
{
    __asm__ (
        "movl $0x80000000,%ebx;"    //load divisor...
        "movl $Output, %ecx;"       //point to output string

        "movl ValueIn,%eax;"        //Load EAX with value
        "cb: ;"                     //loop start
        "xorl %edx,%edx;"           //clear things
        "divl %ebx;"                //eax = quotient, edx = remainder
        "add $48,%eax;"             //ascii adjust
        "mov %al,(%ecx);"           //store number...
        "inc %ecx;"                 //inc String Pointer
        "movl %edx,%eax;"           //get remainer
        "rorl $1,%ebx;"             //rotate divisor
        "cmpl $0x80000000,%ebx;"    //have we gone thru whole thing?
        "jne cb;"                   //loop if not
        "movl $0,(%ecx);"           //null term
    );
}
```