Watch this 14 minutes long YouTube (Pre-Algebra 3- Decimal, Binary, Octal and Hexadecimal) (at https://www.youtube.com/watch?v=5sS7w-CMHkU) which explains the different numbers system very well! Afterwards, practice converting numbers between different bases.

(a) \((89)_{10} = (?)_2\), i.e., converting 89 in decimal to binary

(b) \((534)_8 = (?)_{10}\) i.e., from octal to decimal

(c) \((534)_8 = (?)_2\) note that you can just write every octal digit in 3 binary digits for example \((755)_8 = (111,101,101)_2\), as 7 is 111 in binary, 5 is 101 in binary.

(d) \((1A3)_{16} = (?)_{10}\)

(e) \((1A3)_{16} = (?)_2\) Note that you can just write every hexadecimal digit using 4 binary digits.

(f) \((101001110)_2 = (?)_8\) Note that you can just group every 3 binary digits, write them in an octal digit.
Recall the sizeof function returns the size of a variable (i.e., the number of bytes used by a variable). Write a C++ program that declares three variables and three pointers that pointing to them (as below), and add cout statements in order to find out the size, and address of the variables. Run your program and write down your answers in the blank.

Note when you cout a pointer variable or value, it will be shown in hexadecimal, as indicated by the 0X prefix of the number. You can cast the pointer to long if you want the address to be displayed in decimal.

```cpp
int a = 10; // the size of a is: _______
double d = 3.1415; // the size of d is: _______
char c = 'A'; // the size of c is: _______
int * p = NULL; // the size of p is: _______
   // the value of p is: __________
double * q = NULL; // the size of q is: _______
   // the value of q is: __________
char * t = NULL; // the size of t is: _______, the value of t is: __________
p = &a; // the value of p is __________
q = &d; // the value of q is __________
t = &c; // the value of c is __________
cout << *p << endl; // This displays ______. When accessing *p, the following ___ bytes are
   // accessed: ________________________________ (give a range of memory)

cout << *q << endl; // This displays ______. When accessing *q, the following ___ bytes are
   // accessed: ________________________________ (give a range of memory)

cout << *t << endl; // This displays ______. When accessing *t, the following ___ bytes are
   // accessed: ________________________________ (give a range of memory)

cout << "p+1=" << p + 1 << endl; // How much is p differs from p+1? ______
cout << "q+1=" << q + 1 << endl; // How much is q differs from q+1? ______
```
cout << "t+1=" << t+1 << endl; // How much is t differs from t+1? ______

// We observe that all pointer variables are of the same size, but we need to differentiate // a pointer to an int against a pointer to a double, because:

3  For the following code, draw the diagram (follow the example in the book) to illustrate variables and poiters after the statements marked with numbers.

    int a=100;
    int *p=&a; // (1)

    int *q= new int; // (2)

    *q = 3; // (3)

    *p = *q; // (4)

4  Answer the following questions about dynamic arrays
    /* A function that reads a sequence of integers from input (with the length of sequence, followed by the numbers), and saves the numbers in an array of the given length
    For example, if the length is 3, and the numbers are
    123 345 99
    then the array returned will be of size 3, and stores values 123, 345, and 99
    @param length: upon return, stores the length/size of the array
    @return the pointer pointing to the array
    */
    int * ReadNumberSequence (int & size)  
    {  
        int * array = NULL; // initialize the pointer to NULL
do {
    cout <<"Enter the length of the number sequence:";
    cin >> size;
} while (size<=0);

// Todo: Write a statement to allocate memory for the array.
// Note: we only know the value of size at run time, so we need to DYNAMICALLY
// allocate memory for this array

// Todo: write a loop to read size # of int from input, and save them to the array

    return array;
}

int main()
{
    // Todo: delcare necessary variables

    //Todo: call the ReadNumberSequence function to read a sequence of numbers

    //Todo: write a loop to display the elements in the array returned ...

    //Todo: free the array returned by ReadNumberSequence.

}