The main element of this assignment is to develop an algorithm and implement it in Java for the following problem:

You are given a base $B$, and the description of a number $m$ represented in that base as "$m=(X.Y)_B$" where $X$ is the integer part of $m$ and $Y$ is the fractional part of $m$. Both parts are provided as arrays of numbers base $B$, so $X[]$ is an array of size $X.length$ and $Y[]$ an array of size $Y.length$. Each $X[i], Y[i]$ is a number among $\{0,1,\ldots,B-1\}$.

The input is $(B,X[],Y[]), R$ where $R$ is the base in which number $m=(X.Y)_B$ is to be represented. In base $R$, $m$ will be represented in the format $m=(U.VW)_R$ where $U$ is the integer part of $m$ and $VW$ is the fractional part of $m$. The $W$ part is used to represent the fractional part as an infinitely repeating pattern following a fixed non-repeating pattern $V$. We restrict the bases $B$ and $R$ to be $2 \leq B, R \leq 60$. Note that the length of $W$ is at most $B^{\text{length}(Y)}$.

For example, in bases $B=10$ and $R=2$, the number $5/2 = (2.5)_{10}$ would be represented by $(10.1)_2$ because $(101/10)_2$ yields the pattern "1.01" when literally dividing $(101)_2$ by $(10)_2$. In the proposed format we obtain $(10.1)_2=(U.VW)_2$ where $U=(10)_2, V=(1)_2, W=(0)_2$ since indeed $(10.1)_2$ is the same as $(10.10)_2$. The number $1/5 = (0.2)_{10}$ would be represented by $(0.0011)_2$ because $(1/101)_2$ yields the infinite pattern $0.001100110011\ldots$ In the proposed format we obtain $(0.0011)_2=(U.VW)_2$ where $U=(0)_2, V=(0)_2, W=(0011)_2$. The answer in base $R$ is not unique since many representations are possible for the same number. For instance, all of the following are equivalent

$$(0.5)_{10} = (0.49)_{10}$$
$$(0.1)_{2} = (0.01)_{2}$$
$$(0.0011)_{2} = (0.0011001100110011\ldots)_{2}$$

The output is going to be of the format $U[],V[],W[]$. Any valid representation of the input number will be accepted. To simplify the input/output I have defined Java objects Number that contains a
Base and three arrays U[], V[], W[]. Your method should be called "public Number convert(Number A, short Base)". Please see the tester code I made available on the course web page and next page.

**Question:**
Are we allowed to use all the regular mathematical operations (+, -, *, /) in HW-1 or do we have to make our own algorithms?

**Answer:**
You can use "+ , -, *, /" for fixed sizes, upto say 16-bit numbers. The input arrays will be of type short[]. Since we only request to handle bases upto 60, each digit will fit in a byte. The sum or product of two bytes will fit into a short. Otherwise you definitely need to implement your own " / " for arbitrarily long numbers which involves " - " and at least " * " of a single digit by arbitrarily long numbers, which involves " + ".

**Question:**
What is an example of input/output?

**Answer:**
Since (19.247)_{10} = (10011.001111111111101101100100010110100001110010111000010011011101001011010011111110)_{2}, it means

- B=10, X=19, Y=247, R=2
- should return
  - U=10011, V=001,
  - W=1111001111101101000101101000001110010100100010110000100110111010010111000110100110001
  - 10101001111110

or more precisely,

- X[1]=1; X[0]=9;
- Y[2]=2; Y[1]=4; Y[0]=7;

and output

- V[2]=0; V[1]=0; V[0]=1;
- W[99]=1; W[98]=1; W[97]=1; W[96]=1; W[95]=1;
package conv;
public class tester {
    public static void main(String[] args) {
        class Number{
            //=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
            // your method for converting belongs here...
            //=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
            public Number convert(Number A, short Base) {
                Number B=new Number();
                B.Base=Base;
                B.Int=A.NonRep; B.NonRep=A.Int; B.Rep=A.NonRep;
                // my code above is just to make sure it compiles and runs
                return B;
            }
            //=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

            public void printShortArray(short[] S) {
                for (int i = S.length-1; i>=0; i--) {
                    System.out.print(S[i]);
                }
            }

            public void printNumber(Number N) {
                System.out.print("(\n");
                N.printShortArray(N.Int);
                System.out.print(\".\n");
                N.printShortArray(N.NonRep);
                System.out.print("\}\_\n");
                System.out.println(N.Base);
            }
        }
        short Base; short[] Int,NonRep,Rep;
    }
    Number N1=new Number() ;
    N1.Int[1]=1; N1.Int[0]=9;
    N1.Rep=new short[0];
    N1.printNumber(N1);

    Number N2=new Number() ;
    short R=2;
    N2=N1.convert(N1,R);
    N2.printNumber(N2);
    }
}