LoanCalc

```
public class LoanCalc {

   static double epsilon = 0.001;
   static int iterationCounter;

   public static void main (String[] args) {

       double loan = Double.parseDouble(args[0]);
       double rate = Double.parseDouble(args[1]);
       int n = Integer.parseInt(args[2]);
       System.out.println("Loan sum = " + loan + ", interest rate = " + rate + "%,
periods = " + n);

       System.out.print("Periodical payment, using brute force: ");
       System.out.printf("%.2f", bruteForceSolver(loan, rate, n, epsilon));
       System.out.println();
       System.out.println("number of iterations: " + iterationCounter);

       System.out.print("Periodical payment, using bi-section search: ");
       System.out.printf("%.2f", bisectionSolver(loan, rate, n, epsilon));
       System.out.println();
       System.out.println("number of iterations: " + iterationCounter);
   }

     public static double bruteForceSolver(double loan, double rate, int n, double
epsilon) {

       double g = loan / n;
       iterationCounter = 0;

       while(endBalance(loan, rate, n, g) >= epsilon) {
           g += epsilon;
           iterationCounter++;
       }

       return g;
   }

      public static double bisectionSolver(double loan, double rate, int n, double
epsilon) {

           double L = loan / n;
           double H = loan;
           double g = (H + L) / 2;
```

```
            iterationCounter = 0;

            while(H - L >= epsilon) {
                if(endBalance(loan, rate, n, g) * endBalance(loan, rate, n, L) > 0)
                    L = g;
                    else
                        H = g;
                g = (H + L) / 2;
                iterationCounter++;
            }
        return g;
    }

      private static double endBalance(double loan, double rate, int n, double
payment) {

        double k = loan;
        for (int i = 0; i < n; i++) {
            k = ((k - payment) * ((100 +rate) / 100));
        }
        return k;
    }
}
```

LowerCase

```java
public class LowerCase {
    public static void main(String[] args) {
        String str = args[0];
        System.out.println(lowerCase(str));
    }

    public static String lowerCase(String s) {
        String newStr = "";
        char c;
        for(int i = 0; i < s.length(); i++) {
            if(s.charAt(i) > 64 && s.charAt(i) < 91) {
                c = (char)(s.charAt(i) + 32);
                newStr = newStr + c ;
            }
            else
                newStr = newStr + (char)(s.charAt(i));
        }

        return newStr;
    }
}
```

UniqueChars

```java
public class UniqueChars {
   public static void main(String[] args) {
       String str = args[0];
       System.out.println(uniqueChars(str));
   }
   public static String uniqueChars(String s) {
       String newStr = "";
       char c;
       for(int i = 0; i < s.length(); i++) {
           c =(char)(s.charAt(i));
           if(c == ' ')
               newStr = newStr + " ";
               else if(newStr.indexOf(c) == -1)
                   newStr = newStr + (char)(s.charAt(i));
       }

       return newStr;
   }
}
```

Calendar

```java
public class Calendar {

   static int dayOfMonth = 1;
   static int month = 1;
   static int year = 1900;
   static int dayOfWeek = 2;
   static int nDaysInMonth = 31;

   public static void main(String[] args) {

       int inYear = Integer.parseInt(args[0]);

       while (year != inYear) {
           advance();
       }

       while(year != (inYear +1)) {
       System.out.print(dayOfMonth + "/" + month + "/" + year);
       if(dayOfWeek == 1) System.out.println(" Sunday");
           else System.out.println();
       advance();
       }

   }

   private static void advance() {

       if(dayOfMonth == nDaysInMonth(month, year)) {
       dayOfMonth = 1;
       if(month == 12) {
               month = 1;
               year++;
           }
           else month++;
       }
       else {
           dayOfMonth++;
       }
       if(dayOfWeek == 7) dayOfWeek =1;
       else dayOfWeek++;
    }

    private static boolean isLeapYear(int year) {
       if(year % 400 == 0)
```

```
            return true;
        else if(year % 4 == 0 && year % 100 == 0)
                return false;
            else if(year % 4 == 0)
                    return true;

    return false;
    }

    private static int nDaysInMonth(int month, int year) {
        switch(month) {
            case 1: return nDaysInMonth;
            case 3: return nDaysInMonth;
            case 4: return nDaysInMonth - 1;
            case 5: return nDaysInMonth;
            case 6: return nDaysInMonth - 1;
            case 7: return nDaysInMonth;
            case 8: return nDaysInMonth;
            case 9: return nDaysInMonth - 1;
            case 10: return nDaysInMonth;
            case 11: return nDaysInMonth - 1;
            case 12: return nDaysInMonth;
            case 2: {
                if(isLeapYear(year)) return nDaysInMonth - 2;
                    else return nDaysInMonth - 3;
                }
            default: return 0;
        }
    }
}
```