```java
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.println(endBalance(loan, rate, n, 9753.60));

       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) {
       iterationCounter = 0;
       double eachPayment = loan / n;
       while (endBalance(loan, rate, n, eachPayment) >= epsilon) {
           eachPayment += epsilon;
           iterationCounter++;
       }
       return eachPayment;
   }

   public static double bisectionSolver(double loan, double rate, int n, double
epsilon) {
       double low = loan / n;
       double high = loan;
       double mid = 0;
```

```
        iterationCounter = 0;

        while (high - low > epsilon) {
            mid = (high + low) / 2;
            double balance = endBalance(loan, rate, n, mid);
            if (Math.abs(balance) <= epsilon) {
                break;
            } else if (balance > 0) {
                low = mid;
            } else {
                high = mid;
            }

            iterationCounter++;
        }
        return mid;
    }

    private static double endBalance(double loan, double rate, int n, double
payment) {
        double prevBal=loan;
        double nextBal=0;
        for (int i=1; i<=n; i++) {
            nextBal = (prevBal - payment)*(1+(rate/100));
            prevBal = nextBal;
        }
        return nextBal;
    }
}
```

```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 newString="";
       for (int i=0; i<s.length(); i++) {
           char letter = s.charAt(i);
           if (s.charAt(i) >= 65 && s.charAt(i) <= 90) {
               newString +=  (char)(letter+32);
           } else {
               newString +=  (char)letter;
           }
       }
       return newString;
   }
}
```

```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 newString = "";
       for (int i = 0; i < s.length(); i++) {
           char letter = s.charAt(i);
           if (letter == ' ' || newString.indexOf(letter) == -1) {
               newString += letter;
           }
       }
       return newString;
   }
}
```

```
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 asdf[]) {
       int yearIn = Integer.parseInt(asdf[0]);
       while (year <= yearIn) {
           advance();
           if (year == yearIn) {
               if (dayOfWeek==1) {
                   System.out.println(dayOfMonth+"/"+month+"/"+year+" Sunday");
               } else System.out.println(dayOfMonth+"/"+month+"/"+year);
           }
       }
    }

   private static void advance() {
       dayOfMonth++;
       dayOfWeek++;
       if (dayOfWeek > 7) dayOfWeek = 1;
       nDaysInMonth = nDaysInMonth(month, year);
       if (dayOfMonth > nDaysInMonth) {
           dayOfMonth = 1;
           month++;
           if (month > 12) {
               month = 1;
               year++;
           }
       }
   }

   public static boolean isLeapYear(int year) {
       return ((year%400==0) || ((year%4==0) && (year%100!=0)));
   }
```

```
    public static int nDaysInMonth(int month, int year) {
        switch(month) {
            case 2: if (isLeapYear(year)) return 29; else return 28;
            case 4: return 30;
            case 6: return 30;
            case 9: return 30;
            case 11: return 30;
            default: return 31;
        }
    }
}
```