```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.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 approximateAnnualPayment = loan/n;

        while (endBalance(loan, rate, n, approximateAnnualPayment) > 0) {
            iterationCounter++;
            approximateAnnualPayment += epsilon;
        }

        return approximateAnnualPayment;
    }


    public static double bisectionSolver(double loan, double rate, int n, double epsilon) {
        iterationCounter = 0;
        double high = loan, low = 0;
        while (high - low > epsilon) {
            if (endBalance(loan, rate, n, (high + low) / 2) < 0) {
                high = (high + low) / 2;
            }
            else {
                low = (high + low) / 2;
            }
```

```
            iterationCounter++;
        }
        return (high + low) / 2;
    }

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

        return loan;
    }
}
```

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

```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 oldStr = s;
        String newStr = "";
        char ch;
        int doesCharExistsNewStr;

        for (int i = 0; i < oldStr.length(); i++) {
            ch = oldStr.charAt(i);
            doesCharExistsNewStr = newStr.indexOf(ch);
            if (doesCharExistsNewStr < 0 || ch == ' ') {
                newStr += ch;
            }
        }

        return newStr;
    }

    public static boolean doesCharMoreThanOnceInStr(String text, char chr) {
        int repeatCharCounter = 0, loopCounter = 0;
        int stringLength = text.length();
        while (repeatCharCounter <= 1 && loopCounter <= stringLength -1) {
            if(text.charAt(loopCounter) == chr) {
                repeatCharCounter++;
            }
            loopCounter++;
        }

        if(repeatCharCounter == 2) {
            return true;
        }
        return false;
    }
}
```

```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 askedYear = Integer.parseInt(args[0]);

        while ((year <= 1999)) {
            if (year == askedYear) {
                if (dayOfWeek % 7 == 0) {
                    System.out.println(dayOfMonth + "/" + month + "/" + year + " Sunday");
                }
                else {
                    System.out.println(dayOfMonth + "/" + month + "/" + year);
                }
            }

            advance();
        }
     }

     private static void advance() {
        int nDaysInMonth = nDaysInMonth(month, year);
        if (dayOfMonth < nDaysInMonth) {
            dayOfMonth++;
        }
        else if (nDaysInMonth == dayOfMonth) {
            dayOfMonth = 1;
            if (month < 12) {
                month++;
            }
            else if (month == 12) {
                month = 1;
                year++;
            }

        }

        dayOfWeek++;
    }

    private static boolean isLeapYear(int year) {
        return year % 4 == 0 ? true : false;

    }
```

```
    private static int nDaysInMonth(int month, int year) {
        if (month == 2) {
            return isLeapYear(year) ? 29 : 28;
        }
        else if (month < 8) {
            return month % 2 == 1 ? 31 : 30;
        }
        else if (month >= 8) {
            return month % 2 == 1 ? 30 : 31;
        }
        return 0;
    }
}
```