```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) {
        double g = loan/n;
        iterationCounter = 0;

        while (endBalance(loan, rate, n, g) > 0) {
            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 =(L+H)/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 endingbalance = loan;
        for(int i=1; i<=n; i++){
            endingbalance = (endingbalance - payment)*(1 +
(rate/100));
        }
        return endingbalance;
    }
}
```

```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 end = "";
        char c;

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

```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 end = "";
        String help = "";
        for(int i=0; i<s.length(); i++){
            if (help.indexOf(s.charAt(i))==-1) {
                if (s.charAt(i)==' ') {
                    end+=s.charAt(i);
                } else{
                    end+=s.charAt(i);
                    help+=s.charAt(i);
                }
            }
        }
        return end;
    }
}
```

```
public class Calendar {

    static int dayOfMonth = 1;
    static int month = 1;
    static int year = 1900;
    static int dayOfWeek = 2;     // 1.1.1900 was a Monday
    static int nDaysInMonth = 31; // Number of days in January

    public static void main(String args[]) {
        int newyear = Integer.parseInt(args[0]);

        while(year!=newyear) {
            for(int j=month; j<=12; j++){
                nDaysInMonth = nDaysInMonth(j, year);
                dayOfMonth = 1;
                for(int k=dayOfMonth; k<=nDaysInMonth; k++){
                    dayOfMonth++;
                    dayOfWeek++;
                    if(dayOfWeek==8) dayOfWeek = 1;
                }
            }
        year++;
        }
        month=1;

                for(int j=month; j<=12; j++){
                    nDaysInMonth = nDaysInMonth(j, year);
                    dayOfMonth = 1;
                    for(int k=dayOfMonth; k<=nDaysInMonth;
k++){
                        if(dayOfWeek==1){
                            System.out.println(dayOfMonth+"/"+
j+"/"+year+" Sunday");
                        }
                        else
System.out.println(dayOfMonth+"/"+j+"/"+year);
                        dayOfMonth++;
                        dayOfWeek++;
                        if(dayOfWeek==8) dayOfWeek = 1;
                    }
                }

     }

    private static boolean isLeapYear(int year) {
        if(year%4==0) return true;
        else return false;
    }
    private static int nDaysInMonth(int month, int year) {
        if(month==4 || month==6 || month==9 || month==11)
return 30;
        if(month==1 || month==3 || month==5 || month==7 ||
month==8 || month==10 || month==12) return 31;
        if(month==2){
            if(isLeapYear(year)==true && year!=1900) return
29;
            else return 28;
        }
        return 0;
    }
}
```