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

```
    }

    public static double bisectionSolver(double loan, double rate, int n, double epsilon) {
        iterationCounter = 0;
        double l = loan / n,
              h = loan,
              g = (l + h) / 2;
        while  ((h - l) > epsilon) {
           if (endBalance(loan, rate, n, g) * endBalance(loan, rate, n, l) > 0)
                 l = g;
              else
                 h = g;
           g = (l + h) / 2;
           iterationCounter++;
        }
       return g;
    }

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

```
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 ans = "";
        for (int i = 0; i < s.length(); i++) {
            char c = s.charAt(i);
            if ('A' <= c && c <='Z') {
                c = (char)(c + 32);
            }
            ans += c;
        }
        return ans;
    }
}
```

```
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 ans = "";
        for (int i = 0; i < s.length(); i++){
            char c = s.charAt(i);
            if (c == 32 || ans.indexOf(c) == -1) {
                ans += c;
            }
        }
        return ans;
    }
}
```

```
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]);
           int debugDaysCounter = 0;
                int sun1 = 0;
                while (year < inYear) {
          advance();
      }
      year = inYear;
      int count = inYear + 1;
      while (year < count) {
      System.out.print(dayOfMonth + "/" + month + "/" + year);
                        if (dayOfWeek == 1) {
                                System.out.print(" Sunday");
                        }
                        System.out.println();
                        advance();
      }
                System.out.println();
                System.out.println("During the 20th century, " + sun1 + " Sundays fell on
                the first day of the month");
          }
        private static void advance() {
                if (dayOfMonth < nDaysInMonth) {
                        dayOfMonth++;
```

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

 private static boolean isLeapYear(int year) {
    if (year % 400 == 0) {
                return true;
        } else if (year % 100 == 0) {
                return false;
        } else if (year % 4 == 0) {
                return true;
        } else {
                return false;
        }
 }

 private static int nDaysInMonth(int month, int year) {
        int feb = 28;
        if (isLeapYear(year) == true) {
                feb = 29;
```

```
            }
            switch (month) {
                case 2:
                return feb;
                case 4:
                return 30;
                case 6:
                return 30;
                case 9:
                return 30;
                case 11:
                return 30;
                default:
                return 31;
            }
        }
}
```