HW3Code – Yitzhak Bar or - 208837278

```
public class Calendar1 {
    // Starting the calendar on 1/1/1900
    static int curMonth;
    static int curDay;
    static int curYear;
    static int endYear;
    static int curDayOfWeek; // ==> (2) 1.1.1900 was a Monday
    static int nDaysInMonth; // num of days at curr month
    static boolean isLeapYear; // true if year is a leap year
    static int nDays; // number of days in the month
    static int countSunday;

    public static void main(String args[]) {
        advance();
    }

    /**
     * This founction print the calender from 1990 - 1999 inclusive.
     */
    public static void advance() {
        curYear = 1900;
        endYear = 1999;
        curDayOfWeek = 2;
        countSunday = 0;
        while (curYear <= endYear) {
            curMonth = 1;
            while (curMonth <= 12) {
                curDay = 1;
                while (curDay <= nDaysInMonth(curMonth, curYear)) {
                    if (curDayOfWeek <= 7) {
                        System.out.print(curDay + "/" + curMonth + "/" + curYear);
                        if ((curDay == 1) && (curDayOfWeek) == 1) {
                            System.out.print(" Sunday");
                            countSunday++;
                            curDay++;
                            curDayOfWeek++;
                        } else {
                            curDay++;
                            curDayOfWeek++;
                        }
                        if (curDayOfWeek > 7) {
                            curDayOfWeek = 1;
                        }
                    }
                    System.out.println();
                }
                curMonth++;
            }
            curYear++;
        }
        System.out.println("During the 20th century, " + countSunday + " Sundays fell on the
first day of the month");
    }
    /**
     * This founction return if the year us leap or common.
     *
     * @param year - represents the year
     * @return - true if the given year is a leap year, false otherwise.
     */
    private static boolean isLeapYear(int year) {
        // check if the year is divisble by 400
        isLeapYear = ((year % 400) == 0);
        // than checks if the year is divisible by 4 and not by 100
        isLeapYear = isLeapYear || ((year % 4) == 0 && (year % 100) != 0);
        return isLeapYear;
    }

    /**
     * Returns the number of days in the given month and year. April, June,
     *
     * @param month - represents the month
     * @param year  - represents the year
     * @return - the number of days in the given month and year
     */
    private static int nDaysInMonth(int curMonth, int curYear) {
        switch (curMonth) {
            case 1, 3, 5, 7, 8, 10, 12: // January, March, May, July, August, October, and
December
                nDays = 31;
                break;
            case 2: // February
                nDays = isLeapYear(curYear) ? 29 : 28;
                break;
            case 4, 6, 9, 11: // April, June, September, and November
                nDays = 30;
                break;
            default:
                nDays = 0;
                System.out.println("Invalid month");
                break;
        }
        return nDays;
    }
}
```

```
public class LoanCalc {
// test
    // declare some few class static variables
    static double epsilon; // The computation tolerance (estimation error)
    static int iterationCounter; // Monitors the efficiency of the calculation
    static int n; // number of periods
    static double g; // periodical payment
    static double loan; // sum of the loan
    static double rate; // periodical interest rate (as a percentage)
    static double payment; // periodical payment
    static double endBalance; // ending balance of the loan
    static double low; // lower bound of the periodical payment
    static double high; // upper bound of the periodical payment

    /**
     * Uses a sequential search method ("brute force") to compute an approximation
     * of the periodical payment that will bring the ending balance
     * of a loan close to 0.
     * Given: the sum of the loan (loan), the periodical interest rate (rate),
     * the number of periods (n), and epsilon (epsilon), a tolerance level.
     */
    // Side effect: modifies the class variable iterationCounter.
    public static double bruteForceSolver(double loan, double rate, int n, double epsilon) {
        g = loan / n;
        iterationCounter = 0;
        while (endBalance(loan, rate, n, g) > epsilon) {
            g += epsilon;
            iterationCounter++;
        }
        return g;
    }

    /**
     * Uses bisection search to compute an approximation of the periodical payment
     * that will bring the ending balance of a loan close to 0.
     * Given: the sum of theloan, the periodical interest rate (as a percentage),
     * the number of periods (n), and epsilon, a tolerance level.
     */
    public static double bisectionSolver(double loan, double rate, int n, double epsilon) {
        iterationCounter = 0;
        low = loan / n;
        high = loan;
        g = (low + high) / 2;
        // logic of the bisection search
        while ((high - low) > epsilon) {
            if (endBalance(loan, rate, n, g) * endBalance(loan, rate, n, low) > 0) {
                low = g;
            } else {
                high = g;
            }
            g = (low + high) / 2;
            iterationCounter++;
        }
        return g;
    }
    /**
     * Computes the ending balance of a loan,
     * given:
     * 1.the sum of the loan (loan).
     * 2.the periodical interest rate as a percentage (rate).
     * 3.the number of periods (n).
     * 4.the periodical payment (payment).
     *
     * @return - the ending balance of the loan.
     */
    private static double endBalance(double loan, double rate, int n, double payment) {
        endBalance = loan;
        for (int i = 0; i < n; i++) {
            endBalance = (endBalance - payment) * (1 + rate / 100);
        }
        return endBalance;
    }

    /**
     * Gets the "loan data" and computes the periodical payment.
     * Expects to get three command-line arguments:
     * 1.sum of the loan (double).
     * 2.interest rate as a percentage(double).
     * 3.number of payments (int).
     */
    public static void main(String[] args) {
        // Gets the loan data
        loan = Double.parseDouble(args[0]); // 100,000
        rate = Double.parseDouble(args[1]); // 5.0
        n = Integer.parseInt(args[2]); // 10

        epsilon = 0.001;
        payment = 10000;
        // Prints the loan data
        System.out.println("Loan sum = " + loan + ", interest rate = " + rate + "%, periods = "
+ n);

        // Computes the periodical payment using brute force search
        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);

        // Computes the periodical payment using bisection search
        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 class LowerCase {
    // declare variables
    static String str1, str2, tempStr;
    static char isTemp;
    static int i;
    public static void lowerCase(String str1) {
        for (i = 0; i < str1.length(); i++) {
            isTemp = str1.charAt(i);
            tempStr = "";
            tempStr += isTemp;
            // call function that convert char from upperCase to lowerCase
            // if true, she add lower case to str2 + break with add 1 to i
            // if false, add to str2 ==> than break with add 1 to i
            convertTolower(tempStr);
            // after finishing loop, str2 = different value
        }
    }
    public static void convertTolower(String tempStr) {
        switch (tempStr) {
            case "A":
                str2 += "a";
                break;
            case "B":
                str2 += "b";
                break;
            case "C":
                str2 += "c";
                break;
            case "D":
                str2 += "d";
                break;
            case "E":
                str2 += "e";
                break;
            case "F":
                str2 += "f";
                break;
            case "G":
                str2 += "g";
                break;
            case "H":
                str2 += "h";
                break;
            case "I":
                str2 += "i";
                break;
            case "J":
                str2 += "j";
                break;
            case "K":
                str2 += "k";
                break;
            case "L":
                str2 += "l";
                break;
            case "M":
                str2 += "m";
                break;
            case "N":
                str2 += "n";
                break;
            case "O":
                str2 += "o";
                break;
            case "P":
                str2 += "p";
                break;
            case "Q":
                str2 += "q";
                break;
            case "R":
                str2 += "r";
                break;
            case "S":
                str2 += "s";
                break;
            case "T":
                str2 += "t";
                break;
            case "U":
                str2 += "u";
                break;
            case "V":
                str2 += "v";
                break;
            case "W":
                str2 += "w";
                break;
            case "X":
                str2 += "x";
                break;
            case "Y":
                str2 += "y";
                break;
            case "Z":
                str2 += "z";
                break;
            default:
                str2 += tempStr;
                break;
        }
    }

    public static void main(String[] args) {
        str1 = args[0];
        str2 = "";
        lowerCase(str1);
        System.out.println(str2);
    }
}
```

```java
public class UniqueChars {
    // declare variables
    static String str1, str2;
    public static void main(String[] args) {
        str1 = args[0];
        str2 = "";
        uniqueChars();
        System.out.println(str2);
    }

    public static void uniqueChars() {
        for (int i = 0; i < str1.length(); i++) {
            int j = 0;
            boolean flag = true;
            if (i == 0) {
                str2 += str1.charAt(i);
            }
            if (i > 0) {
                if (str1.charAt(i) == ' ') {
                    str2 += str1.charAt(i);
                } else {
                    // check if the character is already in the string
                    while (j < i && j < str2.length()) {
                        // if the character is already in the string, set flag to false
                        if (str1.charAt(i) == str2.charAt(j) ) {
                            flag = false;
                        }
                        j++;
                    }
                    // if the character is not in the string, add it to the string
                    if (flag) {
                        str2 += str1.charAt(i);
                    }
                }
            }
        }
    }
}
```