Loan Calc

```
    public static double bruteForceSolver(double loan, double rate,
                                           int n, double epsilon) {
        double l = loan;
        double r = rate;
        double e = epsilon;
        double p = l/n;
        boolean b  = false;
        while (!b) {
            l = loan;
            l = endBalance(l, r, n, p);
            if (l > e){
                p += e;
            } else if (l < -e) {
                p -= e/2;
            } else {
                b = true;
            }
            iterationCounter++;
        }
        return p;
    }


    public static double bisectionSolver(double loan, double rate, int
                                          n, double epsilon) {
        iterationCounter = 0;
        double l = loan;
        double r = rate;
        double e = epsilon;
        boolean b  = false;
        Double low = (l/n);
        Double high = (l/n)*Math.pow(1+r/100,n -1);
```

```
        double p = (low + high) / 2;
        double temp = 0;
        while (!b) {
            l = loan;
            l = endBalance(l, r, n, p);
            if (l < (0 - e)) {
                high = p;
                p = (low + temp) / 2;
            } else if (l > e) {
                low = p;
                p = (temp + high) / 2;
            } else {
                b = true;
            }
            iterationCounter++;
        }
        return p;
    }


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

Lowercase

```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 lower = "";
        for (int i = 0; i < s.length(); i++) {
            if (s.charAt(i) <= 91 && s.charAt(i) >= 65) {
                lower += (char)(s.charAt(i) + 32);
            } else {
                lower += s.charAt(i);
            }
        }
        return lower;
    }
}
```

UniqueChars

```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 u = "";
        int cnt = 0;
        u += s.charAt(0);
        for (int i = 0; i < s.length(); i++) {
            cnt = 0;
            if (s.charAt(i) == 32) {
                u += (char)(s.charAt(i));
            } else {
                for (int j = 0; j < u.length(); j++) {
                    if ((s.charAt(i)) == u.charAt(j)) {
                        cnt++;
                    }
                }
                if (cnt == 0) {
                    u += (char)(s.charAt(i));
                }
            }
        }
        return u;
    }
}
```

Calendar

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

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

```
            month = 1;
        }
        if (dayOfWeek > 7) {
            dayOfWeek = 1;
        }
    }

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


    private static int nDaysInMonth(int month, int year) {
        int d = 0;
        if (month == 2) {
            if (isLeapYear(year)) {
                d = 29;
            } else {
                d = 28;
            }
        } else if (month == 4 || month == 6 || month == 9 ||
                    month == 11) {
            d = 30;
        } else {
            d = 31;
        }
        return d;
    }
}
```