Homework: Loops, Methods, Classes

This document defines homework assignments from the "Java Basics" Course @ Software University. Please submit as homework a single **zip** / **rar** / **7z** archive holding the solutions (source code) of all below described problems.

Problem 1. Symmetric Numbers in Range

Write a program to generate and **print all symmetric numbers in given range** [**start**...**end**] (0 ≤ **start** ≤ **end** ≤ 999). A number is symmetric if its digits are symmetric toward its middle. For example, the numbers 101, 33, 989 and 5 are symmetric, but 102, 34 and 997 are not symmetric. Examples:

Input	Output
5 11	5 6 7 8 9 11
101 110	101
555 777	555 565 575 585 595 606 616 626 636 646 656 666 676 686 696 707 717 727 737 747 757 767 777

Problem 2. Generate 3-Letter Words

Write a program to generate and **print all 3-letter words consisting of given set of characters**. For example if we have the characters '**a**' and '**b**', all possible words will be "**aaa**", "**aab**", "**aba**", "**abb**", "**baa**", "**bab**", "**bba**" and "**bbb**". The input characters are given as string at the first line of the input. Input characters are **unique** (there are no repeating characters). Examples:

Input	Output
x	xxx
ab	aaa aab aba abb baa bab bba bbb
abx	aaa aab aax aba abb abx axa axb axx baa bab bax bba bbb bbx bxa bxb bxx xaa xab xax xba xbb xbx xxa xxb xxx

Problem 3. Full House

In most Poker games, the "**full house**" hand is defined as three cards of the same face + two cards of the same face, other than the first, regardless of the card's suits. The cards faces are "**2**", "**3**", "**4**", "**5**", "**6**", "**7**", "**8**", "**9**", "**10**", "**J**", "**Q**", "**K**" and "**A**". The card suits are "♣", "♦", "♥" and "♠". Write a program to generate and print all full houses and print their number. Example:

Output
(2♣ 2♦ 2♥ 3♣ 3♦) … (2♣ 2♦ 2♥ 3♣ 3♦) … (2♣ 2♦ 2♥ 3♣ 3♥) … (A♠ A♦ A♥ K♠ K♦) … 3744 full houses

Problem 4. ** Full House with Jokers

In most Poker games, the "**full house**" hand is defined as three cards of the same face + two cards of the same face, other than the first, regardless of the card's suits. The cards faces are "**2**", "**3**", "**4**", "**5**", "**6**", "**7**", "**8**", "**9**", "**10**", "**J**", "**Q**", "**K**" and "**A**". The card suits are "♣", "♦", "♥" and "♠". A special card "**Joker**" (denoted as "*") is used as **wildcard** and can replace any other card. Jokers do not have a suite. Jokes can be used several times in a hand. Write a program to generate and print all full houses and print their number. Example:

Output
(2♣ 2♦ 2♥ 3♣ 3♦) … (2♣ 2♦ 2♥ 3♣ 3♦) … (2♣ 2♦ 2♥ 3♣ 3♥) … (2♣ 2♦ 2♥ 3♣ *) … (2♣ * * 3♣ *) … (A♠ A♥ A♦ K♠ K♣) … (A♦ A♥ A♠ * *) … (* * * * *) 119808 full houses

Problem 5. Angle Unit Converter (Degrees ↔ Radians)

Write a **method** to **convert from degrees to radians**. Write a **method** to **convert from radians to degrees**. You are given a number **n** and **n** queries for conversion. Each conversion **query** will consist of a **number** + space + **measure**. Measures are "**deg**" and "**rad**". Convert all radians to degrees and all degrees to radians. Print the results as **n** lines, each holding a number + space + measure. Format all numbers with 6 digit after the decimal point. Examples:

Input	Output
3 180 deg 90 deg 3 rad	3.141593 rad 1.570796rad 171.887339 deg

Input	Output
2 3.141592 rad 5.5 rad	179.999963 315.126787

Input	Output
4 0 rad 120 deg 1.55 rad 2.1 rad	0.000000 2.094395 88.808458 120.321137

Problem 6. Random Hands of 5 Cards

Write a program to generate **n random hands of 5 different cards** form a standard suit of 52 cards. Examples:

Input	Output
5	Q♣ J♦ 6♠ 6♣ A♥ 4♦ 7♣ 8♦ 9♣ 3♦ 10♣ 8♥ 10♥ A♣ Q♥ 2♥ 2♠ 2♣ 8♠ J♦ J♣ 10♦ J♠ A♠ K♥

Input	Output
3	10♠ 7♣ A♥ 3♣ A♦ 2♦ 6♦ 10♣ 5♦ 5♣ J♥ A♣ 6♥ 6♦ J♣

Problem 7. Days between Two Dates

Write a program to calculate the **difference between two dates** in number of days. The dates are entered at two consecutive lines in format **day-month-year**. Days are in range [1…31]. Months are in range [1…12]. Years are in range [1900…2100]. Examples:

Input	Output
1-01-2014 2-01-2014	1

Input	Output
28-02-2000 8-03-2000	9

Input	Output
30-11-2014 27-03-2015	117

Input	Output
14-05-2014 14-06-1980	-12387

Problem 8. Sum Numbers from a Text File

Write a program to read a text file "**Input.txt**" holding a sequence of integer numbers, each at a separate line. Print the **sum of the numbers** at the console. Ensure you close correctly the file in case of exception or in case of normal execution. In case of exception (e.g. the file is missing), print "**Error**" as a result. Examples:

Input.txt	Output
3 5 -1	7

Input.txt	Output
100 200	300

Input.txt	Output
(missing file)	Error

Problem 9. List of Products

Create a class **Product** to hold products, which have **name** (string) and **price** (decimal number). Read from a text file named "**Input.txt**" a **list of products**. Each product will be in format **name** + space + **price**. You should hold the products in objects of class **Product**. **Sort** the products **by price** and write them in format **price** + space + **name** in a file named "**Output.txt**". Ensure you close correctly all used resources. Examples:

Input.txt	Output.txt
milk 2.80 apple 1.20 coffee 8.50	1.20 apple 2.80 milk 8.50 coffee

Input.txt	Output.txt
juice 2.50 water 1.20 vodka 18.70 beer 1.12	1.12 beer 1.20 water 2.50 juice 18.70 vodka

Problem 10.* Order of Products

Create a class **Product** to hold products, which have **name** (string) and **price** (decimal number). Read from a text file named "**Products.txt**" a **list of product with prices** and keep them in a list of products. Each product will be in format **name** + **space** + **price**. You should hold the products in objects of class **Product**. Read from a text file named "**Order.txt**" an order of products with quantities. Each order line will be in format **product** + space + **quantity**. Calculate and print in a text file "**Output.txt**" the **total price** of the order. Ensure you close correctly all used resources. Examples:

Products.txt	Order.txt	Output.txt
milk 1.80 apple 3.20 coffee 8.50	12 milk 3.2 coffee 2 coffee 1.5 apple	70.5

Input.txt		Output.txt
juice 2.50 water 1.20 vodka 18.70 beer 1.12	15 water 2 vodka 3 juice 1 water	64.1

Problem 11.*** Excel

You are given an Excel file **Incomes-Report.xlsx** holding an incomes report in the following format:

	A	B	C	D	E	F
1	Office	Date	Description	Income	VAT	Total Incomes
2	Sofia-Rakovski	07-02-2014	Payment by contract #343	960.00	192.00	**1152.00**
3	Bourgas	08-03-2014	Sold a license	240.00	48.00	**288.00**
4	Pleven	12-03-2014	Outsorcing work	4200.00	840.00	**5040.00**
5	Plovdiv	14-03-2014	Sold 3 license	600.00	120.00	**720.00**
6	Sofia-Vitosha	22-03-2014	Consulting a customer	3500.00	700.00	**4200.00**
7	Plovdiv	24-03-2014	Sold a license	276.00	55.20	**331.20**
8	Sofia-Vitosha	26-03-2014	Consulting services	560.00	112.00	**672.00**
9	Varna-central	28-03-2014	Sold a license	870.00	174.00	**1044.00**
10	Bourgas	09-04-2014	Sold a demo license	40.00	8.00	**48.00**
11	Pleven	31-05-2014	Consulting	140.00	28.00	**168.00**
12	Pleven	11-07-2014	Sold a demo license	40.00	8.00	**48.00**
13						

Each office puts in this Excel file all their incomes (office, date, description, income, 20% VAT, total income). Your task is to **read the report** and to calculate the **incomes sub-totals for each office** (with VAT). Order the offices alphabetically. Print the result at the console in format **town** Total -> **incomes**. Finally calculate and print the grand total (the sum of all incomes in all offices). Sample output (for the above report):

Output
Bourgas Total -> 336.00 Pleven Total -> 5256.00 Plovdiv Total -> 1051.20 Sofia-Rakovski Total -> 1152.00 Sofia-Vitosha Total -> 4872.00 Varna-central Total -> 1044.00 Grand Total -> 13711.20

You are free to use a Java library of choice to open and read Excel spreadsheets (**.xlsx** files).