Homework: Functional Programming

Problem 1. Action Print

Write a program that reads a collection of strings from the console and then prints them onto the console. Each name should be printed on a new line. Use **Action<T>**.

Input	Output
Pesho Gosho Adasha	Pesho Gosho Adasha

Problem 2. Knights of Honor

Write a program that reads a collection of names as strings from the console then appends "Sir" in front of every name and prints it back onto the console. Use **Action<T>**.

Input	Output
Pesho Gosho Adasha StanleyRoyce	Sir Pesho Sir Gosho Sir Adasha Sir StanleyRoyce

Problem 3. Custom Min Function

Write a simple program that reads from the console a set of numbers and prints back onto the console the smallest number from the collection. Use **Func<T, T>**.

Input	Output
1 4 3 2 1 7 13	1

Problem 4. Find Evens or Odds

You are given a lower and an upper bound for a range of integer numbers. Then a command specifies if you need to list all even or odd numbers in the given range. Use **Predicate<T>**.

Input	Output
1 10 odd	1 3 5 7 9
20 30	20 22 24 26 28 30

even	

Problem 5. Applied Arithmetics

Write a program that executes some mathematical operations on a given collection. On the first line you are given a list of numbers. On the next lines you are passed different commands that you need to apply to all numbers in the list: "add" -> add 1 to each number; "multiply" -> multiply each number by 2; "subtract" -> subtract 1 from each number; “print” -> print the collection. The input will end with and "end" command, after which you need to print the result. Use functions.

Input	Output
1 2 3 4 5 add add print end	3 4 5 6 7
5 10 multiply subtract print end	9 19

Problem 6. Reverse and exclude

Write a program that reverses a collection and removes elements that are divisible by a given integer **n**. Use predicates/functions.

Input	Output
1 2 3 4 5 6 2	5 3 1
20 10 40 30 60 50 3	50 40 10 20

Problem 7. Predicate for names

Write a program that filters a list of names according to their length. On the first line you will be given integer **n** representing name length. On the second line you will be given some names as strings separated by space. Write a function that prints only the names whose length is **less than or equal** to **n**.

Input	Output
4 Kurnelia Qnaki Geo Muk Ivan	Geo Muk Ivan
4	Asen

Karaman Asen Kiril Yordan	

Problem 8. Custom Comparator

Write a custom comparator that sorts all even numbers before all odd ones in ascending order. Pass it to an Array.sort() function and print the result.

Input	Output
1 2 3 4 5 6	2 4 6 1 3 5
-3 2	2 -3

Problem 9. List of Predicates

Find all numbers in the range 1..N that are divisible by the numbers of a given sequence. Use predicates/functions.

Input	Output
10 1 1 1 2	0 2 4 6 8 10
100 2 5 10 20	0 20 40 60 80 100

Problem 10. Predicate Party!

Ivancho's parents are on a vacation for the holidays and he is planning an epic party at home. Unfortunately, his organizational skills are next to non-existent so you are given the task to help him with the reservations.

On the first line you get a list with all the people that are coming. On the next lines, until you get the "Party!" command, you may be asked to double or remove all the people that apply to given criteria. There are three different criteria are: 1. everyone that has a name starting with a given string; 2. everyone that has a name ending with a given string; 3. everyone that has a name with a given length. See the examples below:

Input	Output
Pesho Misho Stefan Remove StartsWith P Double Length 5 Party!	Misho, Misho, Stefan are going to the party!
Pesho Double StartsWith Pesh Double EndsWith esho Party!	Pesho, Pesho, Pesho, Pesho are going to the party!

Problem 11. *Party Reservation Filter Module

You are a young and talented developer. The first task you need to do is to implement a filtering module to a party reservation software. First, The Party Reservation Filter Module (TPRF Module for short) is passed a list with invitations. Next the TPRF receives a sequence of commands that specify if you need to add or remove a given filter.

TPRF Commands are in the given format **{command;filter type;filter parameter}**

You can receive the following TPRF commands: "Add filter", "Remove filter" or "Print". The possible TPRF filter types are: "Starts with", "Ends with", "Length" and "Contains". All TPRF filter parameters will be a string (or an integer for the length filter).

The input will end with a "Print" command after which you should print all the party-goers that are left after the filtration. See the examples below:

Input	Output
Pesho Misho Slav Add filter;Starts with;P Add filter;Starts with;M Print	Slav
Pesho Misho Jica Add filter;Starts with;P Add filter;Starts with;M Remove filter;Starts with;M Print	Misho Jica

Problem 12. *Inferno III

Your game studio's next triple A big-budget-killer app is the Hack and Slash Action RPG Inferno III. The main purpose of the game is well, to hack and slash things. But the secondary purpose is to craft items and recently the main fan base has started complaining that once you craft an item you can't change it. So your next job is to implement a feature for one time reforging an item.

On the first line you are given the gems already inserted in the form of numbers, representing their power. On the next lines, until you receive the "Forge" command, you can receive commands in the following format **{command;filter type;filter parameter}**:

Commands can be: "Exclude", "Reverse" or "Forge". The possible filter types are: "Sum Left", "Sum Right" and "Sum Left Right". All filter parameters will be an integer.

"Exclude" marks a gem for exclusion from the set if it meets a given condition. "Reverse" deletes a previous exclusion.

"Sum Left" tests if a gems power summed with the gem standing to its right gives a certain value. "Sum Right" is the same but looks to a gems right peer. "Sum Left Right" sums the gems power with both its left and right neighbors. If a gem has no neighbor to its right or to its left (first or last element), then simply add 0 to the gem.

Note that changes on to the item are made only after forging. This means that the gems are fixed at their positions and every function occurs on the original set, so every gems power is considered, no matter if it is marked or not.

To better understand the problem, see the examples below:

Input	Output	Comments
1 2 3 4 5 Exclude;Sum Left;1 Exclude;Sum Left Right;9 Forge	2 4	1. Marks for exclusion all gems for which the sum with neighbors to their left equals 1, e.g. 0 + **1** = 1 2. Marks for exclusion all gems for which the sum with neighbors to their left and their right equals 9, e.g. 2 + **3** + 4 = 9 4 + **5** + 0 = 9
1 2 3 4 5 Exclude;Sum Left;1 Reverse;Sum Left;1 Forge	1 2 3 4 5	1. Marks for exclusion all gems for which the sum with their gem peers to the left equals 1, e.g. 0 + 1 = 1 2. Reverses the previous exclusion.

Problem 13.**TriFunction

Write a program that traverses a collection of names and returns the first name whose sum of characters is equal to or larger than a given number **n**. Use a function that accepts another function as one of its parameters. Start off by building a regular function to hold the basic logic of the program something along the lines of **Func<string, int, bool>**. Afterwards create your main function which should accept the first function as one of its parameters.