Assignment Set 3 (Functional Programming with Haskell)

- **Assignments will be evaluated by the TAs.**
- **You should submit report (for answering non-code related questions), complete source codes and executable files.**
- **All codes must be properly documented and good code writing practice should be followed (carry marks).**
- **Copying is strictly prohibited. Any case of copying will automatically result in F for the whole course, irrespective of your performance in the other parts of the lab.**
- **Submission deadline: 30th November, 2018**
- **Total weight = 20%**
- **Marks distribution: 10, 20, 20, 50**

Problem-1 (Palindrome Maker) (Mark 10)

Problem Statement: Write a program to convert a given string of alphabets to a palindrome. You are allowed to replace an alphabet by its immediately preceding alphabet i.e. 'x' can be replaced by 'w'. Similarly, 'b' can be replaced by 'a'. However, the alphabet 'a' cannot be replaced by any other alphabet. Each replace operation is counted as one operation.

Function's description: It should take as input a string and return the minimum number of operations required to convert the input string to a palindrome. The length of the input string is maximum 5. If the input string is already a palindrome, then the function should return '0'.

Input: a string (lower case and no space)

Example:

Input: pqrqp Output: 0 Explanation: No more operation would be needed since given string is already palindrome.	Input: cba Output: 2 Explanation: **c***ba* → **b***ba* → *aba*

In your report, write answer to the following.

1. How many functions you used?
2. Are all those pure?
3. If not, why? (that means, why the problem can't be solved with pure functions only).
4. How can you use impure functions in Haskell?

Problem-2 (Free coupon Problem) **(Mark 20)**

Problem Statement: Currently IITG Canteen is providing special offer for its customers[1]. With each *aloo paratha* one token is provided. One can accumulate these tokens. The customer can get an *aloo paratha* either by paying the price in full or in exchange of *z* tokens. Write a Haskell program to find the maximum number of *aloo paratha(s)* the customer can get with a given money. Assume the customer is having no tokens with him/her initially.

Note:

1. **All tokens are uniform (indistinguishable).**
2. **A customer purchases it till s/he runs out of either money or tokens.**

Example: Currently Chhotu has Rs. 100/- (x) in his pocket. Each *aloo paratha* costs Rs. 20/- (y) or 2 tokens (z). Initially, Chhotu buys 5 *aloo parathas* which gave him 5 tokens. He then uses 4 tokens to buy 2 more *aloo parathas* which gave him 2 more tokens. Now he is left with 3 tokens. He again uses 2 of them to buy one *aloo paratha* which gave him another token. He is now having 2 tokens. He uses these two tokens to buy another *aloo paratha*. Currently, he is left with only 1 token which is not sufficient to get an *aloo paratha* in exchange. Overall, Chhotu has got a total of 9 (5+2+1+1) *aloo parathas*.

Input: x = 100, y= 20, z=2

Output: Chhotu can get 9 *aloo parathas*

Write a brief report on the following.

1. How many functions you used?
2. Are all those pure?
3. If not, why? (that means, why the problem can't be solved with pure functions only).

Probelem-3 (Minimum Number of Moves) **(Mark 20)**

Problem Statement: A businessman owns a restaurant in the city having **N** workers. The salary of the **i**-th worker is equal to $\mathbf{W_i}$ (**i** = **1**, **2**, ..., **N**). Initially all the workers had different salaries. Once, the businessman decides to make salaries of all the workers to be equal. In order to do this, he can use only one operation: choose one worker and increase the salary of all the other workers by 1, except the chosen worker. In other words, the chosen worker is the loser. The businessman can perform this operation as many times as he wants. But he is a busy man. So, he wants to minimize the total number of operations needed to equalize the salaries of all the workers. Your task is to find the minimum number of times the operation is performed. Write a Haskell program to solve the problem.

[1] This is only an assumption. Please don't approach the canteen for coupons. We are not responsible if they refuse!

Input: The first line of the input contains a single integer denoting the number of workers. The second line contains space-separated integers denoting the salaries of the workers.

Output: A single line containing the minimum number of operations needed to equalize all workers.

Example

Input (case 1):
3
1 2 3
Output (case 1):
3
Input (case 2):
2
42 42
Output (case 2):
0

Case 1. The Businessman can equalize all salaries in 3 turns:

Move Count	IDs of workers whose salaries are increased	Salaries after the move
1	1 2	2 3 3
2	1 2	3 4 3
3	1 3	4 4 4

Case 2. All salaries are already equal. He doesn't need to do anything

Write a brief report on the following.

1. In order to get the minimum number of the operations, which worker should be chosen and why?
2. How many functions you used?
3. Are all those pure?
4. If not, why? (that means, why the problem can’t be solved with pure functions only).

Problem-4 (House Planner) **(50 Marks)**

To create a dream house with the best utilization of space, a good design is necessary. Write a program in Haskell to suggest the best possible design of a house in the available space. Consider the following assumptions:

The house can have bedroom(s), hall(s), kitchen(s), bathroom(s), balcony and garden. Their dimension can be of the following range (the numbers represent some unit):

Bedroom→ 10 x 10 to 15x15

Hall→ 15 X 10 to 20 X15

Kitchen → 7 X 5 to 15 X 13

Bathroom→ 4x 5 to 8 x 9

Balcony→ 5X5 to 10 X10

Garden→ 10 X 10 to 20 x20

Given a space (assume a square area) in square units and the number of the two components (bedroom and hall); find the dimensions of all the components (bedroom, hall, kitchen, bathroom, garden and balcony). Note the following while designing:

1) The dimension of a kitchen must not be larger than that of a hall or a bedroom
2) The dimension of a bathroom must not be larger than that of a kitchen
3) There can be one kitchen for up to three bedrooms
4) Number of bathrooms is one more than the number of bedrooms
5) There must be one garden and one balcony
6) If some space remains such that no further increase in any component is possible, then return it as 'Unused Space'

If no design is possible with the given constraint, return "No design possible for the given constraints" as output

Sample input/output

Input:

Design (1000, 3, 2)

Output:

Bedroom: 3 (11x11)
Hall: 2 (16 X 11)
Kitchen: 1(8 X6)
Bathroom: 4 (4 X 5)
Garden: 1 (11 X 11)
Balcony: 1(6X6)
Unused Space: 0

Answer the following in your report.

1. Write the algorithm (in pseudo-code) that you devised to solve the problem (**you must not write the code for the algorithm**).
2. How many functions you used?
3. Are all those pure?
4. If not, why? (that means, why the problem can't be solved with pure functions only).

In addition, write short notes on the following in the report.

a) Do you think the *lazy* evaluation feature of Haskell can be exploited for better performance in the solutions to the assignments? If so, which solution(s) and how?

b) We can solve the problems using any imperative language as well. Do you find any advantage of using Haskell for these problems (w.r.t the property of lack of side effect)? If your answer is no, elaborate on why not?