CS-431 Programming Language Lab
ASSIGNMENT-3

CHIRAG GUPTA | 170101019

**NOTE- Each Problem Folder contains the corresponding README file.
Check that for compilation and execution.**

PROBLEM 3 - House Planner

a) Write the algorithm (in pseudo-code) that you devised to solve the problem (you must not write the code for the report's algorithm).

- We take the totalArea, the number of bedrooms, and the number of Halls as the input.
- Compute all the possible dimensions for each room component with the given range (a_1, b_1) to (a_2, b_2).
- Compute the number of Kitchens and Bathrooms based on the input given.
- Then we get those combinations of Bedrooms, halls, kitchen, bathrooms, balcony, and garden by concatenating each dimension tuple of all components.
- Also, we filter those combinations with dimensions of kitchen less than that of hall and bedrooms both.
- Also, we choose those combinations where the bathroom has dimensions less than that of the kitchen.
- We remove unwanted tuples that have the same area, as this won't affect the final answer.
- Then of all the feasible combinations, we find the maximum area of all the tuples.
- We find that tuple (bedroom, halls, kitchen, bathrooms, balcony, kitchen) with the area with a max feasible area.
- Print the output in the required format.

b) How many functions did you use?

```
1) formTuple1
2) formTuple2
3) formTuple3
4) formTuple4
5) formTuple5
6) listAreasHelper
7) listAllPossibleAreas
8) removeUnwantedTuples3
9) removeUnwantedTuples4
10) removeUnwantedTuples5
11) calculateMaximumArea
```

```
12) getAllPossibleCouples
13) getAllPossibleTriplets
14) getAllPossibleQuadruple
15) getAllPossibleQintuple
16) getAllPossibleSextuple
17) printOutput
18) design
```

c) **Are all those pure?**

No, ***printOutput*** and ***design*** functions are impure as these functions are using I/O methods. However, none of the other functions updates a global variable or uses an updatable global variable, i.e these functions will return the same result if given the same input. Hence they are classified as pure functions.

d) **If not, why? (Means, why the problem can't be solved with pure functions only).**

As the input and output are necessary for our problem, that's why this problem of impure function is occurring in our case. This can't be solved with pure functions as this has to be necessarily done in our case.

Also, 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?**

Lazy evaluation is a method to evaluate a Haskell program by which the expressions are not considered when they are bound to variables. However, their evaluation is deferred until other computations need their results. Resulting in this, arguments are only evaluated when their values are used, but not when they are passed to a function.

Haskell's lazy evaluation feature can be used in handling lists of significant size as they are evaluated whenever needed. In the ***third Problem,*** the computation of all the rooms' dimensions is not executed when calling the function, but when required.

Hence, lazy evaluation saves computation time as well as memory.

b) **We can solve the problems using an 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?**

Calling a function once is the same as calling it twice and discarding the first call. If you discard any function call result, Haskell will spare itself the trouble and never call the function. Haskell has a considerable advantage because of the ***lack of side effects***. Lack of side effects means that variables are not changed once declared, and hence we can use parallel processing to compute such variables as they will not be altered at any point of execution. Thus, there ***won't be many critical sections,*** and hence we c***an have parallel processing*** to save computational time.