CS431: Programming Languages Lab

Assignment Set III

-Aman Raj (170101006)

❖ Question 1

- **Algorithm**

  Just loop through elements of both the sets taken input as lists and then do the required operation, ensuring no element occurs twice in the output.

- **How many functions did you use?**

  7 Functions

- **Are all those pure?**

  No, almost all the functions are impure which take input from the keyboard except presentInSet and gereratePairs which are used as intermediate functions.

- **If not, why?**

  In any function in which we need to take input from user can't be made pure, it can be done through impure functions only.

❖ Question 2

- **Algorithm**

  We get a random permutation of the teams' list, store the index of its permutation out of all permutation.
  Then simply pair first and second, third and fourth, and so on.
  And assign match timings sequentially
  So first and second teams get the match of index 0.

- **How many functions did you use?**

  7 Functions

- **Are all those pure?**

  No, all the functions are impure which take input from the keyboard like fixture and nextMatch while others are pure functions.

- **If not, why?**

  In any function in which we need to take input from user can't be made pure, it can be done through impure functions only.

❖ Question 3

- **Algorithm**

  First, we find all possible designs of all different types of rooms with given constraints individually. Then we combine them in a group of 2, so a total of 3 groups, then we combine the first two groups, making it a group of 4 and then combined it with the last group.

  In each step, we maintain all the constraints given in the problem statement. At last, we have all possible designs with the design details and their area covered. For the best design, we find the one with the maximum area, it automatically reduces the unused space as we already have placed the constraint of the total possible area in part 1 of the algorithm.
- **How many functions did you use?**

  8 Functions
- **Are all those pure?**

  No, the only impure function is design which takes input from the keyboard while all others are pure functions.
- **If not, why?**

  In any function in which we need to take input from user can't be made pure, it can be done through impure functions only.

❖ Short Notes

I. **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?**

Since the lazy evaluation feature of Haskell helps in better handling of list of very big sizes, it could be exploited to enhance the performance of various

solutions.
In our assignment, in question 3, while combining various designs possible, we are handling lists of very big size and lazy computations helps to efficiently evaluate them.

II. 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?

Many functional programming languages including Haskell has features like *Lack of Side Effect*. In this feature, the value of any variable once assigned cannot be changed, hence issues like global variables are taken care of.
It helps us parallelize our code too as much as possible as there would be no changes to any values once assigned. Using Lazy Computation, this is achieved which helps to reduce the total computation time, hence improving our solution.