Haskell Report

Tamilselvan S 150101079

Bishwendra Chaudry 150101017

Q1:

The program snippet was created by recursively calculating the cumulative ASCII distance between first and last letters in the string.

1. 3 functions were used to create the program.  
   firstLast : takes string as input and returns the substring by removing the first and last characters  
   ascii : takes character as input an returns the integer from asci table  
   palimdrome : takes string as input and calculates the operations required as abs(asci(first)-ascii(last))+palimdrome(firstLast (input))
2. All functions are pure as no variable is reassigned
3. This problem has been solved with pure functions hence there is no side effects.
4. Impure functions can be used by reassigning variables hence creating side effects.

Q2:

The program snippet was created by recursively purchasing parathas unless the customer runs out of money or tokens

1. 2 functions have been used to create the program.  
   parathas: takes initial amount, cost and token  
   parathas1: if both token and money are insufficient return 0  
    if money is sufficient call parathas1  
    if token is sufficient and not money call parathas1
2. Yes, all the functions are pure as not variable is reassigned during function call

Q3:

The program takes the list of salaries of employees as input and return the minimum number of steps required.

1. The worker with highest salary is selected so that no other worker’s modified exceeds the salary of selected worker.
2. Sum1 : takes list as input and returns the sum of elements.  
   Operations : calculates the no. of operations required to normalise the salaries.  
   moves : takes the input from the user and calls Operations
3. The function moves is impure
4. since the function moves takes input and output and it changes the state of variables it impure.

Q4:

The program was created by parsing through all possible configurations of house and finds the configuration with maximum area that is less than or equal to the assigned area.

1. Design (area,bed,hall) : head(combination(area,bed,hall))  
   Combination(area,bed,hall) : sorted(possiblehbktgv)  
   possiblehbktgv = Cartesian product of all possible dimensions of hall, kitchen etc such that all constrints are matched and area of resulting configuration is less than input area.
2. Design : takes input and calls combination  
   combination : finds all possible configuration under the given constraints  
   consumehb : returns all possible configurations involving only hall, bed  
   consumehbk : returns all possible configurations involving only hall, bed, kitchen  
   consumehbkt : returns all possible configurations involving only hall, bed ,kitchen ,bathroom  
   consumehbktg : returns all possible configurations involving only hall, bed ,kitchen ,bathroom ,garden  
   consumehbktgv : returns all possible configurations involving only hall, bed ,kitchen ,bathroom ,garden ,balcony.
3. No, Design function is impure.
4. Design function takes input hence changes the state of IO variable hence impure.
5. Lazy evaluations can help in reducing the amount of unwanted computations required to carry out an operation. In 4th question the program didn’t check for other possible sub-cases of configs. after finding out that the parent block has failed.
6. Since Haskell is a declarative language and with lazy evaluations many heavy computation blocks are avoided from being computed Haskell improves efficiency of runtime compared to traditional imperative languages. And since the state of many variables is not changed, many confusions are avoided in Haskell and multiple RAM access is avoided, increasing the efficiency of the program again.