DATA STRUCTURES

BATCH – A
[TUESDAY MARCH 7, 2017: 2:00 PM – 5:00 PM]

ASSIGNMENTS – 7 CODE: assign08

INSTRUCTIONS: [Total Marks: 30]

i) Read all assignments and each problem has to be answered in the same c file.
ii) Create a .c file following the file name convention: abc-assign08.c
Where abc is your roll number and assign08 is the assignment code
iii) Strictly follow the file name convention and do not use scanf()

PROBLEMS: (on BFS / DFS OR Binary Search Trees)

1) **[Marks: 5 marks]**

Define a node - BSTREE - of a Binary Search Tree (BST) with following fields:

```
nID: <int> - Node Identifier - [1, 2000]
proname: <char> - it can hold a maximum of 16 characters
size: <int> - size [480, 3375] – measured in square feet
nbeds: <float> - [1, 5] - increment can be in steps of 0.5
base: <float> - [2499.0, 4850.0] – base price per square feet
charges: <float> - represents percentage of the base ranging from
                   [2.14, 8.23]
ptype: <int> - ranges in {0, 1, 2}: 0 – basement parking;
               1- covered parking; and 2 – open parking
```

The values of these fields could be generated using a random number generator in the specified range. Assume a list of specific names for the field "proname"

2) **[Marks: 25 marks]**
Using above data structure and function prototypes given below, write your code for following tasks:

a) **[Marks: 5 marks]**
Assume that we are going to generate the Binary Search Tree with the details of n (=20) flats. Create a binary tree with n nodes.

```
BSTREE *genFlatsDataset (BSTREE *bstree, int n);
```

This function should internally insert an element into the binary Search tree in such a way that the nodes insertion is based on size of the flats.

b) **[Marks: 2 marks]**
Write a function to print the details of each item on per line:

```
void printFlatDetails(BSTREE *bstree);
```

c) **[Marks: 3 marks]**
Write a function to search and print the details of the flats by specific number of bedrooms - `n`

```
void FindFlatsByNBeds(BSTREE *bstree, int n);
```

d) **[Marks: 3 marks]**

Write a function to change the parking option of a flat: from `oldPark` and to `oldPark` option.

```
void ModifyParking(BSTREE *bstree,  int oldPark, int newpark);
```

e) **[Marks: 8 marks]**

Write a function to convert the existing Binary Search Tree into a COMPLETE binary tree based on the median of the field: `size`

```
BSTREE *BalanceBST(BSTREE *bstree);
```

f) **[Marks: 4 marks]**

Write a function to delete all flats that carries `charges` within the given range [`min`, `max`]

```
BTNODE *deleteElements(BSTREE *bstree, int min, int max);
```

At the end of the function call print the details of the remaining nodes in the binary search tree.