Programming Assignments for DSA (MCA 2nd Sem)

PA1 : Give an array consisting of only two elements 'a' and 'b'. Arrange the array so that all the a's come before all the b's in O(n) time and O(1) additional space

PA2 :

a. Implement a circular linked list as an ADT with the operations discussed in the lectures.

b. Use the ADT in part (a) to Implement Stack as an ADT with the operations discussed in the lectures.

b. Use the ADT in part (b) to convert an in-fix expression into a post-fix expression.

c. Use the ADT in part (b) and the postfix expression obtained in part (c) to evaluate the expression.

The input should be most general with operations +, -, *, / and ^

PA3 :

a. Implement (growable) Stack as an ADT using arrays with the operations discussed in the lectures. Use the tight strategy to grow the stack.

b. Use the ADT of part (a) to find if a given string of characters drawn from the english alphabet, is a pallindrome.

PA4 :

Checking for balanced parenthesis is one of the most important tasks of a compiler. For example in the following code:

```
main()
{
for(i = 1, i  < n, i++)
{
{
print (i);
}
```

The compiler will give an error as it finds 3 opening parentheses followed by only one closing parenthesis. Consider a hypothetical compiler that generates a sequence 's' of parenthesis in the first pass of the program and calls a module check_bal_parenthesis(s) that returns true if the parentheses are balanced else returns false. In the above example, s is "{ { { }" and the module returns false.

In the following code

```
main()
{
for(i = 1, i  < n, i++)
{
print (i);
}
}
```

s is "{ { } }" and the module returns true.

Write the module check_bal_parenthesis(s) with all the required declarations.

PA5 : Call Center of a bank receives several calls in a day. The representatives of the CC cannot attend to all the calls immediately. Hence a calling client is put on hold until the next representative is available. To be fair to the clients, the calls are attended on first come first serve basis.
a. Use the Implementation of a circular linked list as an ADT of PA 1 to Implement Queue as an ADT with the operations discussed in the lectures.
b. Use the ADT in part (a) to store the call requests. Write a program to add the calls in your database and delete after they are served.

PA6 : Implement AVL trees with following operations:

1. Insertion
2. Deletion

PA7 : In the University of Delhi, there are 50 departments. Each department has a Master's program with an intake of 65 students and a PhD program with different intake for different departments with a maximum of 50 in any department. The roll number of a student is a string consisting of 9 characters: first four digits being the year of admission, next letter indicates Master's (M) program or PhD (P) program followed by 2 letter department code (for example CS for Computer Science, MA for Maths) and the last two characters are roll numbers in the range 00 to 64.

a. Implement Dictionary as an ADT to store the records of the students (Roll No., Name, Address, date of birth) with roll number being the key.
b. Also, add the following operations: Insert_c(), delete_c() and search_c() that performs the specified operation as well as count the number of look-ups for each operation. Compute the average number of look-ups performed each time the program is run.
c. Implement the above ADT using various hash functions discussed in the lecture. For each hash function used, implement all three ways of resolving collision (chaining, linear probing, double hashing). Compare all the combinations of (hash function, collision resolving) with respect to the average number of lookups performed. You must compare them on the same sequence of operations and hence on the same data set. The size of the data should not too small to be trivial.
d. Implement the above ADT using Universal Hashing and double hashing for collision resolution. Also, implement re-hashing when too many entries have been marked deleted.

PA8 :
1. Write a code to implement BST as an ADT with the operations discussed in the video lecture including insertion, search, traversal (general and then specialized), minimum, maximum, successor and predecessor.

2. Add "delete" operation to the above ADT.

3 .For running the program, Input values to be generated using random number generator.

PA9 : Write a program that given a string s in the set S of strings drawn from the input alphabet {a, b}, defined recursively as given below, returns yes if s is in S else returns false

S1 = {e}, e represents an empty string
S2 = {a s b , where s is in S}
Delete: S3 = {s1 s2 where s1, s2 are in S}

Delete S = S1 U S2 U S3

S = S1 U S2

Implement an appropriate data structure as ADT and use it to perform the above task.