IECS 104 Introduction to Computer Science Lab II

Lab 9

1. Using the following structure definition,

```
struct stackNode
{
      int data;
      struct stackNode *nextPtr;
};
```

   a. Write a C function to push an integer into the stack.
   b. Write a C function to pop an integer from the stack.
   c. Write a C function to check if the stack is empty, return an integer 1, otherwise, return an integer 0.
   d. Write a C function to print all integers from the bottom of the stack.
   e. Write a main function that
      I. Read stack data (from a text file, “stack.txt”), i.e. a set of integers, and create a stack (the first data in the “stack.txt” should be at the bottom of the stack).
      II. Print all integers from the bottom of the stack.
      III. Pop three (3) integers from the stack and push back these three (3) integers using the same order, i.e., the first pop the first push.
      IV. Print all integers from the bottom of the stack.

2. Using the following structure definition,

```
struct node
{
      char string[15];
      node *nptr;
};
```

   a. Write a C function to insert a node data into a linked list.
   b. Write a C function to print all node data in the linked list line by line.
   c. Write a C function that input a string and check if the string is in the linked list, return an integer 1, otherwise, return an integer 0.
   d. Write a C function that input a string and delete the string in the linked list which matches the input string.
   e. Write a main function that
      I. Read node data (from a text file, “linked_list.txt”), i.e. a set of strings, and create a linked list for the strings.

II. Print all node data in the linked list line by line.

III. Randomly select a string from the text file, “linked_list.txt”, and print the selected string as well as delete the string in the linked list.

IV. Print all node data in the linked list line by line.

3. Using the following structure definition,

```
struct node
{
      int item;
      node *lptr;
      node *rptr;
};
```

a. Write a C function to insert a node data, i.e. int item, into a binary tree.

b. Write a C function to print all node data in the binary tree using Inorder traversal, Preorder traversal, or Postorder traversal..

c. Write a C function that input an integer and check if the integer is in the binary tree, return an integer 1, otherwise, return an integer 0.

d. Write a C function that input an integer and delete the integer in the binary tree which matches the input integer.

e. Write a main function that

I. Read node data (from a text file, “binary_tree.txt”), i.e. a set of integers, and create a binary tree for the integers.

II. Print all node data in the binary tree using Inorder traversal , Preorder traversal, and Postorder traversal.

III. Randomly select an integer from the text file, “binary_tree.txt”, and print the selected integer as well as delete the integer from the binary tree.

IV. Print all node data in the binary tree using Inorder traversal, Preorder traversal, and Postorder traversal.