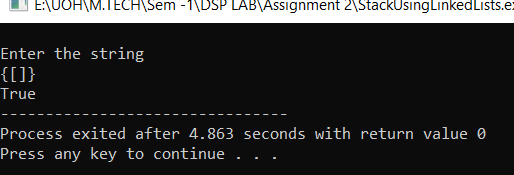
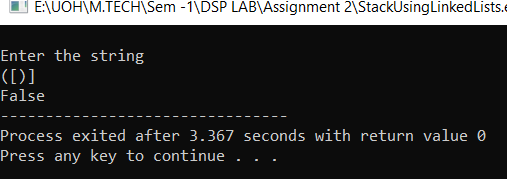
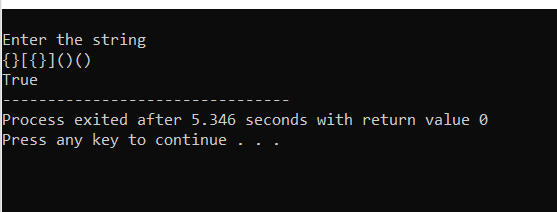
**Question 1:**

**Write a Program that ask user to enter the string (Note : string can contain only**

**‘(’ , ’)’ , ’[’ , ’]’, ‘{’ , ‘}’ these mentioned 6 characters only )**

* The problem is solved by making use of stack implemented by linked list.
* Program is written on C with three basic functions.
* (i) **Push(char):** is used to push the character elements into the linked list.
* (ii) **Pop**: **):** is used to pop the character elements out of the linked list. These elements are then matched with the current element being traversed from the input string, if it not the opening – closing pair of brackets received in right order, flag is updated to 1, which in turn returns **false**, later in the main().
* (iii) **main()**: is used as the driver function.
* The input is checked against any exception, by using if and else to check for expected input.  
  If unexpected input is received, The program returns Invalid Input and exits.

**Execution Screenshots:**  
****

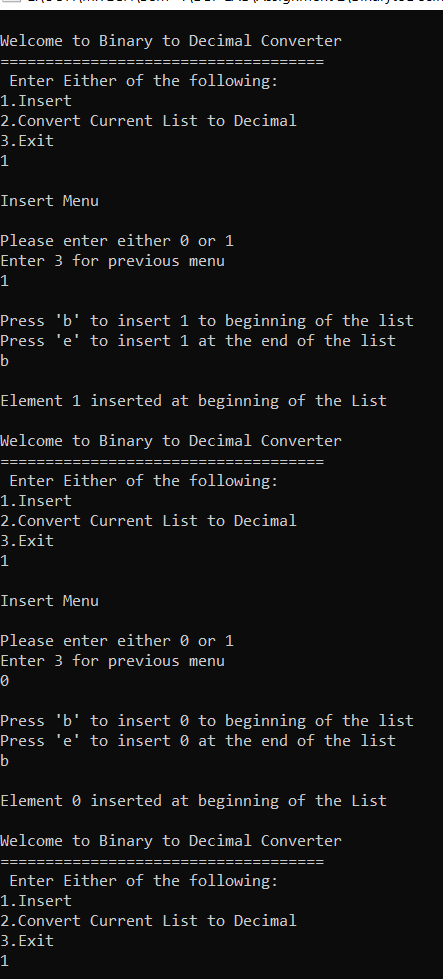
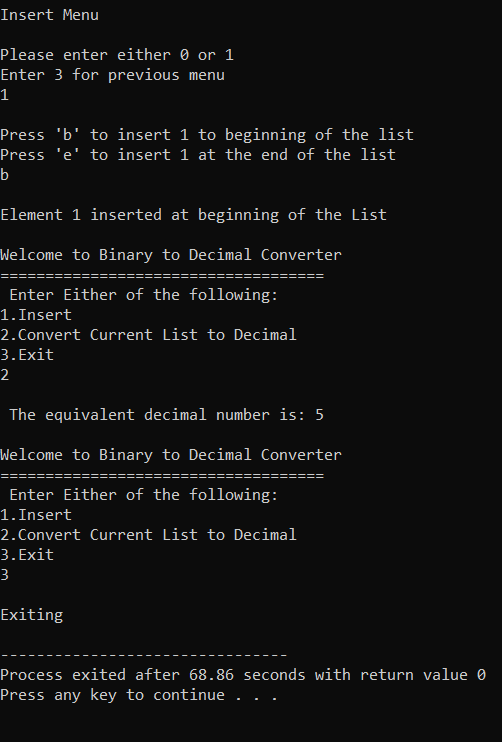
****

**Question 2:**

**Binary to Decimal Conversion of the Input elements in a Linked List.**

* The problem is solved by making use of simple linked list.
* Program is written on C with four basic functions.
* (i) **insertatbeg():** is used to add the node with it’s data at the beginning of the linked list.
* (ii) **insertatend():** is used to add the node with it’s data at the end of the linked list.
* (iii) **bintodec():** traverses through the linked list and gets the count first, and then uses it and traverses the list to calculate the decimal equivalent of the binary sequence digits stored in each node combined.
* (iv) **main()**: is used as the driver function.
* The input is checked against any exception, by using if and else to check for expected input.  
  If unexpected input is received, The program returns Invalid Input and goes back to main menu using **goto** statement.

**Execution Screenshots:**

**Question 3:**

**Write a source code to create a singly circular linked list. A node should consist of  
• Character array to store three strings in “info” part  
• Address of next node as “link” part**

* The problem is solved by making use of singly circular linked list.
* Program is written on C with these basic functions.
* (i) **addnode():** adds a node to the linked list in the last node.  
  First we traverse the list to reach the last node, the ‘link’ of the last node will be the head node.  
  We add the new node to the last node and place it’s link pointer towards head node, so the circular property is maintained.
* **(ii) displayall():** is a function used to display the current character array in each node of the linked list
* **(iii) displaysecondonly():** custom function that helps in the subproblem to get the second string element in the character array of each node.
* **(iv) checkstring():** checks the string rule mentioned in the problem statement. It is also checked after every new node is added to the linked list. Prints ‘string not matching’ when the first string of a node does not match with the third string of the previous node.
* **(v) replacenodestring():** custom method with the functionality to change the first/second/third string of any desired node.
* **(vi) main():** driver function, with menu driven program, supported by simple if and else statements, and use of **goto** statements whenever required.

**Execution Screenshots:**

