Lab4 Part a

1. It will not work because we are trying to insert a new node between two nodes in the list and in the process the newly created node’s pointer with data 9 has to be set to the previous node’s next pointer before the previous node’s next pointer points to the new node (9). The process is however reversed in this case hence the code will not work. The correct code will be:

SNode \*n = new SNode(9);

n->next = node4->next;

node4->next = n;

1. a. Definitions

* **ADT**: Abstract data type is a data type which is not implemented but has only a conceptual format. An example is a List which can be implemented with Arrays.
* **List**: A list is an abstract data type which consist of elements of the same data type in an ordered manner, with a size and can also be duplicated.
* **Push**: Push is a method which adds an element to the end of a list
* **Pop**: Pop involves removing the last element from a list
* **Stack**: A stack is a data structure which operates in Last in First Out context. Elements in a stack are inserted and removed from only one end.
* **Arrays**: Arrays are data structures which constant-size sequential elements of the same data type.
* **Time Analysis**: Time analysis refers to the amount of time it takes to execute an algorithm. It involves counting the number of elementary operations performed by the function. In the context of a list, time analysis is a function of the number of steps to carry out a function.
* **Linked List**: This is another implementation of the List Abstract data type. Linked list is characterized with no fixed size as well as no wasted space.
* **Friend**: A friend of a class B is a function or class that is not a member of B, but is granted the same access to B as the members of B. That is, the friend class can have access to the private and protected properties and methods of class B
* **Kluge**: Kluge refers to a configuration while inelegant, inefficient or clumsy can solve a specific problem.

b. koala

c. This is because the elements of a linked list are not sequential in memory and since pop involve removing the last element, we must traverse the entire list.

d. This is because when inserting we have to create a new node at k and it is the node at k-1 which holders the pointer to the kth node.

e. It would be very inefficient to reverse or traverse in reverse order as there is no previous pointer. This means that doing such tasks would take O(n).

3.

a. Arrays are sequential in memory so finding the kth element involves just one step

b. It is O(n) because we must traverse both the array and the Linked list n number of times where n refers to the size.