DSA Assignment # 2

Submission: Sun, 12/04/2020

Note: All questions must have a menu to test each function e.g.

```
**** Menu *****
Press 1 for….
Press 2 for…
Press 3 for…
Press 0 for Quit
***************
```

Q # 1: Implement a recursive function to find the product of two numbers a and b.

```
int product(int a, int b)
```

Note: Use your singly linked list implementation for the following question. Use only recursion to implement these operations

1. Implement a recursive **member function** "recursivePrint" which prints the singly linked in reverse order. `void recursivePrint() const`
2. Implement a recursive **member function** "length" which recursively finds the length of the linked list. `int length() const`
3. Implement a recursive **member function** "isSorted" which recursively checks whether the linked list is sorted (ascendingly). `bool isSorted() const`
4. Implement a **member function** "deleteAll" which recursively deletes all nodes of linked list. `void deleteAll();`

5. Create a main function with following instructions:
   a. Find product of 15 and -9. Print the result
   b. Insert at head of your singly linked list: 10, 9, 7, 5.
   c. Call recursivePrint function.
   d. Print the output of isSorted.
   e. Print the length of linked list.
   f. Call deleteAll function.
   g. Print the length of linked list.

Q # 2: Implement a template-based **stack** using a **singly linked list**. The required member methods are:

int size(): returns the count of total element stored in the stack.
bool isEmpty(): returns true if the stack is empty else false.
bool top(T&): returns, but does not delete, the topmost element from the stack via the parameter passed by reference. It returns false via a return statement if there is no element in the stack, else it returns true and assigns the top most element to the parameter passed by reference.
void pop(): deletes the top most element from the stack. If there is no element, return some error.
push(T const& e): pushes the element "e" on top of the stack.

Q # 3: Implement a template-based **queue** using a **singly linked list**. The required member methods are:

int size() : returns the count of total element stored in the queue.
bool isEmpty(): returns true if the queue is empty else false.
bool front(T&): returns, but does not delete, the front element from the queue via the parameter passed by reference. It returns false via a return statement if there is no element in the queue, else it returns true and assigns the front element of the queue to the parameter passed by reference.
void dequeue(): deletes the front element from the queue. If there is no element, return some error.
Void enqueue(T const& e): inserts the element "e" at the back of the queue if there is some space available. Otherwise it returns some error.