(2) What is recursion?

Recursion is when a function refers to itself to get an answer.

The next two questions involve the following data types:

template <class T>

struct Node

{

T data;

Node<T>\* next;

};

template <class T>

class List

{

private:

Node<T>\* head;

Node<T>\* tail;

public:

.

.

.

};

Also assume that current is a pointer to a node in the middle of a singly linked list, and that newNode is a pointer to a newly allocated node.

(4) Write a code fragment (not a complete method) that will insert newNode at the front of a non-empty list.

newNode -> next = head;

(4) Write a code fragment that will insert newNode at the rear of a non-empty list.

tail = newNode;

(4) Write a code fragment that will insert newNode into the list after the node pointed to by current.

current -> next = newNode;

(4) Write a recursive method that will return the size of the list. The method should take a pointer to an Node<T> and return an int. When the method is initially called, it will be passed the list's head pointer.

int size(Node<T>\* integer)

{ return size(head); }

(4) Write a non-recursive method that will return the size of the list. The method should take no arguments and return an int.

int size()

{ return size; }

(2) Declare a pointer to a function that returns a bool and takes two double arguments.

bool (\*function)(double, double);

(4) Rewrite the contents of the following array once the array has been partitioned by the quicksort partition code used on Assignment 8.

26 10 67 30 18 54 41 28

26 10 18 67 30 54 41 28