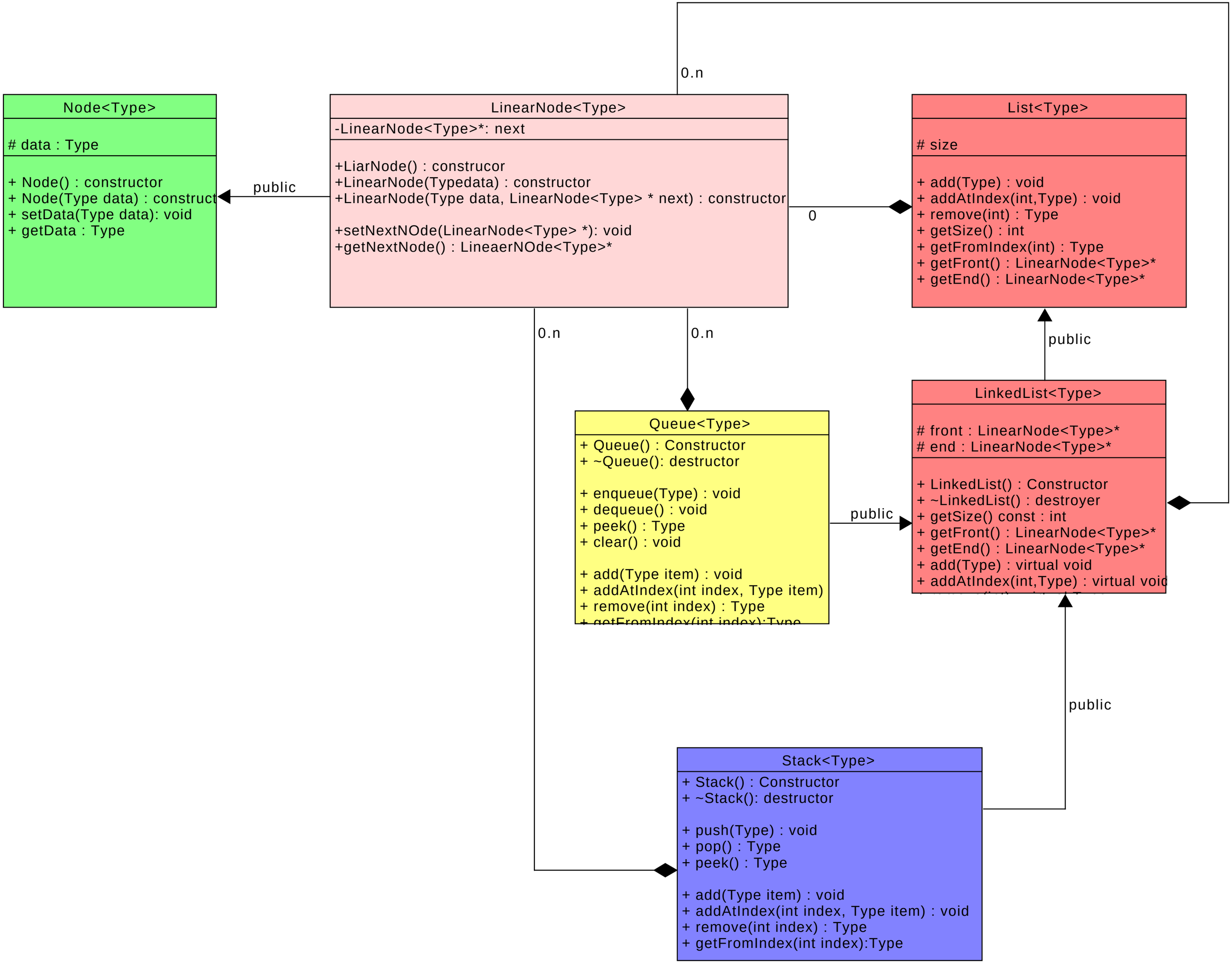
Node<Type>
data : Type
+ Node() : constructor
+ Node(Type data) : construct
+ setData(Type data): void
+ getData : Type
public
LinearNode<Type>
-LinearNode<Type>*: next
+LiarNode() : construcor
+LinearNode(Typedata) : constructor
+LinearNode(Type data, LinearNode<Type> * next) : constructor
+setNextNOde(LinearNode<Type> *): void
+getNextNode() : LineaerNOde<Type>*
0.n
0
List<Type>
size
+ add(Type) : void
+ addAtIndex(int,Type) : void
+ remove(int) : Type
+ getSize() : int
+ getFromIndex(int) : Type
+ getFront() : LinearNode<Type>*
+ getEnd() : LinearNode<Type>*
0.n
0.n
public
Queue<Type>
+ Queue() : Constructor
+ ~Queue(): destructor
+ enqueue(Type) : void
+ dequeue() : void
+ peek() : Type
+ clear() : void
+ add(Type item) : void
+ addAtIndex(int index, Type item)
+ remove(int index) : Type
public
LinkedList<Type>
front : LinearNode<Type>*
end : LinearNode<Type>*
+ LinkedList() : Constructor
+ ~LinkedList() : destroyer
+ getSize() const : int
+ getFront() : LinearNode<Type>*
+ getEnd() : LinearNode<Type>*
+ add(Type) : virtual void
+ addAtIndex(int,Type) : virtual void
public
Stack<Type>
+ Stack() : Constructor
+ ~Stack(): destructor
+ push(Type) : void
+ pop() : Type
+ peek() : Type
+ add(Type item) : void
+ addAtIndex(int index, Type item) : void
+ remove(int index) : Type
+ getFromIndex(int index):Type