```cpp
#include <iostream>
using namespace std;

template <typename T>
class Node {
public:
  T data;
  Node* next;

  Node(T data) : data(data), next(nullptr) {}
};

template <typename T>
class LinkedList {
private:
  Node<T>* head;
  int size;

public:
  LinkedList() : head(nullptr), size(0) {}

  // Insert at the beginning
  void insertAtBeginning(T data) {
    Node<T>* newNode = new Node<T>(data);
    newNode->next = head;
    head = newNode;
    size++;
  }

  // Insert at the end
  void insertAtEnd(T data) {
    Node<T>* newNode = new Node<T>(data);
    if (head == nullptr) {
      head = newNode;
    } else {
      Node<T>* current = head;
      while (current->next != nullptr) {
        current = current->next;
      }
      current->next = newNode;
    }
    size++;
  }

  // Delete at the beginning
  void deleteAtBeginning() {
    if (head == nullptr) {
      return;
    }
    Node<T>* temp = head;
    head = head->next;
    delete temp;
    size--;
  }

  // Print the linked list
  void printList() {
    Node<T>* current = head;
    while (current != nullptr) {
      cout << current->data << " ";
      current = current->next;
    }
    cout << endl;
  }

  // Get the size of the linked list
```

```cpp
  int getSize() {
    return size;
  }

  // Destructor to free memory
  ~LinkedList() {
    Node<T>* current = head;
    while (current != nullptr) {
      Node<T>* temp = current;
      current = current->next;
      delete temp;
    }
    head = nullptr;
  }
};

int main() {
  LinkedList<int> intList;
  intList.insertAtEnd(10);
  intList.insertAtBeginning(5);
  intList.insertAtEnd(15);

  cout << "Linked List: ";
  intList.printList();  // Output: Linked List: 5 10 15

  cout << "Size of List: " << intList.getSize() << endl;  // Output: Size of List: 3

  intList.deleteAtBeginning();

  cout << "Deleting the element at the first of the list" << endl;

  cout << "Linked List after deletion: ";
  intList.printList();  // Output: Linked List: 10 15

  return 0;
}
```