```cpp
#include "node.h"

template <class T>
class LinkedList
{
    public:
        LinkedList();
        ~LinkedList();
        bool isEmpty();
        int getSize();
        void addFirst(T data);
        void addLast(T data);
        bool add(T data, int pos);
        void deleteFirst();
        void deleteLast();
        void del(int pos);
        int deleteAll();
        T set(T data, int pos);
        T get(int pos);
        bool change(int pos1, int pos2);
        void print();

    private:
        node<T> *head;
        int size;
        void borraTodo();
};


template <class T>
LinkedList<T>::LinkedList(){
    head = nullptr;
    size = 0;
}

template <class T>
LinkedList<T>::~LinkedList(){
    borraTodo();
}

template <class T>
void LinkedList<T>::borraTodo(){
    node<T> *curr = head;
    while (head != nullptr){
        head = head->getNext();
        delete curr;
        curr = head;
    }
}

template <class T>
bool LinkedList<T>::isEmpty(){
    return (head == nullptr);
}

template <class T>
int LinkedList<T>::getSize(){
    return size;
}

template <class T>
void LinkedList<T>::addFirst(T data){
    head = new node<T>(data, head);
    size++;
}

```

```
67  template <class T>
68  void LinkedList<T>::addLast(T data){
69      if (isEmpty()){
70          addFirst(data);
71      }
72      else{
73          node<T> *curr = head;
74          while (curr->getNext() != nullptr){
75              curr = curr->getNext();
76          }
77          curr->setNext(new node<T>(data));
78          size++;
79      }
80  }
81
82  template <class T>
83  bool LinkedList<T>::add(T data, int pos){
84      if (pos < 0 || pos > size){
85          return false;
86      }
87      if (pos == 0){
88          addFirst(data);
89      }
90      else if (pos == size){
91          addLast(data);
92      }
93      else{
94          node<T> *curr = head;
95          for (int i=1; i<pos; i++){
96              curr = curr->getNext();
97          }
98          curr->setNext(new node<T>(data, curr->getNext()));
99          size++;
100     }
101     return true;
102 }
103
104 template <class T>
105 void LinkedList<T>::deleteFirst(){
106     if (!isEmpty()){
107         node<T> *curr = head;
108         head = head->getNext();
109         delete curr;
110         size--;
111     }
112 }
113
114 template <class T>
115 void LinkedList<T>::deleteLast(){
116     if (size <= 1){
117         deleteFirst();
118     }
119     else{
120         node<T> *curr = head;
121         while (curr->getNext()->getNext() != nullptr){
122             curr = curr->getNext();
123         }
124         delete curr->getNext();
125         curr->setNext(nullptr);
126         size--;
127     }
128 }
129
130 template <class T>
131 int LinkedList<T>::deleteAll(){
132     borraTodo();
```

```
133     int auxSize = size;
134     size = 0;
135     return auxSize;
136 }
137 
138 template <class T>
139 void LinkedList<T>::del(int pos){
140     if (pos == 0){
141         deleteFirst();
142     }
143     else{
144         node<T> *curr = head;
145         for (int i=1; i<pos; i++){
146             curr = curr->getNext();
147         }
148         node<T> *aux = curr->getNext();
149         curr->setNext(aux->getNext());
150         delete aux;
151         size--;
152     }
153 }
154 
155 template <class T>
156 T LinkedList<T>::get(int pos){
157     node<T> *curr = head;
158     for (int i=1; i<=pos; i++){
159         curr = curr->getNext();
160     }
161     return curr->getData();
162 }
163 
164 template <class T>
165 T LinkedList<T>::set(T data, int pos){
166     node<T> *curr = head;
167     for (int i=1; i<=pos; i++){
168         curr = curr->getNext();
169     }
170     T dataAux = curr->getData();
171     curr->setData(data);
172     return dataAux;
173 }
174 
175 template <class T>
176 bool LinkedList<T>::change(int pos1, int pos2){
177     if (pos1 < 0 || pos2 < 0 || pos1 >= size || pos2 >=size){
178         return false;
179     }
180     if (pos1 == pos2){
181         return true;
182     }
183     int posMen = (pos1 < pos2 ? pos1 : pos2);
184     int posMay = (pos1 > pos2 ? pos1 : pos2);
185     node<T> *curr1 = head;
186     for (int i=1; i<=posMen; i++){
187         curr1 = curr1->getNext();
188     }
189     node<T> *curr2 = curr1;
190     for (int i=1; i<=(posMay-posMen); i++){
191         curr2 = curr2->getNext();
192     }
193     T dataAux = curr1->getData();
194     curr1->setData(curr2->getData());
195     curr2->setData(dataAux);
196     return true;
197 }
198 
```

```cpp
template <class T>
void LinkedList<T>::print(){
    node<T> *curr = head;
    while (curr != nullptr){
        cout << curr->getData()<< endl;
        curr = curr->getNext();
    }
}
```