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

class element {
    public:
        double value;
        element *next;
        element *prev;

        element(double v = 0)
        {
                this->value = v;
                        next = NULL;
                        prev = NULL;
                }
                void print()
                {
                        cout<<"value: " << value;
                }
};

class list{
        private:
                element *head; //указатель на начало списка
        element *tail; //указатель на конец списка
        public:
        list(){
                        head = NULL;
                tail = NULL;
                }

        void push_back(int value){
                element *newElem = new element(value);
                if (head == NULL)
                        {
                        head = newElem;
                        tail = newElem;
                        return;
                }
                newElem->prev = tail;
                        tail->next = newElem;
                tail = newElem;
                }

                void push_front(int value){
                        element *newElem = new element(value);
                if (head == NULL)
                        {
                        head = newElem;
                        tail = newElem;
                        return;
                }
                newElem->next = head;
                head->prev = newElem;
                head = newElem;
                }

                void pop_back(){
                        if(size() == 0)
                        {
                                cout << "List too short" << endl;
                        return;
                        }
                        element *temp = head;
                        while(temp->next != NULL)
                                temp = temp->next;

                        temp->prev->next = NULL;
                }

                void pop_front(){
                        if(size() == 0){
                                cout << "List too short" << endl;
                        return;
                        }
                        head = head->next;
                        head->prev = NULL;
                }

                void erase(int index){
                        if (index > (size())){
                        cout << "List too short" << endl;
                        return;
                        }
                if (index == 1){
                        head = head->next;
                        return;
                }
                if (index == size()){
                        tail = tail->prev;
                        return;
                }
                element *temp = head;
                int count = 0;
                while (temp->next != NULL){
                        count++;
                        if (count == index - 1)
                        break;
                        temp = temp->next;
                }
                temp->next = temp->next->next;
                }

                int size(){
                        element *temp = head;
                int count = 0;
                    while (temp != NULL){
                        temp = temp->next;
                        count++;
                }
                return count;
                }

                int sizelist(){
                        element *temp = head;
                int count = 0;
                    while (temp != NULL){
                        temp = temp->next;
                        count++;
                }
                cout<<"Size of list: "<<count << endl<< endl;
                }

                void insert(int index, int value){
                        element *newElem = new element(value);
                if (index > (size()+1)){
                        cout << "List too short" << endl;
                        return;
                }
                if (head == NULL || index == 1){
                        newElem->next = head;
                        head->prev = newElem;
                        head = newElem;

                        if (tail == NULL)
                        tail = newElem;

                        return;
                }
                if (index == size() + 1){
                        tail->next = newElem;
                        newElem->prev = tail;
                        tail = newElem;
                        return;
                }
                element *temp = head;
                int count = 0;
                while (temp->next != NULL)
                {
                        count++;
                        if (count == index - 1)
                        break;
                        temp = temp->next;
                }
                element *temp2 = temp->next;
                temp->next = newElem;
                newElem->prev = temp;
```

```
                newElem->next = temp2;
                temp2->prev = newElem;
                }

        void swap(int indexA, int indexB){
                element *Elem1, *Elem2, *prev1, *prev2, *temp;
                int i;
                int maxPos = (indexA > indexB) ? indexA : indexB;
                if ((indexA <= 0 || indexA > size()) || (indexB <= 0 || indexB > size())){
                        cout << "Index error" << endl;
                        return;
                }
                    if (indexA == indexB){
                        cout << "Index error" << endl;
                        return;
                }
                    i = 1;
                temp  = head;
                prev1 = NULL;
                prev2 = NULL;
                    while (temp != NULL && (i <= size())){
                        if (i == indexA - 1)
                        prev1 = temp;
                        if (i == indexA)
                        Elem1 = temp;

                        if (i == indexB - 1)
                        prev2 = temp;
                        if (i == indexB)
                        Elem2 = temp;

                        temp = temp->next;
                        i++;
                }
                    if (Elem1 != NULL && Elem2 != NULL){
                        if (prev1 != NULL)
                        prev1->next = Elem2;

                        if (prev2 != NULL)
                        prev2->next = Elem1;

                        temp = Elem1->next;
                        Elem1->next = Elem2->next;
                        Elem2->next = temp;

                        if (prev1 == NULL)
                        head = Elem2;
                        else if (prev2 == NULL)
                        head = Elem1;
                }

        }

                void print(){
                        element *temp = head;
                while (temp != NULL){
                        temp->print();
                        cout << endl;
                        temp = temp->next;
                }
                cout << endl;
                }

                void inverted_print(){
                        element *temp = tail;
                while (temp != NULL){
                        temp->print();
                        cout << endl;
                        temp = temp->prev;
                }
                cout << endl;
                }
};


int main()
{
        list *l = new list();
        cout<<"create and filling"<< endl;
        l->push_back(2); //Добавляет элемент в конец списка
        l->push_front(4);//Добавляет элемент в начало списка
        l->print();//Вывод элементов списка
        l->sizelist(); // выводим размер списка

        cout<<"erase one element"<< endl;
        l->erase(1);//удаляет элементы с индексом 1
        l->print();
        l->sizelist(); // выводим размер списка

        cout<<"add 1 element in fron and 2 in the back"<< endl;
        l->push_front(6); // добавляем 6 в начало списка
        l->push_back(8); // добавляем 8 в конец списка
        l->push_back(2); // добавляем 2 в конец списка
        l->print(); // выводим список для наглядности
        l->sizelist(); // выводим размер списка

        cout<<"erasing 3 elements"<< endl;
        l->pop_front(); // удаляем элемент в начале списка
        l->print(); // выводим список для наглядности результата исполнения
        l->sizelist(); // выводим размер списка
        l->pop_back(); // удаляем элемент в конце списка
        l->print();// выводим список для наглядности результата исполнения
        l->sizelist(); // выводим размер списка

        cout<<"adding 3 elements in front"<< endl;
        l->push_front(1);// добавляем 1 в начало списка
        l->push_front(3);// добавляем 3 в начало списка
        l->push_front(5);// добавляем 5 в начало списка
        l->print();// выводим список для наглядности результата исполнения
        l->sizelist(); // выводим размер списка

        cout<<"adding 2 elements in back and invert"<< endl;
        l->push_back(7);// добавляем 7 в конец списка
        l->push_back(9);// добавляем 9 в конец списка
        l->inverted_print();//инвертированный вывод
        l->sizelist(); // выводим размер списка

        cout<<"swaping 1 and 2 elements and inserting 3 element"<< endl;
        l->swap(1,2); // меняем местами два элемента
        l->print();// выводим список для наглядности результата исполнения
        l->insert(3, 33);//заменяем значение третьего элемента на 33
        l->print();// выводим список для наглядности результата исполнения



        return 0;
}
```