Đã bắt đầu vào lúc	Thứ năm, 28 Tháng chín 2023, 10:13 AM
Tình trạng	Đã hoàn thành
Hoàn thành vào lúc	Thứ hai, 2 Tháng mười 2023, 3:27 PM
Thời gian thực hiện	4 ngày 5 giờ
Điểm	3,00/3,00
Điểm	**10,00** của 10,00 (**100**%)

Câu hỏi **1**

Chính xác

Điểm 1,00 của 1,00

Implement methods **ensureCapacity, add, size** in template class **ArrayList** representing the array list with type T with the initialized frame. The description of each method is given in the code.

```
template <class T>
class ArrayList {
protected:
    T* data;        // dynamic array to store the list's items
    int capacity;   // size of the dynamic array
    int count;      // number of items stored in the array
public:
    ArrayList(){capacity = 5; count = 0; data = new T[5];}
```

```
    ~ArrayList(){ delete[] data; }
    void    add(T e);
    void    add(int index, T e);
    int     size();
    void    ensureCapacity(int index);
};
```

For example:

Test	Result

ArrayList<int> arr;
int size = 10;
 	
for(int index = 0; index < size; index++){
    arr.add(index);
}
 	
cout << arr.toString() << '\n';
cout << arr.size();
	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
	10

Test	Result

ArrayList<int> arr;
int size = 20;
 	
for(int index = 0; index < size; index++){
    arr.add(0, index);
}
 	
cout << arr.toString() << '\n';
cout << arr.size() << '\n';
arr.ensureCapacity(5);
	[19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0]
	20

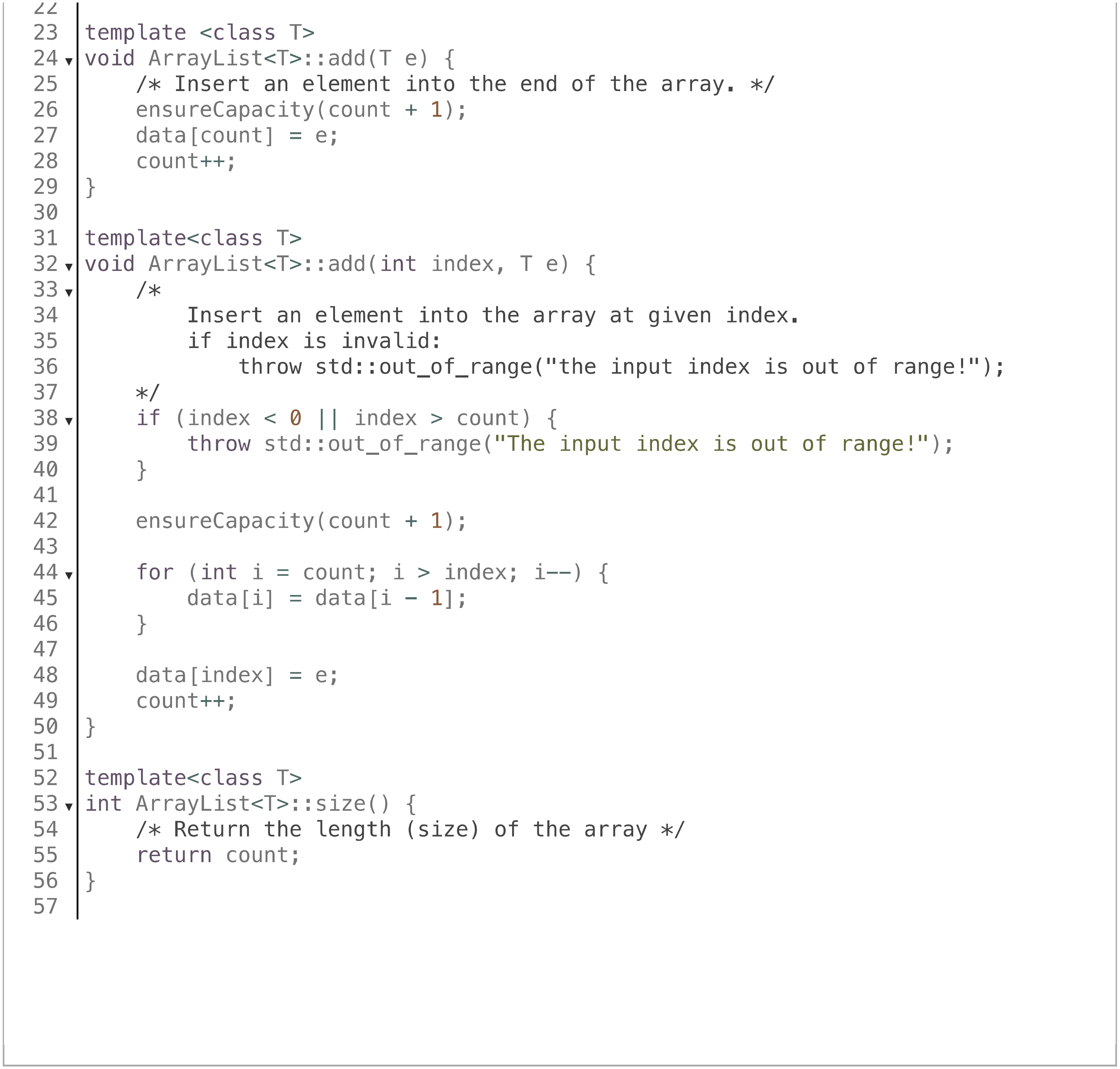
Answer: (penalty regime: 0, 0, 0, 0, 0, 100 %)

Reset answer

```
template<class T>
void ArrayList<T>::ensureCapacity(int cap){
    /*
        if cap == capacity:
            new_capacity = capacity * 1.5;
            create new array with new_capacity
        else: do nothing
    */
    if (cap > capacity) {
        int newCapacity = capacity * 1.5;
        T* newData = new T[newCapacity];

        for (int i = 0; i < count; i++) {
            newData[i] = data[i];
        }

        delete[] data;
        data = newData;
        capacity = newCapacity;
    }
}
```



```
template <class T>
void ArrayList<T>::add(T e) {
    /* Insert an element into the end of the array. */
    ensureCapacity(count + 1);
    data[count] = e;
    count++;
}

template<class T>
void ArrayList<T>::add(int index, T e) {
    /*
        Insert an element into the array at given index.
        if index is invalid:
            throw std::out_of_range("the input index is out of range!");
    */
    if (index < 0 || index > count) {
        throw std::out_of_range("The input index is out of range!");
    }

    ensureCapacity(count + 1);

    for (int i = count; i > index; i--) {
        data[i] = data[i - 1];
    }

    data[index] = e;
    count++;
}

template<class T>
int ArrayList<T>::size() {
    /* Return the length (size) of the array */
    return count;
}

```

	Test	Expected	Got	

✓	ArrayList<int> arr;
	int size = 10;
	
	for(int index = 0; index < size; index++){
	arr.add(index);
	}
	
	cout << arr.toString() << '\n';
	cout << arr.size();
		[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
		10
			[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
			10
				✓

✓	ArrayList<int> arr;
	int size = 20;
	
	for(int index = 0; index < size; index++){
	arr.add(0, index);
	}
	
	cout << arr.toString() << '\n';
	cout << arr.size() << '\n';
	arr.ensureCapacity(5);
		[19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0]
		20
			[19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0]
			20
				✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi **2**

Chính xác

Điểm 1,00 của 1,00

Implement methods **Get, set, clear, empty, indexOf, contains** in template class **ArrayList** representing the array list with type T with the initialized frame. The description of each method is given in the code.

```
template <class T>
class ArrayList {
protected:
    T* data;        // dynamic array to store the list's items
    int capacity;   // size of the dynamic array
    int count;      // number of items stored in the array
public:

```

```
 ArrayList(){capacity = 5; count = 0; data = new T[5];}
   ~ArrayList(){ delete[] data; }
```

```
    void    add(T e);
    void    add(int index, T e);
    int     size();
    bool    empty();  // check if the list is empty or not
    void    clear();  //remove data and set the list to the initial condition
    T       get(int index);  //get the element at the index, if the index is out of range, "throw
std::out_of_range("index is out of range");"
```

```
    void    set(int index, T e);  //set the index position in the list with the value e
    int     indexOf(T item);  //get the first index of item in the list, else return -1
    bool    contains(T item);   //check if the item is in the list
    T       removeAt(int index);
    bool    removeItem(T item);
```

```
};
```

Notice: You just have to implement the methods: set, get, clear, empty, indexOf, contains. Other methods have been implemented already.

For example:

Test	Result

ArrayList<int> arr;
int size = 10;
for(int index = 0; index < size; index++){
arr.add(index);
}
cout << arr.toString() << '\n';
arr.set(0,100);
cout << arr.get(0) << '\n';
cout << arr.toString() << '\n';
arr.clear();
cout << arr.toString() << '\n';
cout << arr.empty() << '\n';
for(int index = 0; index < size; index++){
arr.add(index);
}
cout << arr.indexOf(7) << '\n';
cout << arr.contains(15) << '\n';
	[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
	100
	[100, 1, 2, 3, 4, 5, 6, 7, 8, 9]
	[]
	1
	7
	0

ArrayList<int> arr;
int size = 10;
for(int index = 0; index < size; index++){
arr.add(index);
}
try{
arr.set(10,100);
}
catch(std::out_of_range e){
e.what();
}	

Answer: (penalty regime: 0, 0, 0, 0, 0, 100 %)

Reset answer

```
template <class T>
void ArrayList<T>::set(int index, T e){
    if (index < 0 || index >= count) {
        throw std::out_of_range("Index is out of range");
    }

    data[index] = e;
}

template <class T>
T ArrayList<T>::get(int index){
    if (index < 0 || index >= count) {
        throw std::out_of_range("Index is out of range");
    }

    return data[index];
}

template <class T>
void ArrayList<T>::clear(){
    delete[] data;
    capacity = 5;
    count = 0;
    data = new T[5];
}

template <class T>
bool ArrayList<T>::empty(){
    return (count == 0);
}

template <class T>
int ArrayList<T>::indexOf(T item){
    for (int i = 0; i < count; i++) {
        if (data[i] == item) {
            return i;
        }
    }
    return -1;
}

template <class T>
bool ArrayList<T>::contains(T item){
    return (indexOf(item) != -1);
```

```
}

```

	Test	Expected	Got	

✓	ArrayList<int> arr;
	int size = 10;
	for(int index = 0; index < size; index++){
	arr.add(index);
	}
	cout << arr.toString() << '\n';
	arr.set(0,100);
	cout << arr.get(0) << '\n';
	cout << arr.toString() << '\n';
	arr.clear();
	cout << arr.toString() << '\n';
	cout << arr.empty() << '\n';
	for(int index = 0; index < size; index++){
	arr.add(index);
	}
	cout << arr.indexOf(7) << '\n';
	cout << arr.contains(15) << '\n';
		[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
		100
		[100, 1, 2, 3, 4, 5, 6, 7, 8, 9]
		[]
		1
		7
		0
			[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
			100
			[100, 1, 2, 3, 4, 5, 6, 7, 8, 9]
			[]
			1
			7
			0
				✓

✓	ArrayList<int> arr;
	int size = 10;
	for(int index = 0; index < size; index++){
	arr.add(index);
	}
	try{
	arr.set(10,100);
	}
	catch(std::out_of_range e){
	e.what();
	}
				✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi **3**

Chính xác

Điểm 1,00 của 1,00

Implement methods **removeAt, removeItem, clear** in template class **ArrayList** representing the singly linked list with type T with the initialized frame. The description of each method is given in the code.

```
template <class T>
class ArrayList {
protected:
    T* data;        // dynamic array to store the list's items
    int capacity;   // size of the dynamic array
    int count;      // number of items stored in the array
public:
    ArrayList(){capacity = 5; count = 0; data = new T[5];}
   ~ArrayList(){ delete[] data; }
```

```
    void    add(T e);
    void    add(int index, T e);
    int     size();
    bool    empty();
    void    clear();
    T       get(int index);
    void    set(int index, T e);
    int     indexOf(T item);
    bool    contains(T item);
    T       removeAt(int index);
    bool    removeItem(T item);
```

```
    void    ensureCapacity(int index);
```

```
};
```

For example:

Test	Result

ArrayList<int> arr;
 	
    for (int i = 0; i < 10; ++i) {
        arr.add(i);
    }
    arr.removeAt(0);
 	
    cout << arr.toString() << '\n';
    cout << arr.size();
	[1, 2, 3, 4, 5, 6, 7, 8, 9]
	9

ArrayList<int> arr;
 	
    for (int i = 0; i < 10; ++i) {
        arr.add(i);
    }
    arr.removeAt(9);
 	
    cout << arr.toString() << '\n';
    cout << arr.size();
	[0, 1, 2, 3, 4, 5, 6, 7, 8]
	9

ArrayList<int> arr;
 	
    for (int i = 0; i < 10; ++i) {
        arr.add(i);
    }
    arr.removeAt(5);
 	
    cout << arr.toString() << '\n';
    cout << arr.size();
	[0, 1, 2, 3, 4, 6, 7, 8, 9]
	9

Answer: (penalty regime: 0, 0, 0, 0, 0, 100 %)

Reset answer

```
template<class T>
T ArrayList<T>::removeAt(int index){
    if (index < 0 || index >= count) {
        throw std::out_of_range("Index is out of range");
    }

    T removedValue = data[index];

    for (int i = index; i < count - 1; i++) {
        data[i] = data[i + 1];
    }

    count--;
    return removedValue;
}

template<class T>
bool ArrayList<T>::removeItem(T item){
    for (int i = 0; i < count; i++) {
        if (data[i] == item) {
            removeAt(i);
            return true;
        }
    }
    return false;
}

template<class T>
void ArrayList<T>::clear(){
    delete[] data;
    capacity = 5;
    count = 0;
    data = new T[5];
}

```

	Test	Expected	Got	

✓	ArrayList<int> arr;
	
	    for (int i = 0; i < 10; ++i) {
	        arr.add(i);
	    }
	    arr.removeAt(0);
	
	    cout << arr.toString() << '\n';
	    cout << arr.size();
		[1, 2, 3, 4, 5, 6, 7, 8, 9]
		9
			[1, 2, 3, 4, 5, 6, 7, 8, 9]
			9
				✓

	Test	Expected	Got	

✓	ArrayList<int> arr;
	
	for (int i = 0; i < 10; ++i) {
	arr.add(i);
	}
	arr.removeAt(9);
	
	cout << arr.toString() << '\n';
	cout << arr.size();
		[0, 1, 2, 3, 4, 5, 6, 7, 8]
		9
			[0, 1, 2, 3, 4, 5, 6, 7, 8]
			9
				✓

✓	ArrayList<int> arr;
	
	for (int i = 0; i < 10; ++i) {
	arr.add(i);
	}
	arr.removeAt(5);
	
	cout << arr.toString() << '\n';
	cout << arr.size();
		[0, 1, 2, 3, 4, 6, 7, 8, 9]
		9
			[0, 1, 2, 3, 4, 6, 7, 8, 9]
			9
				✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

BÁCH KHOA E-LEARNING WEBSITE LIÊN HỆ

HCMUT

MyBK

BKSI

📍 268 Lý Thường Kiệt, P.14, Q.10, TP.HCM

☎ (028) 38 651 670 - (028) 38 647 256 (Ext: 5258, 5234)

✉ elearning@hcmut.edu.vn