1. Write your own vector class `ITIntVector` to use as a "smart array" with integers.

   Methods of your class will be things like:

   `push_back(int val)` to add an element to the end of the vector

   `at(int i)` to return the element at index position `i` in your vector

   `size()` to return the no of elements in the array

   `capacity()` to return the amount of space currently allocated for the array

   `resize(int i)` to change the size of the array - setting new elements to zero if i is larger than the present size of the array, or chopping off the end of the array if smaller

Extra Note:

Recall, the `at(int i)` method of the STL vector class is implemented just like the array index operator `[]`,with one difference:

the `at()` method does array bounds checking, whereas the `[]` acts as it does for an array - array bounds checking is the programmers responsibility.

For robustness, the method returns a const reference to the value, and is marked as a const method:

```
const type& at(int i) const;
```

Also, in order to facilitate its use as in `v[i] = value`, the `at()` method provides a second version which is not const:

`type& at(int i);` (so we should be able to say `v.at(i) = value`)

2. Re-write your `at()` method (including array bounds checking) in these two flavours.

3. Write a main program to test the `ITIntVector` class you have made.