SORTING\_VISUALIZATION

Visualize the sorting process of bubble sort. The integers are stored in an array. The sorting procedure sorts the integers in an ascending order. After each swap operation, show the result of the integers in the array.

The pseudo code of bubble sort for sorting an array a:

for ( int i = 0; i < n-1; ++i ) {

for ( int j = 0; j < n – i-1; ++j ) {

if ( a[ j ] > a[ j+1 ]) **swap**( a[ j ], a[ j+1 ] )

}

}

Thus the indices i and j must be tracked.

/\*

Generate randomly the elements between mMinValue and mMaxValue.

The total number of elements is mNumElements.

\*/

void SORTING\_VISUALIZATION::reset( )

/\*

Show the system title.

Ask to input the range [mMinValue, mMaxValue] of the elements.

Ask to input the number of elements.

Generate the elements.

\*/

void SORTING\_VISUALIZATION::askForInput( )

{

cout << "//////////////////////////////" << endl;

cout << "SORTING\_VISUALIZATION" << endl;

……

cout << "//////////////////////////////" << endl;

cout << "Press SPACEBAR to visualize the process..." << endl;

cout << "//////////////////////////////" << endl;

……

}

/\*

Two elements whose

indices are (index) and (index+1).

Swap these two elements if their order is incorrect.

\*/

void SORTING\_VISUALIZATION::checkAndSwapTwoElements( int index )

/\*

Perform one swap step for two elements.

The first element is indicated by mCurrentIndex, i.e., j.

\*/

void SORTING\_VISUALIZATION::processOneSwapStep( )

/\*

Return the current element that will be processed, i.e., return j.

\*/

int getCurrentElementIndex( ) const

/\*

Return the index of the last non-handled element, i.e., return n-j.

\*/

int SORTING\_VISUALIZATION::getMaxNonHandledElementIndex( ) const

/\*

Return the number of elements.

\*/

int SORTING\_VISUALIZATION::getNumOfElements( ) const

/\*

Return the element whose index is elementIndex.

\*/

int SORTING\_VISUALIZATION::getElement( int elementIndex ) const