Dmitri Stanchevici
Unit 3
Module 3

Ex. 3.8

The first for loop, beginning with A[0] as the smallest, resulted in the swapping of 51 in position 0 with 24 in position 1. So at this stage the array looks like this
Intermediary: [24, 51, 63, 73, 42, 85, 71, 41, 87, 32]

The second for loop starts with A[1]=51 as the smallest. Upon completion, it determines that 32 in position 9 is the smallest. It is swapped with 51, which thus ends up in position 9, the last position in the array.

Ex. 3.9 Selectin Sort

[51, 24, 63, 73, 42, 85, 71, 41, 87, 32]
Smallest = 51
Pos = 0

i	**smallest after inner loop**	**pos after inner loop**	**array after swap**
0	24	1	[24, 51, 63, 73, 42, 85, 71, 41, 87, 32]
1	32	9	[24, 32, 63, 73, 42, 85, 71, 41, 87, 51]
2	41	7	[24, 32, 41, 73, 42, 85, 71, 63, 87, 51]
3	42	4	[24, 32, 41, 42, 73, 85, 71, 63, 87, 51]
4	51	9	[24, 32, 41, 42, 51, 85, 71, 63, 87, 73]
5	63	7	[24, 32, 41, 42, 51, 63, 71, 85, 87, 73]
6	71	6	[24, 32, 41, 42, 51, 63, 71, 85, 87, 73]
7	73	9	[24, 32, 41, 42, 51, 63, 71, 73, 87, 85]
8	85	9	[24, 32, 41, 42, 51, 63, 71, 73, 85, 87]

Ex. 3.13 Bubble Sort

[51, 24, 63, 73, 42, 85, 71, 41, 87, 32]

Value	51	24	63	73	42	85	71	41	87	32
Index	0	1	2	3	4	5	6	7	8	9

<table>
<tr><th>i</th><th>j</th><th>temp</th><th>array</th></tr>
<tr><td>0</td><td></td><td></td><td></td></tr>
<tr><td></td><td>9</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>32</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>8</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>32</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>7</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>32</td><td>71</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>6</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>32</td><td>85</td><td>71</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>5</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>63</td><td>73</td><td>32</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>4</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>63</td><td>32</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>3</td><td>32</td><td><table><tr><td>Value</td><td>51</td><td>24</td><td>32</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>2</td><td>none</td><td>Same as above</td></tr>
<tr><td></td><td>1</td><td>24</td><td><table><tr><td>Value</td><td>24</td><td>51</td><td>32</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td>1</td><td></td><td></td><td></td></tr>
<tr><td></td><td>9</td><td>none</td><td>Same as above</td></tr>
<tr><td></td><td>8</td><td>41</td><td><table><tr><td>Value</td><td>24</td><td>51</td><td>32</td><td>63</td><td>73</td><td>42</td><td>85</td><td>41</td><td>71</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
<tr><td></td><td>7</td><td>41</td><td><table><tr><td>Value</td><td>24</td><td>51</td><td>32</td><td>63</td><td>73</td><td>42</td><td>41</td><td>85</td><td>71</td><td>87</td></tr><tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr></table></td></tr>
</table>

	6	41	Value	24	51	32	63	73	41	42	85	71	87
			Index	0	1	2	3	4	5	6	7	8	9
	5	41	Value	24	51	32	63	41	73	42	85	71	87
			Index	0	1	2	3	4	5	6	7	8	9
	4	41	Value	24	51	32	41	63	73	42	85	71	87
			Index	0	1	2	3	4	5	6	7	8	9
	3	None	Same										
	2	32	Value	24	32	51	41	63	73	42	85	71	87
			Index	0	1	2	3	4	5	6	7	8	9
2													
	9	none	Same										
	8	71	Value	24	32	51	41	63	73	42	71	85	87
			Index	0	1	2	3	4	5	6	7	8	9
	7	None	Same										
	6	42	Value	24	32	51	41	63	42	73	71	85	87
			Index	0	1	2	3	4	5	6	7	8	9
	5	42	Value	24	32	51	41	42	63	73	71	85	87
			Index	0	1	2	3	4	5	6	7	8	9
	4	None	Same										
	3	41	Value	24	32	41	51	42	63	73	71	85	87
			Index	0	1	2	3	4	5	6	7	8	9
3													
	9	None	Same										
	8	None	same										
	7	71	Value	24	32	41	51	42	63	71	73	85	87
			Index	0	1	2	3	4	5	6	7	8	9
	6	None	same										
	5	None	Same										
	4	42	Value	24	32	41	42	51	63	71	73	85	87
			Index	0	1	2	3	4	5	6	7	8	9

4			
	9	none	same
	8	none	same
	7	none	same
	6	none	same
	5	none	same
5			
	9	none	same
	8	none	same
	7	none	same
	6	none	Same
6			
	9	none	same
	8	none	same
	7	none	same
7			
	9	none	same
	8	none	Same
8			
	9	none	Same

Ex. 3.16 Insertion Sort

A	51	24	63	73	42	85	71	41	87	32
B										
Index	0	1	2	3	4	5	6	7	8	9

<table>
<tr><th>i (0- 9)</th><th>j (for posInB) (0- (i- 1))</th><th>posInB</th><th>j (for shifting right) ((i+1) – posInB)</th><th>arrays</th></tr>
<tr><td>0</td><td></td><td>0</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>1</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>51</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td>1</td><td></td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>2</td><td></td></tr>
<tr><td></td><td></td><td></td><td>1</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>51</td><td>51</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td>2</td><td></td><td>2</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>63</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td>3</td><td></td><td>3</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td></td><td></td><td></td></tr>
<tr><td></td><td>2</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>4</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>63</td><td>73</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td>4</td><td></td><td>4</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>5</td><td></td></tr>
<tr><td></td><td></td><td></td><td>4</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>63</td><td>73</td><td>73</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>3</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>63</td><td>63</td><td>73</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>2</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>51</td><td>63</td><td>73</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>42</td><td>51</td><td>63</td><td>73</td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
</table>

				Index	0	1	2	3	4	5	6	7	8	9
5		5												
	0													
	1													
	2													
	3													
	4													
			6											
				A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	73	85				
				Index	0	1	2	3	4	5	6	7	8	9
6		6												
	0													
	1													
	2													
	3													
	4	4												
			7											
			6	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	73	85	85			
				Index	0	1	2	3	4	5	6	7	8	9
			5	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	73	73	85			
				Index	0	1	2	3	4	5	6	7	8	9
				A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	71	73	85			
				Index	0	1	2	3	4	5	6	7	8	9
7		7												
	0													
	1	1												
			8											
			7	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	71	73	85	85		
				Index	0	1	2	3	4	5	6	7	8	9
			6	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	71	73	73	85		
				Index	0	1	2	3	4	5	6	7	8	9
			5	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	71	71	73	85		
				Index	0	1	2	3	4	5	6	7	8	9
			4	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	63	63	71	73	85		
				Index	0	1	2	3	4	5	6	7	8	9
			3	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	51	51	63	71	73	85		
				Index	0	1	2	3	4	5	6	7	8	9
			2	A	51	24	63	73	42	85	71	41	87	32
				B	24	42	42	51	63	71	73	85		
				Index	0	1	2	3	4	5	6	7	8	9
				A	51	24	63	73	42	85	71	41	87	32
				B	24	41	42	51	63	71	73	85		
				Index	0	1	2	3	4	5	6	7	8	9
8		8												
	0													
	1													
	2													
	3													
	4													

<table>
<tr><td></td><td>5</td><td></td><td></td><td></td></tr>
<tr><td></td><td>6</td><td></td><td></td><td></td></tr>
<tr><td></td><td>7</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>9</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td>9</td><td></td><td>9</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td></td><td></td><td></td></tr>
<tr><td></td><td>1</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td>10</td><td></td></tr>
<tr><td></td><td></td><td></td><td>9</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>8</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>7</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>6</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>71</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>5</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>63</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>4</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>51</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>3</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>42</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td>2</td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>41</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>32</td><td>41</td><td>42</td><td>51</td><td>63</td><td>71</td><td>73</td><td>85</td><td>87</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table>

Ex. 3.19

A	51	24	63	73	42	85	71	41	87	32
B										
Index	0	1	2	3	4	5	6	7	8	9

<table>
<tr><td>i</td><td>j</td><td>Arrays</td></tr>
<tr><td>0</td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>51</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
<tr><td></td><td>0</td><td></td></tr>
<tr><td>1</td><td></td><td><table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>51</td><td>24</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td>1</td><td>
<table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table>
</td></tr>
<tr><td></td><td>0</td><td></td></tr>
<tr><td>2</td><td></td><td>
<table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>63</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table>
</td></tr>
<tr><td></td><td>2</td><td></td></tr>
<tr><td>3</td><td></td><td>
<table>
<tr><td>A</td><td>51</td><td>24</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>B</td><td>24</td><td>51</td><td>63</td><td>73</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table>
</td></tr>
<tr><td></td><td>3</td><td></td></tr>
</table>

Ex. 3.20

A	51	24	63	73	42	85	71	41	87	32
Index	0	1	2	3	4	5	6	7	8	9

<table>
<tr><td>i</td><td>j</td><td>Arrays</td></tr>
<tr><td>0</td><td></td><td></td></tr>
<tr><td></td><td>0</td><td></td></tr>
<tr><td>1</td><td></td><td></td></tr>
<tr><td></td><td>1</td><td>
<table>
<tr><td>A</td><td>24</td><td>51</td><td>63</td><td>73</td><td>42</td><td>85</td><td>71</td><td>41</td><td>87</td><td>32</td></tr>
<tr><td>Index</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr>
</table>
</td></tr>
<tr><td></td><td>0</td><td></td></tr>
<tr><td>2</td><td></td><td></td></tr>
<tr><td></td><td>2</td><td>same</td></tr>
<tr><td>3</td><td></td><td></td></tr>
<tr><td></td><td>3</td><td>same</td></tr>
</table>

Ex. 3.23

Why is a copy of the array being made each time?

The method copy (int[] A) copies the testData array by value, creating a new array in the heap memory. Thus when the sorting method changes this copied array, the array pointed to by testData remains unchanged and thus ready for the next sorting test. The copied array in C gets sorted after every test, so it must be re-copied for the next test so that it points to the array as it is presented in testData.

Why is Bubble-Sort much slower than the other two?
Here are the results of the comparison:
Selection: count=0
Time taken by SelectionSort: 2702
Bubble: count=0
Time taken by BubbleSort: 13817
Insertion: count=0
Time taken by InsertionSort: 1410

While InsertionSort is almost twice as fast as SelectionSort, BubbleSort is by far the slowest, almost ten times as slow as InsertionSort and three to four times slower than SelectionSort.

Looking at my tracing tables, I see that all three methods should result in times that are roughly proportional to n:

- SelectionSort: f(n) = n * n/2
- BubbleSort: f(n) = n * n/2
- InsertionSort: f(n) = n * n/5

While the factor in SelectionSort and BubbleSort is always n/2, it seems that the factor in InsertionSort can be potentially smaller, resulting in faster execution.

This analysis explains why InsertionSort may be the fastest.

SelectionSort and BubbleSort have the same time complexity, but the latter is slower because it must perform more swaps. In SelectionSort, the number of swaps is equal to n-1. In BubbleSwap, depending on the case, there may be as many as n*n/2 swaps.

I have added a swapCount for each method in the given program. Here are the results for n=100:

Selection: swapCount=99
Bubble: swapCount=2539
Insertion: swapCount=2539

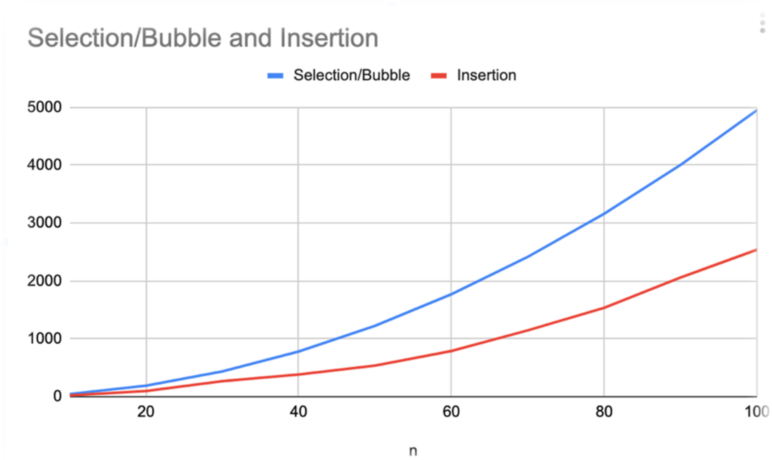
In conclusion, BubbleSort is the slowest because with n(n/2) iterations, it must perform approx. 25n swaps.

Ex. 3.25

Sort Comparison

Comparison of Selection, Bubble, and Insertion sorting methods based on the number of comparisons they perform.

The size of the array (n) is presented in the x-axis.
The number of comparisons is presented in the y-axis.



n	Selection	Bubble	Insertion
10	45	45	20
20	190	190	97
30	435	435	267
40	780	780	382
50	1225	1225	537
60	1770	1770	789
70	2415	2415	1145
80	3160	3160	1535
90	4005	4005	2059
100	4950	4950	2539

Ex. 3.25

Trace through quickSortPartion for the first two calls.

CALL to quickSortPartion 1

Parameters:

- data = 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
- left = 0
- right = data.length-1 = 9

left != right -> Continue with the execution inside the method.

partionElement = data[right] = 32
currentSwapPosition = right = 9
for i=right-1=8 to i>=left=0

i	data[i] > partitionElement (32)	currentSwapPosition	data after swap(data, currentSwapPosition, i)
8	data[8] == 87 > 32 -> TRUE	8	data after swap(data, 8, 8) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
7	data[7] == 41 > 32 -> TRUE	7	data after swap(data, 7, 7) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
6	data[6] == 71 > 32 -> TRUE	6	data after swap(data, 6, 6) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
5	data[5] == 85 > 32 -> TRUE	5	data after swap(data, 5, 5) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
4	data[4] == 42 > 32 -> TRUE	4	data after swap(data, 4, 4) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
3	data[3] == 73 > 32 -> TRUE	3	data after swap(data, 4, 4) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
2	data[2] == 63 > 32 -> TRUE	2	data after swap(data, 4, 4) NO CHANGE 51, 24, 63, 73, 42, 85, 71, 41, 87, 32
1	data[1] == 24 ! > 32 -> FALSE		
0	data[0] == 51 > 32 -> TRUE	1	data after swap(data, 1, 0) CHANGE 24, 52, 63, 73, 42, 85, 71, 41, 87, 32

End for loop

Data after data after swap(data, currentSwapPosition=1, right=9):
24, 32, 63, 73, 42, 85, 71, 41, 87, 52
Return currentSwapPosition=1

Back in quickSortRecursive

partitionPosition = 1
// Recursive on left side:
quickSortRecursive (data, left=0, partitionPosition-1=0)

Back in quickSortRecursive
Parameters:

data = 24, 32, 63, 73, 42, 85, 71, 41, 87, 52
left=0
right=0

left == right -> return left=0;

Back in quickSortRecursive

I assume that partitionPosition = 1, unaffected by the above call recurse on left side
// Recurse on right side:
quickSortRecursive (data, partitionPosition+1=2, right=9)

Back in quickSortRecursive
Parameters:

- data = 24, 32, 63, 73, 42, 85, 71, 41, 87, 52
- left=2
- right=9

left < right -> continue with the execution
partition Position = quickSortPartition (data, 2, 9)

CALL 2

Parameters:

- data = 24, 32, 63, 73, 42, 85, 71, 41, 87, 52
- left = 2
- right = 9

left != right -> Continue with the execution inside the method.

partitionElement = data[right] = 52
currentSwapPosition = right = 9
for i=right-1=8 to i>=left=2

i	data[i] > partitionElement (52)	currentSwapPosition	data after swap(data, currentSwapPosition, i)
8	data[8] == 87 > 52 -> TRUE	8	data after swap(data, 8, 8) NO CHANGE 24, 32, 63, 73, 42, 85, 71, 41, 87, 52
7	data[7] == 41 !> 52 -> FALSE		
6	data[6] == 71 > 52 -> TRUE	7	data after swap(data, 7, 6) 24, 32, 63, 73, 42, 85, 41, 71, 87, 52

5	data[5] == 85 > 52 -> TRUE	6	data after swap(data, 6, 5) 24, 32, 63, 73, 42, 41, 85, 71, 87, 52
4	data[4] == 42 !> 52 -> FALSE		
3	data[3] == 73 > 52 -> TRUE	5	data after swap(data, 5, 3) 24, 32, 63, 41, 42, 73, 85, 71, 87, 52
2	data[2] == 63 > 52 -> TRUE	4	data after swap(data, 4, 2) 24, 32, 42, 41, 63, 73, 85, 71, 87, 52

End for loop

Data after data after swap(data, currentSwapPosition=4, right=9):
24, 32, 42, 52, 63, 73, 85, 71, 87, 41
Return currentSwapPosition=4

Ex. 3.29

Sort Comparison

Comparison of Selection, Bubble, Insertion, and Quick sorting methods based on the number of comparisons they perform.

The size of the array (n) is presented in the x-axis.
The number of comparisons is presented in the y-axis.

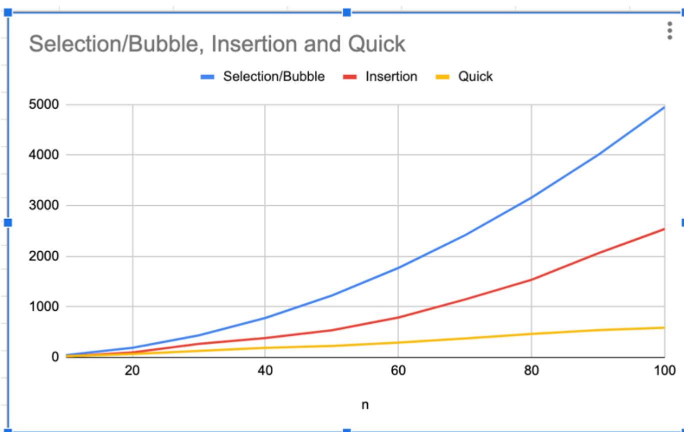


Table with numeric values.

n	Selection/Bubble	Insertion	Quick
10	45	20	25
20	190	97	65
30	435	267	128
40	780	382	190
50	1225	537	225
60	1770	789	292
70	2415	1145	375
80	3160	1535	463
90	4005	2059	540
100	4950	2539	588

Ex. 3.30

Comparison of the sorting methods' speeds

Here are the results:

- Selection sort time: 35
- Bubble sort time: 111
- Insertion sort time: 19
- Quick sort time: 3

It is obvious that Quick Sort is by far the fastest of the four methods. It is appr. six times faster than Insertion, 11 times faster than Selection, and 33 times faster than Bubble.