Lab 5

Team members: Kerim Amansaryyev,
Muneeb Ahmed,
Mahri Ilmedova.

Question 1:

Bubble Sort:

Iteration	Inversions	# Inversions
0	(34,8), (34,32), (34,21), (64,51), (64,32), (64,21), (51,32), (51,21), (32,21)	9
1	(8,34), (32,34), (21,34), (51,64), (32,64), (21,64), (32,51), (21,51)	8
2	(8,32), (21,32), (32,51), (21,51)	4
3	(8,21), (21,32)	2
4	(8,21)	1

Total number of inversions for Bubble Sort = 24

Selection Sort:

Iteration	Inversions	# Inversions
0	(34,8), (34,32), (34,21), (64,51), (64,32), (64,21), (51,32), (51,21), (32,21)	9
1	(8,34), (32,34), (21,34), (51,64), (32,64), (21,64), (32,51), (21,51)	8
2	(8,32), (21,32), (32,51), (21,51)	4
3	(8,21), (21,32)	2
4	(8,21)	1

Total number of inversions for Selection Sort = 24

Insertion Sort:

Iteration	Inversions	# Inversions
0	(34,8), (34,32), (34,21), (64,51), (64,32), (64,21), (51,32), (51,21), (32,21)	9
1	(8,34), (32,34), (21,34), (51,64), (32,64), (21,64), (32,51), (21,51)	8
2	(8,32), (21,32), (32,51), (21,51)	4
3	(8,21), (21,32)	2
4	(8,21)	1

Total number of inversions for Insertion Sort = 24

Question 2:

(a) add, add, add, add, clear.

We need to consider each add operation as 2, so that clear operation we can skip it, because we already calculated it’s cost when we add items. So

add – 2 operations
clear – 0 operations

(b) 

16 – add operations costs – 16
resize operation costs – 3k = 3 * 32 = 96

16+96 = 112 – of all operations

112/16 = all operations/add operations = 7 operations per each array item.

Now we see that each item in an array costs us 7 operations. So resizing an array will cost per each array push operation 7 operations.

Question 3:

1. **Amortized Cost of Add (Φ(add)):**
   Suppose we triple the size during a resize, so the previous size was 3. The total cost of the previous array (3 elements) was 3, and the new array (9 elements) has a cost of 9 (copying each element once). So, the total cost is 3 + 9 = 12. The amortized cost per operation is 12/3 = 4.

2. **Amortized Cost of Resize (Φ(resize)):**
   The resize operation is performed when the array is full (9 elements). The cost of resizing is 9 (copying each element once). The amortized cost per operation is 9/1 = 9.

3. **Amortized Cost Analysis for a Sequence of n Operations:**
   Let's consider a sequence of n operations (some add, some resize). The total amortized cost for these operations would be n * (Φ(add) + Φ(resize)), which is 4n + 9n = 13n. The average cost per operation is then (13n) / n = 13. As n grows, the average cost per operation remains constant (13), demonstrating that the amortized cost is constant time.