

Solutions

Each answer shows the steps, then the **result**.

2.1

- pass 1: 1 swapped with 7: {1, 4, 9, 7, 5}
- pass 2: 4 already in place: {1, 4, 9, 7, 5}

2.2

- pass 1: 4 inserted before 7: {4, 7, 9, 1, 5}
- pass 2: 9 stays: {4, 7, 9, 1, 5}

2.3

- $4 + 3 + 2 + 1 = 10$

2.4

- **insertion sort**
- each key is compared once with the value before it and stays; selection sort still scans the whole unsorted part every pass

3.1 B

- the smallest value, 1, was swapped with the first; the rest is unchanged

3.2 B

- pass 1: {1, 4, 3, 2}; pass 2 inserts 3: {1, 3, 4, 2}

3.3 B

- reversed order: every key shifts past every sorted value

3.4

(a)

- pass 1: {4, 9, 20, 9, 15}; pass 2: {4, 9, 20, 9, 15}
- pass 3: {4, 9, 9, 20, 15}
- pass 4: {4, 9, 9, 15, 20}

(b)

- $4 + 3 + 2 + 1 = 10$

(c)

- **descending** order: largest first

(d)

- in an almost-sorted array most keys are already in place, so insertion sort does about **one comparison** for them and few shifts

- selection sort makes **all** its comparisons whatever the order