**Exercise 3.1** *Comparison of Sorting Algorithms.*

Let $A[1..n]$ be an array. Consider the following naive implementations of the sorting algorithms *bubble sort*, *insertion sort*, *selection sort*, and *quicksort*. These algorithms are called with the parameters $l = 1$ and $r = n$ to sort $A$ in ascending order.

```java
public void bubbleSort(int[] A, int l, int r) {
    for(int i=r; i>l; i--)
        for(int j=l; j<i; j++)
            if(A[j]>A[j+1])
                swap(A, j, j+1);
}

public void insertionSort(int[] A, int l, int r) {
    for(int i=l; i<r; i++) {
        int minJ = i;
        for(int j=i+1; j<=r; j++)
            if(A[j]<A[minJ])
                minJ = j;
        if(minJ != i)
            swap(A, i, minJ);
    }
}

public void selectionSort(int[] A, int l, int r) {
    for(int i=l; i<r; i++) {
        int minJ = i;
        for(int j=i+1; j<=r; j++)
            if(A[j]<A[minJ])
                minJ = j;
        if(minJ != i)
            swap(A, i, minJ);
    }
}

public void quicksort(int[] A, int l, int r) {
    if(l<r) {
        int i=l+1, j=r;
        do {
            while(i<j && A[i]<=A[l]) i++;
            while(i<=j && A[j]>=A[l]) j--;
            if(i<j) swap(A, i, j);
        } while(i<j);
        swap(A, l, j);
        quicksort(A, l, j-1);
        quicksort(A, j+1, r);
    }
}
```

The function `swap(A, i, j)` exchanges (swaps) the elements $A[i]$ and $A[j]$. For each of the above algorithms, estimate asymptotically both the minimum and the maximum number of performed swaps and comparisons of elements of $A$. For each of these cases, give an example sequence of the numbers $1, 2, \ldots, n$ for which the particular case occurs. The sequence should be described in such a way that any $n$ can be chosen arbitrarily (i.e., the descending sorted sequence can be described as $n, n-1, \ldots, 1$). Enter your results in a table of the following form.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Comparisons</th>
<th>Input sequence</th>
<th>Permutations</th>
<th>Input sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>bubbleSort</td>
<td>min</td>
<td>max</td>
<td></td>
<td></td>
</tr>
<tr>
<td>insertionSort</td>
<td>min</td>
<td>max</td>
<td></td>
<td></td>
</tr>
<tr>
<td>selectionSort</td>
<td>min</td>
<td>max</td>
<td></td>
<td></td>
</tr>
<tr>
<td>quicksort</td>
<td>min</td>
<td>max</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Please turn over.*
Exercise 3.2  Algorithm Design: Sums of Numbers.

Let $A[1..n]$ be an array of natural numbers. For each of the following problems, provide an algorithm that is as efficient as possible, and determine its running time in the worst case.

a) Given a natural number $z$, does the array $A$ contain two entries $a$ and $b$ such that $a + b = z$?

b) Suppose that $A$ is sorted in ascending order. How efficiently can the problem from a) be solved now? Hint: In this case it is possible to achieve a better running time than in the previous case.

c) Does the array $A$ contain any three different entries $a$, $b$ and $c$ such that $a + b = c$?

Exercise 3.3  Blum’s Median-of-Medians Strategy.

We consider finding the median of a sequence using the median-of-medians strategy from the lecture (see Chapter 3.1 in the book). We will consider only the highest level of recursion, so only the very first invocation of the procedure $\text{Auswahl}$ that determines the $i$-th smallest element with $i = \lceil \frac{N}{2} \rceil$.

a) Given the following sequence

\[7, 12, 17, 3, 10, 1, 6, 2, 4, 8, 11, 9, 9, 6, 5, 14, 20, 13, 1, 7, 19, 8,\]

provide the two sequences on which $\text{Auswahl}$ invokes itself recursively.

b) In general, how long are each of the two sequences used in the two recursive calls of the procedure $\text{Auswahl}$ for $i = \lceil \frac{N}{2} \rceil$ at least and at most?

Hint: For the analysis you may assume that all numbers in the input sequence are pairwise distinct.