

Algorithmen und Datenstrukturen

Aufgabe 1 **Stack und Queue**

a) Gegeben sei ein Stack S :

$\leftrightarrow (5, 8, 2, 3, 9)$

Geben Sie jeweils den Zustand von S und die Rückgabe an, wenn folgende Operationen ausgeführt werden:

- (i) $\text{pop}(S)$
- (ii) $\text{push}(S, 8)$
- (iii) $\text{top}(S)$
- (iv) $\text{push}(S, 1)$
- (v) $\text{pop}(S)$

b) Gegeben sei eine Queue Q :

$\rightarrow (0, 9, 1, 3, 4) \rightarrow$

Geben Sie jeweils den Zustand von Q und die Rückgabe an, wenn folgende Operationen ausgeführt werden:

- (i) $\text{dequeue}(Q)$
- (ii) $\text{enqueue}(Q, 7)$
- (iii) $\text{enqueue}(Q, 5)$
- (iv) $\text{dequeue}(Q)$
- (v) $\text{isEmpty}(Q)$

Aufgabe 2 **Insertion Sort**

Sortieren Sie folgende Arrays mit *Insertion Sort*!

a) $(1, 9, 4, 3, 7)$

b) $(2, 7, 5, 1, 8)$

Aufgabe 3 **Verifikation der Multiplikation durch Addition**

Verifizieren Sie nachfolgende Implementierung der Multiplikation durch wiederholte Addition!

Input: Ganze Zahlen a und b

Output: Produkt $c = a \cdot b$

Multiply (a, b) :

```
if (a == 0  $\vee$  b == 0) {  
    c = 0;  
}
```

```
else if (a < 0) {
    c = -Multiply(-a, b);
}
else {
    c = 0;
    i = 1;
    while (i ≤ a) {
        c = c + b;
        i = i + 1;
    }
}
return c;
```

- Formulieren Sie Vor- und Nachbedingung der Methode!
- Formulieren Sie eine Schleifeninvariante!
- Verifizieren Sie unter Nutzung der zuvor erarbeiteten logischen Bedingungen die Korrektheit des Programmcodes.

Aufgabe 4 Verifikation von Selection Sort

In Pseudocode lautet das Sortierverfahren *Selection Sort* (mit Hilfsfunktionen **IndexOfMin** und **Swap**) wie folgt:

Input: Array $A[0..n-1]$ von $n \geq 0$ natürlichen Zahlen

Output: Array A aufsteigend sortiert

SelectionSort(A):

```
for j = 0 to n - 1 {
    i = IndexOfMin(A, j);
    if (j ≠ i) {
        Swap(A, i, j);
    }
}
```

Input: Array $A[0..n-1]$ von $n > 0$ natürlichen Zahlen; Startindex j

Output: Index $i \geq j$ des kleinsten Elements im Rest $A[j..n-1]$

IndexOfMin(A, j):

```
i = j;
Ai = A[j];
for k = j + 1 to n - 1 {
    if (A[k] < Ai) {
        i = k;
        Ai = A[k];
    }
}
return i;
```

Input: Array $A[0..n-1]$ von $n > 0$ natürlichen Zahlen; Indices i, j

Output: Array A mit Zellen i und j vertauscht

Swap(A, i, j):

```
k = A[j];
A[j] = A[i];
A[i] = k;
```

- a) Geben Sie für alle drei Funktionen jeweils Vor- und Nachbedingungen an!
- b) Zeigen Sie die partielle Korrektheit der Funktion **SelectionSort**! Rechtfertigen Sie Ihre Wahl der Schleifeninvariante!
- c) Zeigen Sie diese auch für die Funktion **IndexOfMin**!