Programmieren und Software-Engineering II
Übung 13

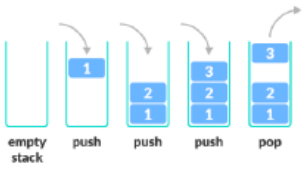
Name: _____ **Klasse:** _____ **Datum:** _____

Lernziele:

- Objektorientierte Programmierung
- Klassen/Objekte: Erstellen und Verwenden
- Dynamische Datenstrukturen

Aufgabe 1: Stack (HUE_11_01_Stack)

Der Stapelspeicher (Stack) ist eine wichtige dynamische Datenstruktur.
Der Stack funktioniert nach dem *LIFO (Last-In-First-Out)* Prinzip.



Erstelle eine Klasse Stack für Ganzzahlen (Integer) und implementiere folgende Methoden:

- `push`: Erweitert das Array um eine Stelle, die neue Zahl wird vorne angefügt.
- `pop`: Verkürzt das Array um eine Stelle, die Zahl an der ersten Stelle (Index 0) wird zurückgegeben.
- `isEmpty`: Prüft, ob der Stack leer ist.
- `isFull`: Prüft, ob der Stack voll ist. (Legen sie die maximale Größe des Stack als Konstante fest.)
- `peek`: Liefert den obersten Wert am Stack, ohne ihn zu entfernen.

Aufgabe 2: Queue (HUE_11_02_Queue)

Eine weitere wichtige Datenstruktur in der Informatik ist die Queue (Warteschlange). Sie funktioniert nach dem F*IFO (First-In-First-Out)* Prinzip.

Erstelle eine Klasse Queue um eine Warteschlange zu simulieren. In der Queue werden Integer-Objekte gespeichert.

- `enqueue`: Erweitert das Array um eine Stelle, die neue Zahl wird vorne angefügt.
- `dequeue`: Verkürzt das Array um eine Stelle, die Zahl an der letzten Stelle wird zurückgegeben.
- `isEmpty`: Prüft, ob die Queue leer ist.
- `isFull`: Prüft, ob die Queue voll ist. (Legen Sie die maximale Länge der Queue als Konstante fest.)
- `peek`: Liefert den nächsten Wert in der Queue, ohne ihn zu entfernen.

Aufgabe 3: LinkedQueue (HUE_11_03_LinkedQueue)

Implementiere die Queue erneut OHNE ein Array für die Datenhaltung zu verwenden. Verlinke lediglich die Objekte miteinander. Erstelle dazu eine Klasse `QueueItem`, das eine Zahl speichert und zusätzlich eine Referenz zum Nachfolger (`QueueItem next`) beinhaltet.

Stelle die gleichen Methoden wie in Aufgabe 2 zur Verfügung.