

1 Queue

1.1 Einleitung

Eine Queue ist eine Warteschlange. In ihr werden Daten in einer bestimmten Reihenfolge abgelegt. Ähnlich wie der Stack braucht auch die Queue nicht viele Methoden, um ihre volle Funktionalität zu erhalten.

Mit Hilfe der Methode *add()* kann ein neues Element der Queue hinzugefügt werden. Die Queue arbeitet nach dem FIFO-Prinzip. FIFO steht für „First In First Out“ und bedeutet, dass das zuerst eingefügte Element auch zuerst wieder ausgegeben wird. Das heißt, neu eingefügte Elemente werden an das Ende der Warteschlange gesetzt.

Um das vorderste Element aus der Queue auszulesen wird die Methode *peek()* benötigt. Diese liest das Element lediglich, wobei die Methode *remove()* das Element nicht nur liest, sondern es außerdem aus der Queue entfernt.

Wie bei jeder Datenstruktur ist es auch bei der Queue sinnvoll, eine Methode *size()* zu implementieren, die die Länge der Warteschlange zurückliefert. In Java ist die Queue ein Interface, das sich auf die Super-Interfaces *Collection* und *Iterable* zurückführen lässt.

1.2 Methodenbeschreibung

In diesem Abschnitt werden die wichtigsten Methoden der Queue beschrieben, die von Java zur Verfügung gestellt werden.

- *add(E item)*
 - fügt das als Argument übergebene Element der Queue an letzter Stelle hinzu
- *peek()*
 - liefert das vorderste Element der Queue zurück ohne es aus der Warteschlange zu löschen
- *remove()*
 - liefert das vorderste Element der Queue zurück und löscht es aus der Warteschlange
- *size()*
 - gibt die Anzahl der Elemente der Queue zurück

1.3 Effizienzanalyse

Graph einfügen und Effizienz ausrechnen. Für einzelne Probleme anwenden.

1.4 Schlussfolgerung

hier die resultierenden Stärken und Schwächen rausschreiben.