cis112

Generic: Stack and Queue

BBBF

Yeditepe University

v2025-03-22

Content

Introduction

Introduction

- We have explored the array-based implementations of the **Stack** and **Queue** data structures.
- In the Java API, these structures are also available as generic types, adhering to their respective Abstract Data Type (ADT) definitions.
- However, we will first examine another important ADT: the **Set**.

Set in Java API

Set in Java API

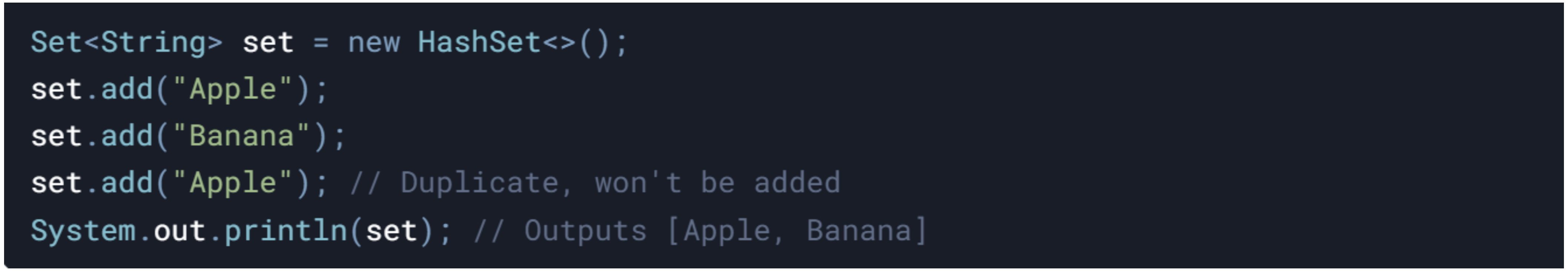
- A Set is a collection of unique elements (no duplicates).

- **Properties:**
  - Does not allow duplicate elements.
  - Does not maintain insertion order.

- **Common Implementations:**
  - **HashSet:** Unordered, fast operations.
  - **TreeSet:** Sorted, based on natural ordering or a comparator.
  - **LinkedHashSet:** Maintains insertion order.

Set in Java API

- **Key methods:**
  - **add(E item):** Adds an element to the set.
  - **remove(Object o):** Removes an element from the set.
  - **contains(Object o):** Checks if the set contains an element.
  - **isEmpty():** Checks if the set is empty.



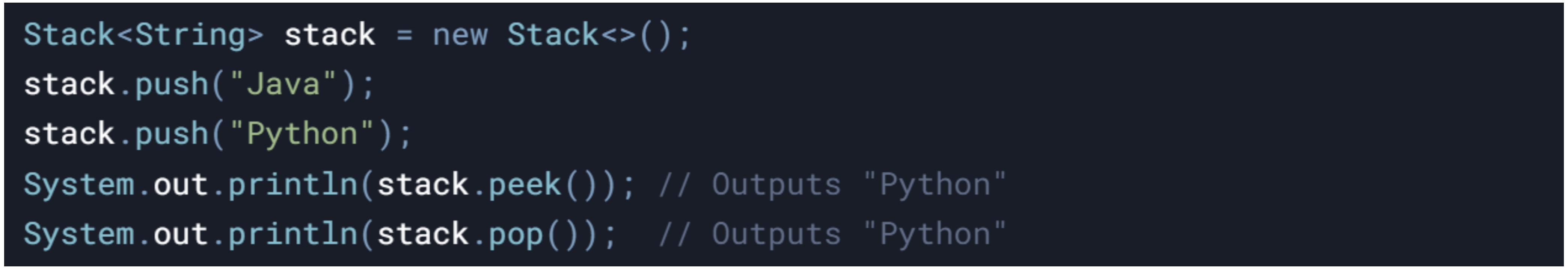
```java
Set<String> set = new HashSet<>();
set.add("Apple");
set.add("Banana");
set.add("Apple"); // Duplicate, won't be added
System.out.println(set); // Outputs [Apple, Banana]
```

Stack in Java API

Stack in Java API

- **Class:** java.util.Stack

- **Key Methods:**
  - **push(E item):** Adds an element to the top of the stack.
  - **pop():** Removes and returns the top element.
  - **peek():** Returns the top element without removing it.
  - **isEmpty():** Checks if the stack is empty.



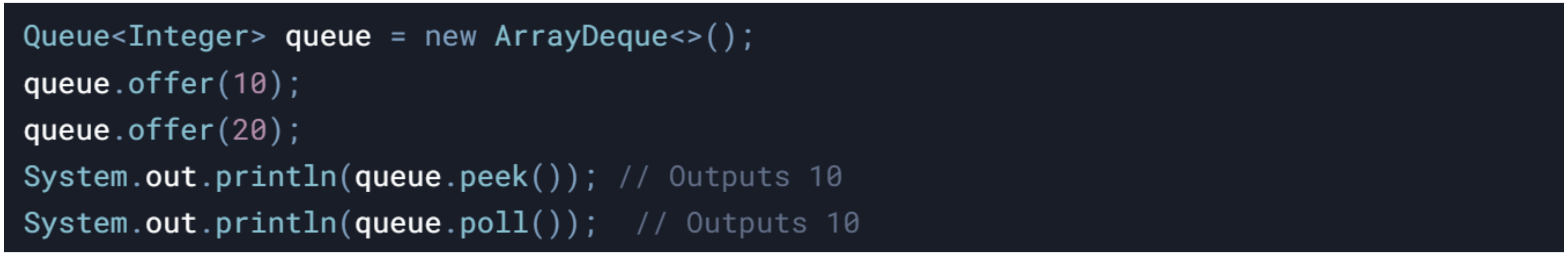
```java
Stack<String> stack = new Stack<>();
stack.push("Java");
stack.push("Python");
System.out.println(stack.peek()); // Outputs "Python"
System.out.println(stack.pop());  // Outputs "Python"
```

Queue in Java API

Queue in Java API

- **Interfaces and Classes:**
  - **Queue** interface.
  - **Implementations: ArrayDeque** and **LinkedList**.

- **Key Methods:**
  - **add(E item)** or **offer(E item):** Adds an element to the queue.
  - **remove()** or **poll():** Removes and returns the front element.
  - **element()** or **peek():** Returns the front element without removing it.
  - **isEmpty():** Checks if the queue is empty.



```java
Queue<Integer> queue = new ArrayDeque<>();
queue.offer(10);
queue.offer(20);
System.out.println(queue.peek()); // Outputs 10
System.out.println(queue.poll());  // Outputs 10
```

Exercises

Set

Remove Duplicates Using a Set

- Write a Java program to remove duplicates from a list using a Set.

Set

Find Common Elements in Two Lists Using a Set

- Write a Java program to find common elements between two lists using a Set.

Stack

Reverse a String Using a Stack

- Write a Java program to reverse a string using a stack.

Queue

Implement a Queue Using Two Stacks

- Write a Java program to implement a queue using two stacks.

References

- [1] Oracle Java Documentation.
- [2] GeeksforGeeks
- [3] "Effective Java" by Joshua Bloch (Best practices for using Java collections)
- [4] "Java: The Complete Reference" by Herbert Schildt (Comprehensive guide to Java, including collections)
- [5] https://docs.oracle.com/javase/8/docs/api/
- [6] https://docs.oracle.com/javase/tutorial/collections/