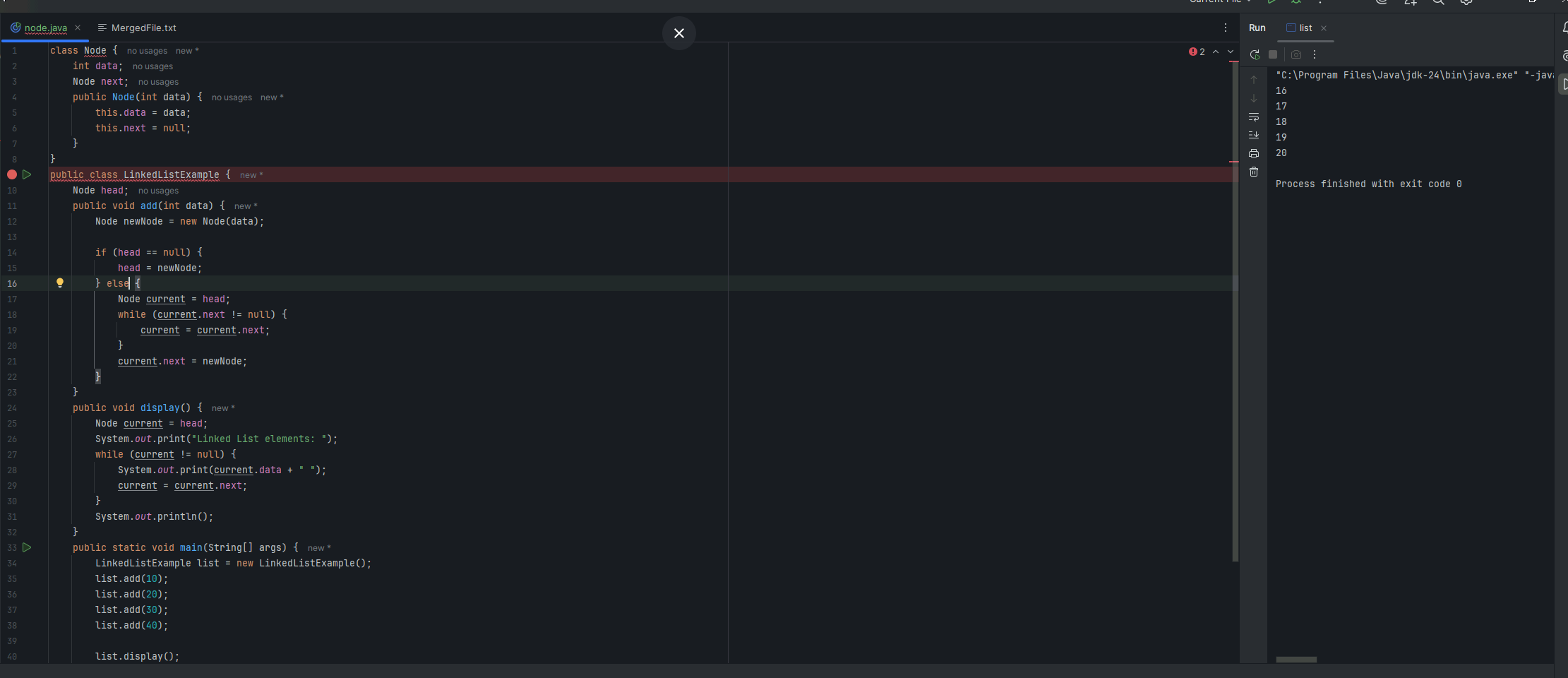
Task 01 :



Task 02:

What is Traversing in a Linked List?

Traversing means visiting each node in the linked list exactly once from the beginning to the end to perform some operation, usually to access or display all the elements stored in the list.

Since linked lists are made up of nodes where each node points to the next one, we can't jump directly to an element by index (like arrays). To visit all nodes, we start at the head node and follow the next pointer to the next node repeatedly until you reach the end (null).

Why do we traverse a linked list?

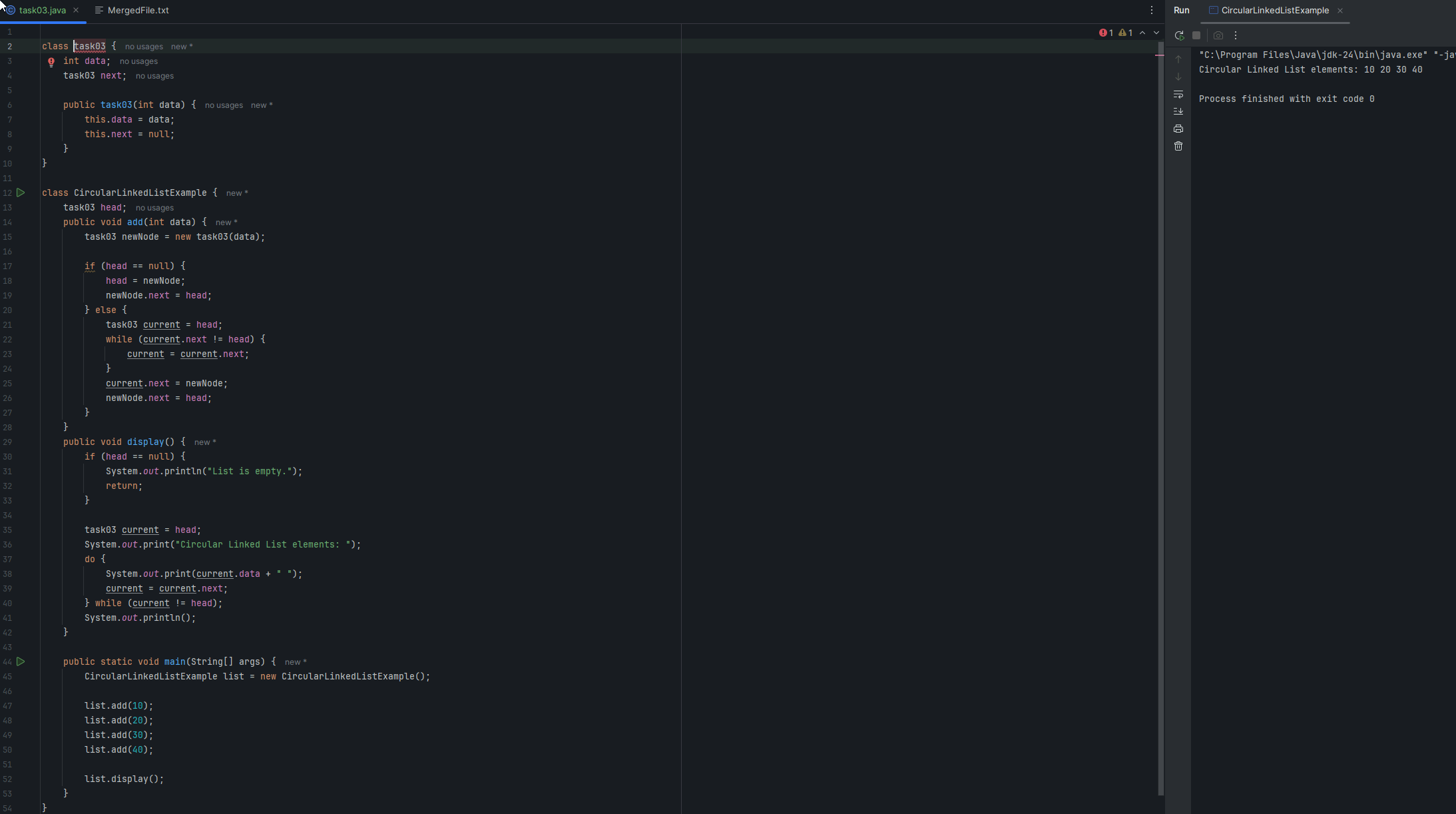
>> To display the data stored in each node.

>> To search for a node with specific data.

>> To perform operations like modification or deletion on every node.

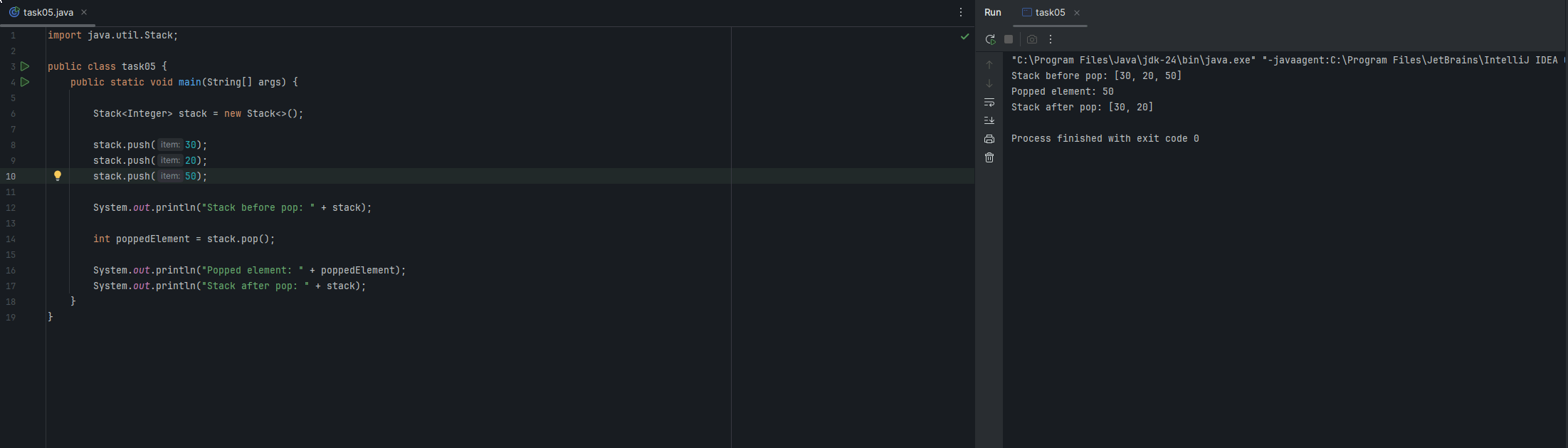
>> To count the number of nodes.

Task03:

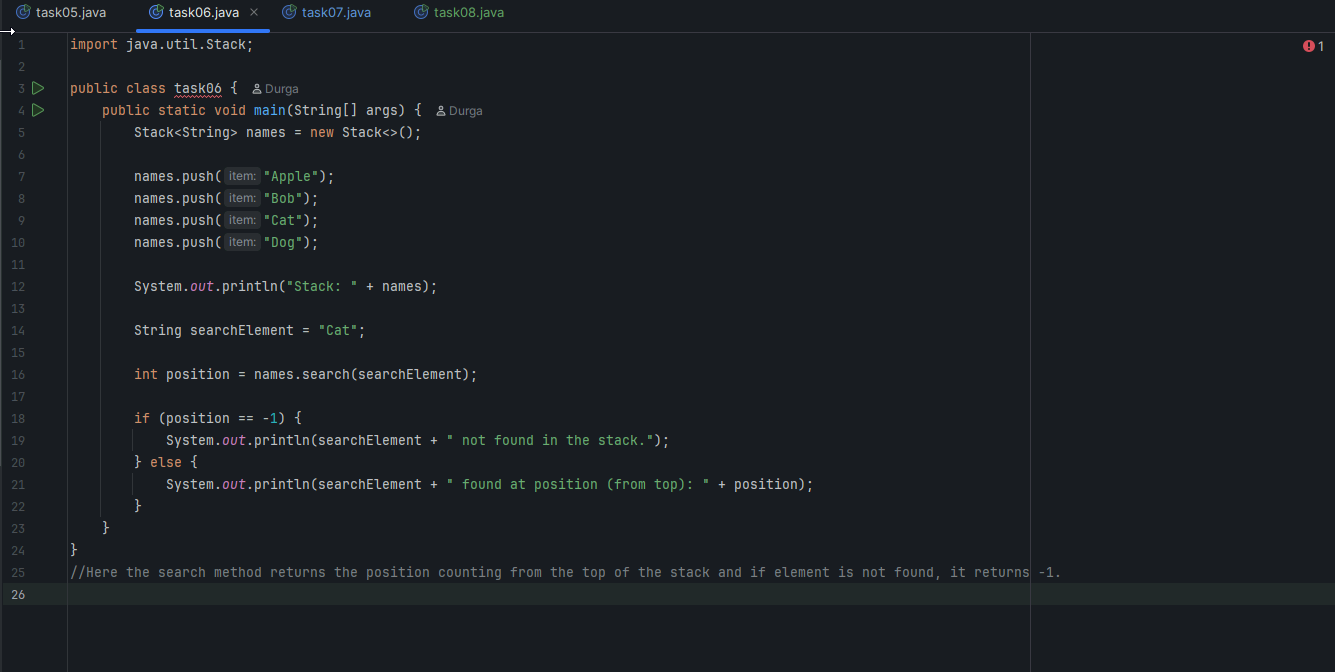


Task 4:

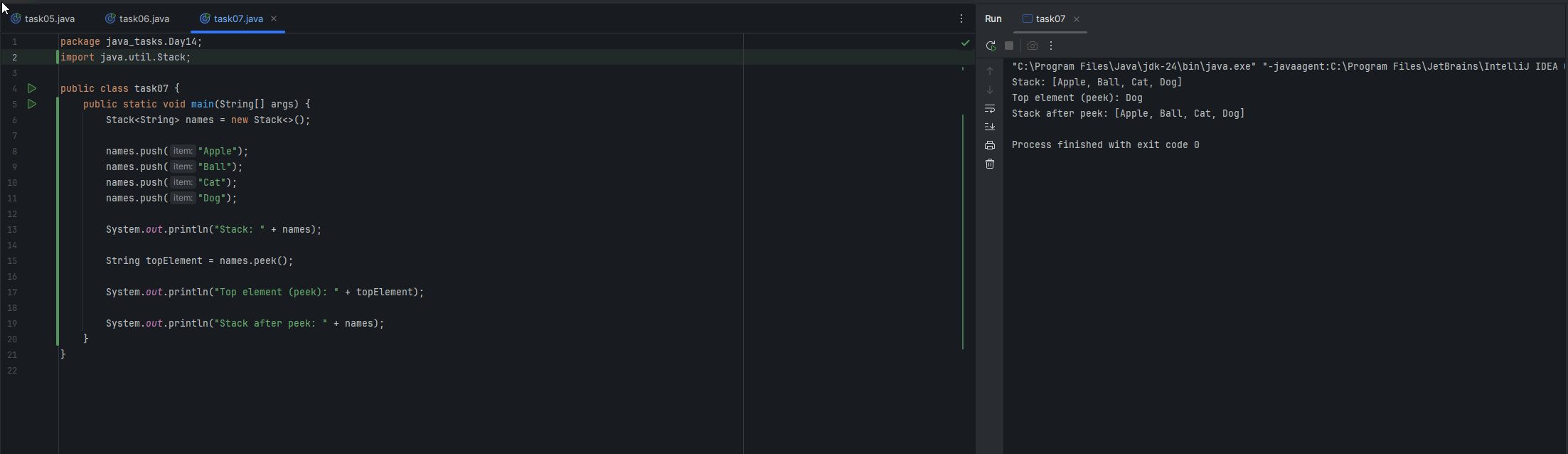
Task 5:



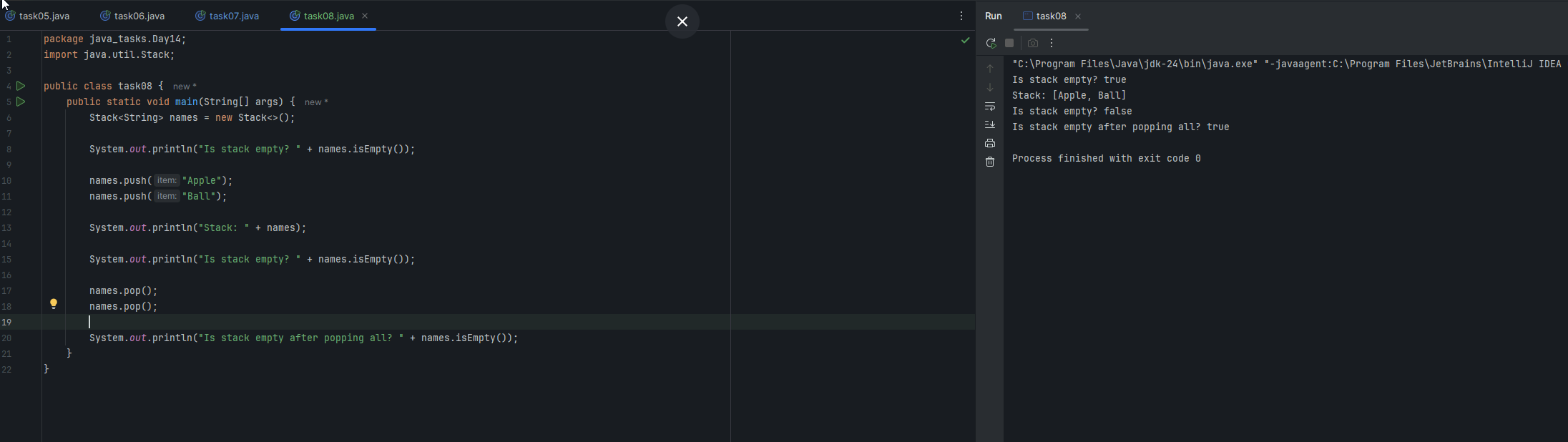
Task 6 :



Task7:



Task 8:



Task 9:

What are the methods of the stack class.. List them down.. With a one liner..

The main methods of the Java “Stack” class -which extends Vector,

Java `Stack` class methods:

-push(E item) - Adds (pushes) an item onto the top of the stack.

-pop() - Removes and returns the item at the top of the stack.

-peek() - Returns (but does not remove) the item at the top of the stack.

- empty() - Returns (true) if the stack is empty, otherwise (false). (Equivalent to `isEmpty()`, inherited from Vector)

-search(Object o) — Returns the 1-based position of an object from the top of the stack, or -1 if not found.

- isEmpty() — Returns true if the stack contains no elements, otherwise (false).

- size() — Returns the number of elements in the stack.

Task 10:

