According to Lysecky, Vahid, Lysecky, and Givargis, the stack is an abstract data type (ADT) that items are only added and removed from one end, the top (2015, section 2.12). To give a visualization of this, say you have three flavors of PEZ candy that you are going to load into the dispenser, strawberry, cherry, and orange. To load the candy, you push it into the dispenser one at a time. Let us say that this dispenser holds ten candies, and you push in two strawberry, three cherries, and two orange, in that order. To get to the strawberry, you will first have to pop out the two orange, then the three cherries to get to the first strawberry. Now, lets put this into programming terms, but not code at this time. You have a container that can hold ten items, an array with ten elements.

pezArray[10]

You load or push, candy one to the top of the stack.

Strawberry

In turn, you push each of the other candies in, until they are all loaded.

Orange

Orange

Cherry

Cherry

Cherry

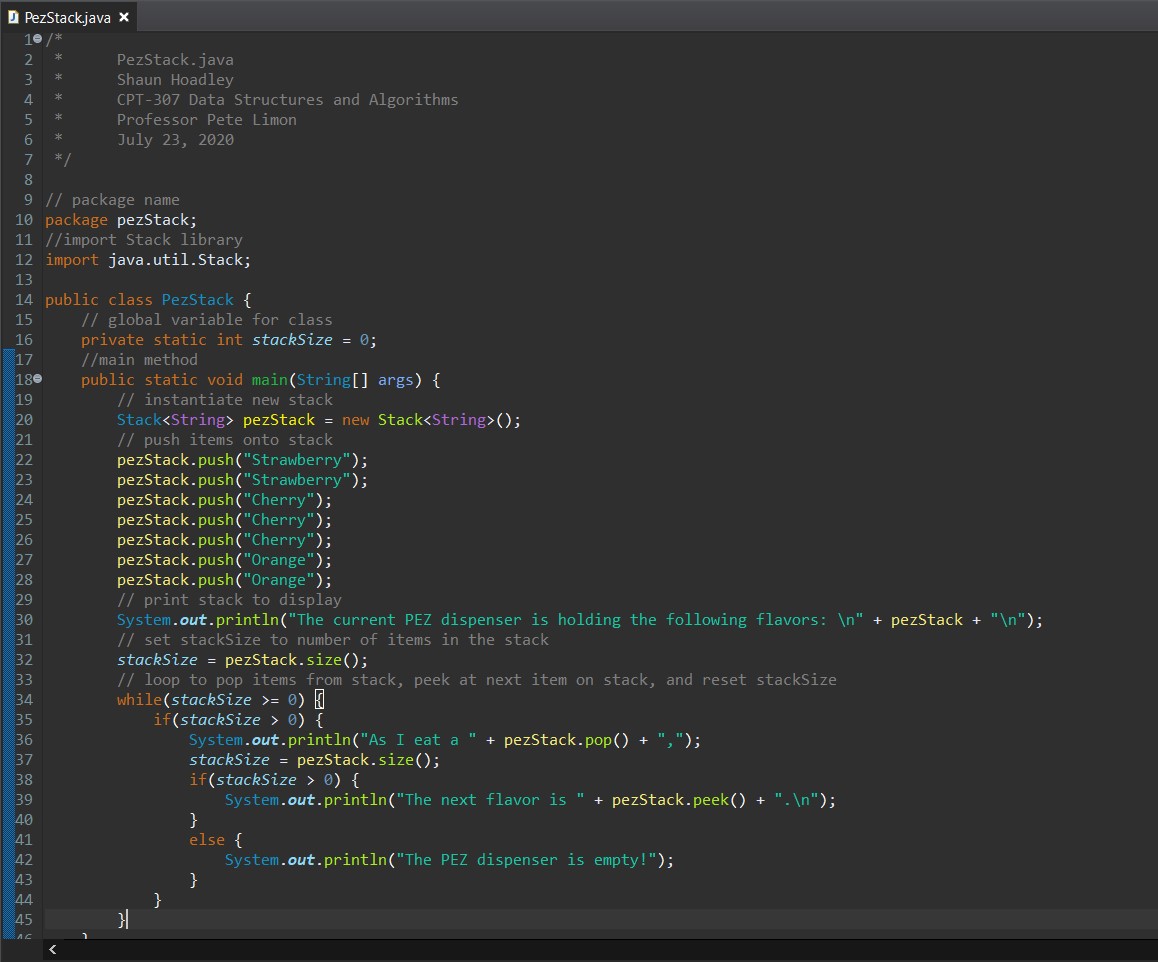
Strawberry

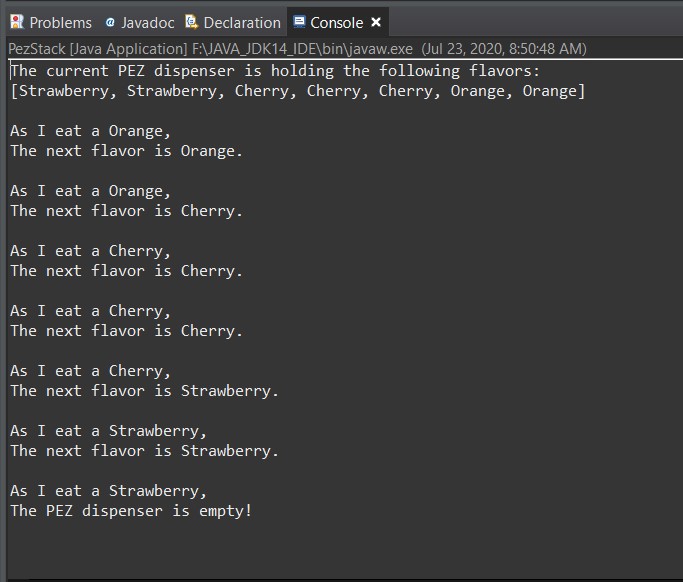
Strawberry

If you then print the array to the display you get the following:

Strawberry, Strawberry, Cherry, Cherry, Cherry, Orange, Orange

Lysecky indicates that “Pop and Peek operations should not be applied to an empty stack; the resulting behavior may be undefined” (2015, section 2.10). When using the Stack Class from the java.utils library, attempting pop or peek operations on an empty stack will throw an exception error. To prevent the exception from occurring, it is necessary to check for an empty stack prior to using those operations. In my little PEZ dispenser program, I populate the dispenser by pushing candy (items) onto the stack. It then loops through to pop each candy out and show what the next flavor is to be eaten, until the dispenser (stack) is empty.





**References**

Lysecky, R., Vahid, F., Lysecky, S., & Givargis, T. (2015). [Data structures essentials](https://ashford.instructure.com/courses/68707/modules/items/3478669). Retrieved from https://zybooks.zyante.com/#/zybook/ DataStructuresEssentialsR25/chapter/1/section/3