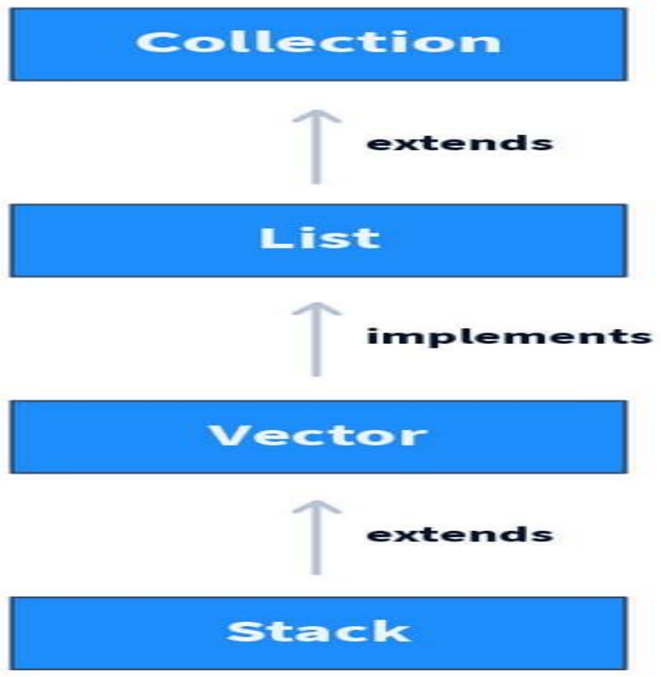
DATA STRUCTURE

Understanding Java’s Stack Class

The “Stack” class

- The **Stack** class is **available** in the package **java.util** from **JDK 1.0** .
- It **extends** the **Vector** class .



The “Stack” class

- It has **insertion order preserved** but it **used** to support **LIFO behaviour**

- **Duplicates** are **allowed.**

Constructor Of Stack

The **Stack** class **has only one constructor** which has the **following prototype**:

1. **public Stack( )**
   - This **constructor** builds an **empty Stack**

Stack specific methods

1 **public Object push(Object) : Pushes** an **item** onto the **top** of this **Stack** and **returns it**.

2 **public Object pop(): Removes** the **object** at the **top** of this **Stack** and **returns that object**. Throws an **EmptyStackException** if **Stack** is empty

3 **public Object peek(): Returns** the **object** at the **top of this stack without removing it** from the **Stack**. Throws an **EmptyStackException** if **Stack** is empty

Stack specific methods

4 **public boolean empty() : Checks** whether the **Stack** is **empty** or **not.**

5 **public int search(Object): Searches** for **element** in the **Stack** and returns its **position** in the **Stack** assuming that **TOP ELEMENT** of the **Stack** is at position 1. If element is **not found** then **-1** is returned.

Example:

```
Stack<Integer> numStack=new Stack<Integer>();
numStack.push(10);
numStack.push(20);
numStack.push(30);
Iterator <Integer> en=numStack.iterator();
while(en.hasNext())
{
    Integer obj=en.next();
    System.out.println(obj);
}
```

Output:

```
10
20
30
```

Example:

```
Stack<Integer> numStack =new Stack<Integer>();
numStack.push(10);
numStack.push(20);
numStack.push(30);
while(!numStack.empty())
{
  Integer obj= numStack.pop();
  System.out.println(obj);

}
```

Output:

```
10
20
30
```

Example:

```
Stack<Integer> numStack =new Stack<Integer>();
System.out.println("stack: " + numStack);
numStack.push(10);
numStack.push(20);
numStack.push(30);
System.out.println(“stack:”+ numStack);
System.out.println(“Top element:”+ numStack.peek());
System.out.println(“Popped ele:”+ numStack.pop());
System.out.println("stack: " + numStack);
```

Output:

```
stack: [ ]
stack:[10, 20, 30]
Top element:30
Popped ele:30
stack: [10, 20]
```

Example:

```
Stack<Integer> numStack =new Stack<Integer>();
System.out.println("stack: " + numStack);
numStack.push(10);
numStack.push(20);
numStack.push(30);
System.out.println("Offset of 10:"+ numStack.search(10));
System.out.println("Offset of 30:"+ numStack.search(30));
System.out.println("Offset of 40:"+ numStack.search(40));
```

Output:

```
stack: [ ]
Offset of 10:3
Offset of 30:1
Offset of 40:-1
```