**TDD**

**Functional Testing**  
Unit – Testing a single block of code  
Integration – Testing to see who blocks of code work with each other  
Smoke – Tests key functionality and acts as a precursor to performance tests

**Non-functional Testing**Scalability – Test to see how machine will perform in the future  
Performance – Mapping out performance characteristics under different loads

**Maintenance**Regression – Test features of previous sprints before testing new sprint

**7 Principles**Early testing is preferred (Shift Left)  
Testing is context dependent  
Exhaustive testing is impossible  
Absence of error fallacy (can’t be 100% sure of error free)  
Defect clustering (80/20)  
Pesticide paradox (Tests get used to/resistant to the code) – Must consistently write new test cases  
Testing shows presence of defects

**Streams**Stream = a flow of data  
A stream can do anything a for-loop can do:  
 – Filter  
 – Map  
 – Collect  
 – Reduce  
 – Sorted

Example: Sorting numbers  
List<Integer> myList = new ArrayList<Integer>();  
myList = {23, 7, 47, 11, 5};  
myList.stream().sorted().forEach(System.out::println);

Example: Removing odd numbers  
List<Integer> myList = new ArrayList<Integer>();  
myList = {4, 11, 21, 8, 16, 101};  
myList = myList.stream().filter(element -> element%2==0).collect(Collectors.toList());