

Practice 5

1. Two Sum Problem

Brute Force approach($O(n^2)$)

```
int n= nums.length;
for(int i=0; i<n; i++){
    for(int j=i+1; j<n; j++){
        if(nums[i]+nums[j] == target){
            System.out.println(i+" " +j);
        }
    }
}
```

i use nested loop to check all possible pair

Optimized Approach (HashMap - $O(n)$)

```
Map<Integer,Integer> map = new HashMap<>();
```

```
for(int i=0; i<nums.length; i++){
    int complement= target-nums[i];

    if(map.containsKey(complement)){
        return new int[]{ map.get(complement),i };
    }
    map.put(nums[i], i);
}
return new int[]{};
```

I use HashMap to store visited elements and reduce time complexity to $O(n)$

contains Duplicate

Optimized Approach (Hashset($O(n)$))

```
Set<Integer> set= new HashSet<>();
```

```
for(int num: nums){
    if(!set.add(num)){
        return true;
    }
}
return false;
```

HashSet does not allow duplicates so insertion failure confirms duplication,
so return value false and true

Java Core

//Count words in String

```
String str="Java is a powerful Language";
String[] words = str.split(" ");
```

```
System.out.println("Word count " + words.length());
```

```
// I split the string using space delimiter
```

SQL

Delete Employee with MINIMUM Salary

```
Delete from employee
where salary=(select MIN(salary) from employee);
```

i Use subquery to delete records with minimum salary

SpringBoot- GET by IntToDoubleFunction

```
public ResponseEntity<Employee> getById(@PathVariable int id){
    return repo.findById(id)
        .map(ResponseEntity::ok)
        .orElse(ResponseEntity.notfound().build());
}
```

//I use responseEntity to handle HTTP status codes properly.
