**Day 2**

[Jewels and Stones](https://leetcode.com/problems/jewels-and-stones/)

class Solution {

public int numJewelsInStones(String jewels, String stones) {

int count = 0;

StringBuilder str = new StringBuilder();

str.append(jewels);

for (char c : stones.toCharArray())

{

if ( str.toString().contains(String.valueOf(c))) count++;

}

return count;

}

}

// *time O (1), space O(n)*

 Kids With the Greatest Number of Candies

class Solution {

public List<Boolean> kidsWithCandies(int[] candies, int extraCandies) {

ArrayList<Boolean> bool = new ArrayList<>();

int max = candies[0];

for (int candy : candies) {

max = Math.max(max, candy);

}

for (int i = 0; i < candies.length; i++) {

if (candies[i] + extraCandies >= max) {

bool.add(true);

} else {

bool.add(false);

}

}

return bool;

}

}

*//time O(n), space O(n)*

[Design Parking System](https://leetcode.com/problems/design-parking-system/)

class ParkingSystem {

private int bigSlots;

private int mediumSlots;

private int smallSlots;

public ParkingSystem(int big, int medium, int small) {

bigSlots = big;

mediumSlots = medium;

smallSlots = small;

}

public boolean addCar(int carType) {

if(carType == 1)

{

if(bigSlots > 0)

{

bigSlots --;

return true;

}

}

else if(carType == 2)

{

if(mediumSlots > 0)

{

mediumSlots --;

return true;

}

}

else if(carType == 3)

{

if(smallSlots > 0)

{

smallSlots --;

return true;

}

}

return false;

}

}

[Richest Customer Wealth](https://leetcode.com/problems/richest-customer-wealth/)

class Solution {

public int maximumWealth(int[][] accounts) {

int largest = accounts[0][0];

for(int customer[] : accounts){

int currentSum = 0;

for(int bank : customer) currentSum += bank;

largest = Math.max(largest, currentSum);

}

return largest;

//*time O(n x m), Space O(1)*

}

}