ASSIGNMENT 2

1.

```
def arrayPairSum(nums):
    nums.sort()
    return sum(nums[::2])
```

2.

```
def distributeCandies(candyType):
    unique_candies = len(set(candyType))
    max_candies = len(candyType) // 2
    return min(unique_candies, max_candies)
```

3.

```
def findLHS(nums):
    num_count = {}
    for num in nums:
        num_count[num] = num_count.get(num, 0) + 1
    longest_subsequence = 0
    for num in nums:
        if num + 1 in num_count:
            longest_subsequence = max(longest_subsequence, num_count[num] + num_count[num + 1])
    return longest_subsequence
```

4.

```
def canPlaceFlowers(flowerbed, n):
    flowerbed.append(0)
    flowerbed.insert(0, 0)
    i = 1
    count = 0
    while i < len(flowerbed) - 1:
        if flowerbed[i-1] == 0 and flowerbed[i] == 0 and flowerbed[i+1] == 0:
            flowerbed[i] = 1
            count += 1
        i += 1
```

```
    return count >= n
```

5.

```
def maximumProduct(nums):
    nums.sort()
    return max(nums[-1] * nums[-2] * nums[-3], nums[0] * nums[1] * nums[-1])
```

6.

```
def search(nums, target):
    left = 0
    right = len(nums) - 1
    while left <= right:
        mid = (left + right) // 2
        if nums[mid] == target:
            return mid
        elif nums[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1
```

7.

```
def isMonotonic(nums):
    increasing = decreasing = True
    for i in range(1, len(nums)):
        if nums[i] < nums[i-1]:
            increasing = False
        if nums[i] > nums[i-1]:
            decreasing = False
    return increasing or decreasing
```

8.

```
def minDifference(nums):
    if len(nums) <= 4:
        return 0
```

```
nums.sort()
return min(nums[-4] - nums[0], nums[-3] - nums[1], nums[-2] - nums[2], nums[-1] - nums[3])
```