Question1: Say you have an array for which the ith element is the price of a given stock on day i. If you were only permitted to complete at most one transaction (i.e., buy one and sell one share of the stock), design an algorithm to find the maximum profit.

Answer

def max\_profit(prices):

if not prices:

return 0

min\_price = prices[0]

max\_profit = 0

for price in prices:

if price < min\_price:

min\_price = price

else:

max\_profit = max(max\_profit, price - min\_price)

return max\_profit

Question:2. Question: Write a function that finds the most repeated character in a string.

ANswer:

def most\_repeated(s):

char\_count = {}

for char in s:

if char in char\_count:

char\_count[char] += 1

else:

char\_count[char] = 1

max\_char = max(char\_count, key=char\_count.get)

return max\_char

string = "aabbbcdddde"

print(most\_repeated(string)) # Output: 'd'

Question3: Write a function that returns the longest consecutive subsequence in a list of numbers.

ANSWER:

def longest\_consecutive\_subsequence(nums):

if not nums:

return []

nums = sorted(set(nums))

longest\_streak = []

current\_streak = [nums[0]]

for i in range(1, len(nums)):

if nums[i] - nums[i - 1] == 1:

current\_streak.append(nums[i])

else:

if len(current\_streak) > len(longest\_streak):

longest\_streak = current\_streak

current\_streak = [nums[i]]

return longest\_streak if len(longest\_streak) > len(current\_streak) else current\_streak

numbers = [1, 2, 3, 5, 6, 7, 8, 10]

print(longest\_consecutive\_subsequence(numbers)) # Output: [5, 6, 7, 8]

Question4: Write a function to find the longest common prefix of a list of strings.

def longest\_common\_prefix(strings):

if not strings:

return ""

prefix = strings[0]

for s in strings[1:]:

while not s.startswith(prefix):

prefix = prefix[:-1]

return prefix

strings = ["flower", "flow", "flight"]

print(longest\_common\_prefix(strings)) # Output: "fl"

Question5: Write a function to find the two numbers in a list that sum up to a specific target.

def two\_sum(nums, target):

num\_dict = {}

for i, num in enumerate(nums):

complement = target - num

if complement in num\_dict:

return [num\_dict[complement], i]

num\_dict[num] = i

return None

numbers = [2, 7, 11, 15]

target\_value = 9

print(two\_sum(numbers, target\_value)) # Output: [0, 1]

Question6: Write a function that reverses a string, but maintains the position of all non-alphabetic characters.

def reverse\_alphabet\_only(s):

s = list(s)

i, j = 0, len(s) - 1

while i < j:

if not s[i].isalpha():

i += 1

elif not s[j].isalpha():

j -= 1

else:

s[i], s[j] = s[j], s[i]

i += 1

j -= 1

return ''.join(s)

string = "ab@cd#ef$gh"

print(reverse\_alphabet\_only(string)) # Output: "hg@fe#dc$ba"