Assignment-I
Python Programming (ITO- 804)

Submitted By

Name of Student: Ayush Sharma
Roll Number: 191203080
Branch / Semester: CSE 8

Submitted To

Saurabh Sharma
Assistant Professor (CSE Department)

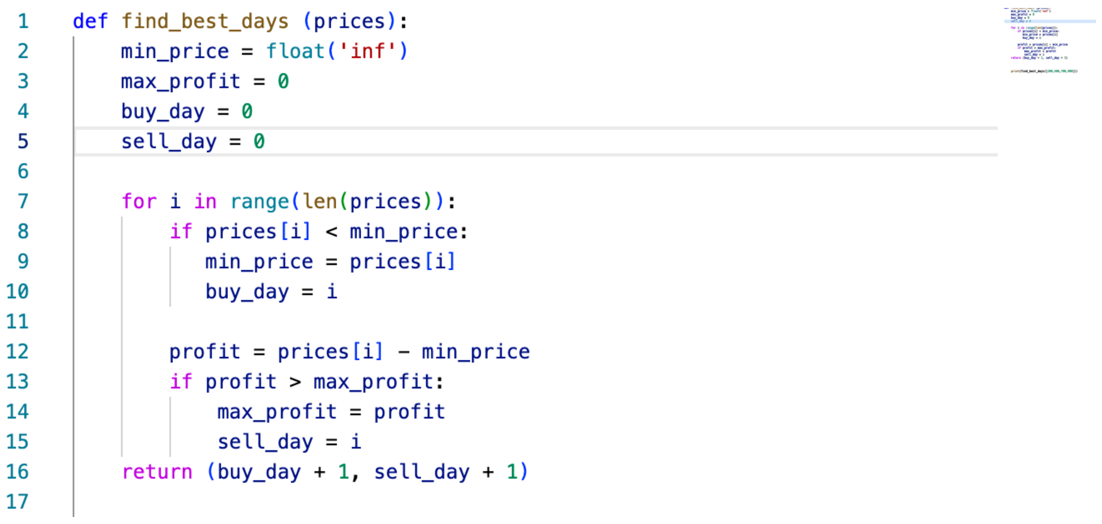
Department of Computer Science and Engineering
Model Institute of Engineering and Technology (Autonomous)
Jammu, India

Index

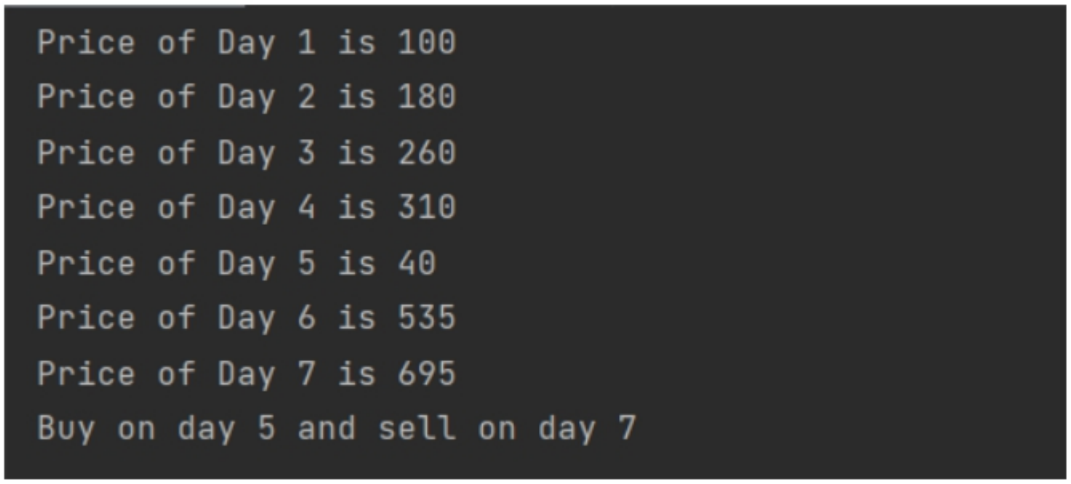
S.no	Question	Marks
1	Write a Python program that takes a list of daily stock prices as input, and returns the best days to buy and sell stocks in order to maximise profit. The list contains the stock prices for each day, starting from the first day. For example, the list (100, 180, 260, 310, 40, 535, 695) represents the stock prices for 7 days, where the price on the first day is 100, the second day is 180, and so on. The program should find the best days to buy and sell stocks such that the profit obtained is maximum. For instance, in the given list of stock prices, the best days to buy and sell stocks would be: Buy stock on the 1st day (price=100) Sell stock on the 4th day (price=310) Buy stock on the 5th day (price=40) Sell stock on the 7th day (price=695) The program should output these buy and sell days as a tuple or list of integers.	2.5
2	You are given a list of book titles and their corresponding publication years. Your task is to find the earliest year in which a trilogy of books was published. A trilogy is defined as a series of three books published in consecutive years. For example, consider the following list of book titles and publication years: titles = ['The Hunger Games', 'Catching Fire', 'Mockingjay', 'The Lord of the Rings', 'The Two Towers', 'The Return of the King', 'Divergent', 'Insurgent', 'Allegiant'] years = [2008, 2009, 2010, 1954, 1955, 1956, 2011, 2012, 2013] The earliest year in which a trilogy was published is 1954. Write a Python function earliest_trilogy_year(titles: List[str], years: List[int]) -> Optional[int] that takes two lists as input: titles containing the titles of the books, and years containing their corresponding publication years. The function should return the earliest year in which a trilogy of books was published, or None if no such trilogy exists. Examples: titles = ['Book1', 'Book2', 'Book3', 'Book4', 'Book5', 'Book6'] years = [2019, 2021, 2012, 2013, 2016, 2017] print(earliest_trilogy_year(titles, years)) The earliest year in which a trilogy was published is : None A trilogy is defined as a series of three books published in consecutive years. Note: · You can assume that the input lists are non-empty and contain an equal number of elements. · If multiple trilogies exist with the same earliest year, return that year.	2.5
3	Write a Python program that reads in a CSV file of stock prices (e.g. ticker symbol, date, price), and then uses dictionaries and lists to calculate the highest and lowest prices for each stock from following table.	2.5
4	A) Write a Python program to remove duplicates from a list of lists. Sample list: [[10, 20], [40], [30, 56, 25], [10, 20], [33], [40]] B) Write a Python program which takes a list and returns a list with the elements "shifted left by one position" so [1, 2, 3] yields [2, 3, 1]. Example: [1, 2, 3] → [2, 3, 1] [11, 12, 13] → [12, 13, 11] C) Iterate a given list and count the occurrence of each element and create a dictionary to show the count of each element. Original list [11, 45, 8, 11, 23, 45, 23, 45, 89, 11, 89] And expected output is: Printing count of each item {11: 3, 45: 3, 8: 1, 23: 2, 89: 2}	2.5

1. Write a Python program that takes a list of daily stock prices as input, and returns the best days to buy and sell stocks in order to maximize profit. The list contains the stock prices for each day, starting from the first day. For example, the list (100, 180, 260, 310, 40, 535, 695) represents the stock prices for 7 days, where the price on the first day is 100, the second day is 180, and so on. The program should find the best days to buy and sell stocks such that the profit obtained is maximum. For instance, in the given list of stock prices, the best days to buy and sell stocks would be: Buy stock on the 1st day (price=100) Sell stock on the 4th day (price=310) Buy stock on the 5th day (price=40) Sell stock on the 7th day (price=695) The program should output these buy and sell days as a tuple or list of integers.

Program + output



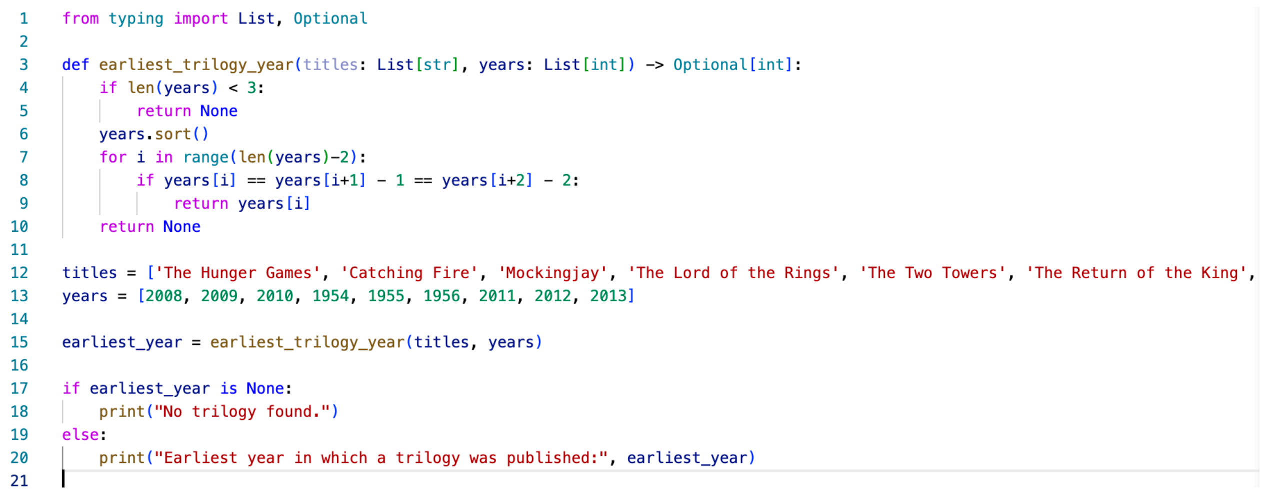
```python
def find_best_days (prices):
    min_price = float('inf')
    max_profit = 0
    buy_day = 0
    sell_day = 0

    for i in range(len(prices)):
        if prices[i] < min_price:
            min_price = prices[i]
            buy_day = i

        profit = prices[i] - min_price
        if profit > max_profit:
            max_profit = profit
            sell_day = i
    return (buy_day + 1, sell_day + 1)

```



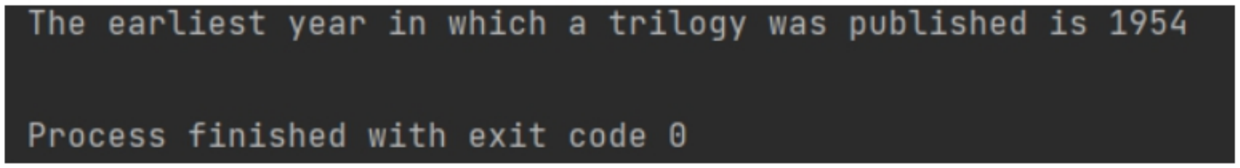
```
Price of Day 1 is 100
Price of Day 2 is 180
Price of Day 3 is 260
Price of Day 4 is 310
Price of Day 5 is 40
Price of Day 6 is 535
Price of Day 7 is 695
Buy on day 5 and sell on day 7
```

2. Write a Python function earliest_trilogy_year(titles: List[str], years: List[int]) -> Optional[int] that takes two lists as input: titles containing the titles of the books, and years containing their corresponding publication years. The function should return the earliest year in which a trilogy of books was published, or None if no such trilogy exists. Examples: titles = ['Book1', 'Book2', 'Book3', 'Book4', 'Book5', 'Book6'] years = [2019, 2021, 2012, 2013, 2016, 2017] print(earliest_trilogy_year(titles, years))

Program + Output

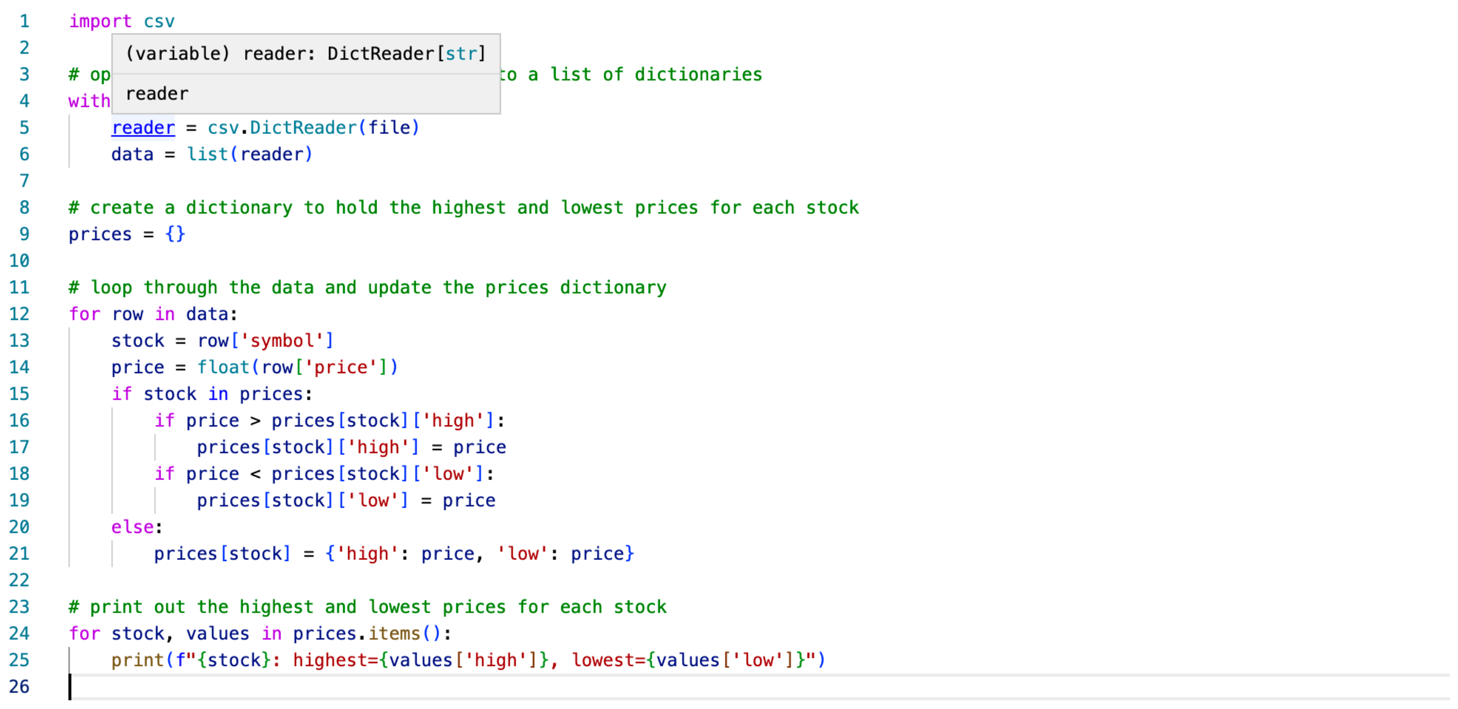


```python
from typing import List, Optional

def earliest_trilogy_year(titles: List[str], years: List[int]) -> Optional[int]:
    if len(years) < 3:
        return None
    years.sort()
    for i in range(len(years)-2):
        if years[i] == years[i+1] - 1 == years[i+2] - 2:
            return years[i]
    return None

titles = ['The Hunger Games', 'Catching Fire', 'Mockingjay', 'The Lord of the Rings', 'The Two Towers', 'The Return of the King',
years = [2008, 2009, 2010, 1954, 1955, 1956, 2011, 2012, 2013]

earliest_year = earliest_trilogy_year(titles, years)

if earliest_year is None:
    print("No trilogy found.")
else:
    print("Earliest year in which a trilogy was published:", earliest_year)

```

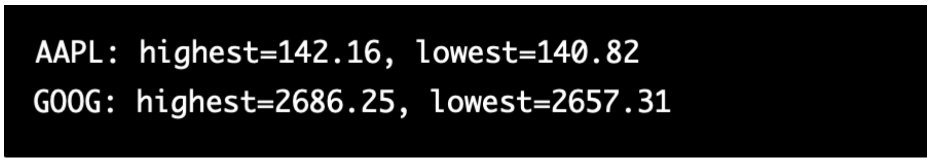


```
The earliest year in which a trilogy was published is 1954

Process finished with exit code 0
```

3. Write a Python program that reads in a CSV file of stock prices (e.g. ticker symbol, date, price), and then uses dictionaries and lists to calculate the highest and lowest prices for each stock from following table.



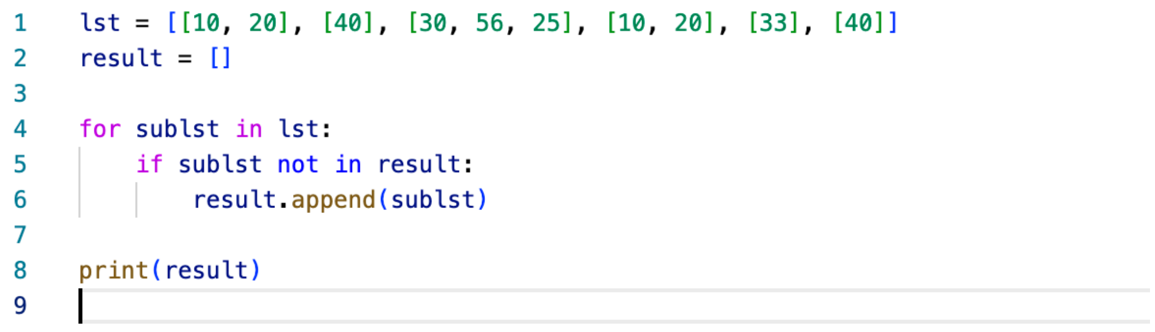
```python
import csv

# op                                       to a list of dictionaries
with
    reader = csv.DictReader(file)
    data = list(reader)

# create a dictionary to hold the highest and lowest prices for each stock
prices = {}

# loop through the data and update the prices dictionary
for row in data:
    stock = row['symbol']
    price = float(row['price'])
    if stock in prices:
        if price > prices[stock]['high']:
            prices[stock]['high'] = price
        if price < prices[stock]['low']:
            prices[stock]['low'] = price
    else:
        prices[stock] = {'high': price, 'low': price}

# print out the highest and lowest prices for each stock
for stock, values in prices.items():
    print(f"{stock}: highest={values['high']}, lowest={values['low']}")

```



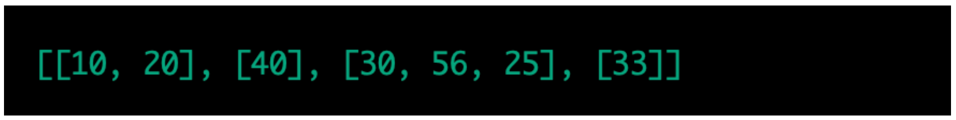
```
AAPL: highest=142.16, lowest=140.82
GOOG: highest=2686.25, lowest=2657.31
```

4. A)) Write a Python program to remove duplicates from a list of lists. Sample list: [[10, 20], [40], [30, 56, 25], [10, 20], [33], [40]]

Program + Output



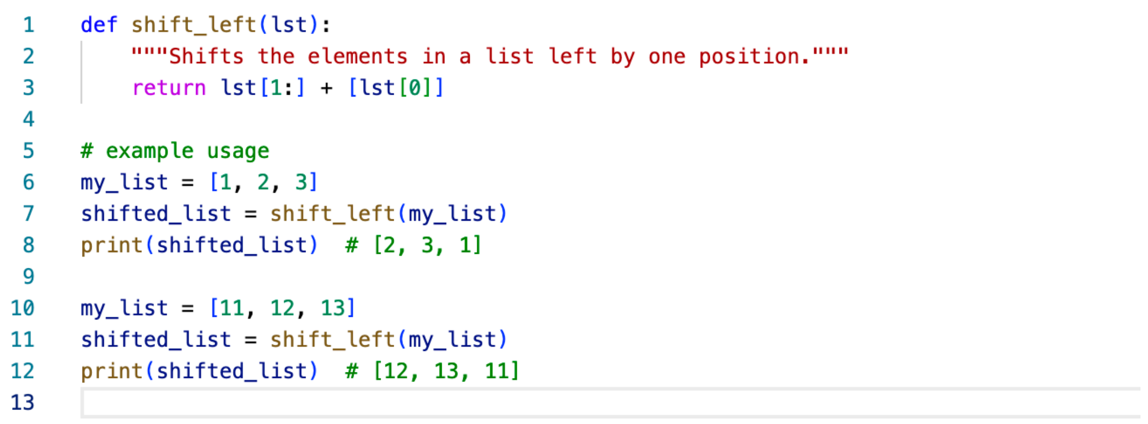
```python
lst = [[10, 20], [40], [30, 56, 25], [10, 20], [33], [40]]
result = []

for sublst in lst:
    if sublst not in result:
        result.append(sublst)

print(result)

```



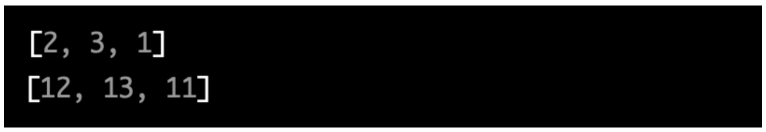
```
[[10, 20], [40], [30, 56, 25], [33]]
```

B) Write a Python program which takes a list and returns a list with the elements "shifted left by one position" so [1, 2, 3] yields [2, 3, 1]. Example: [1, 2, 3] → [2, 3, 1] [11, 12, 13] → [12, 13, 11]

Program + Output



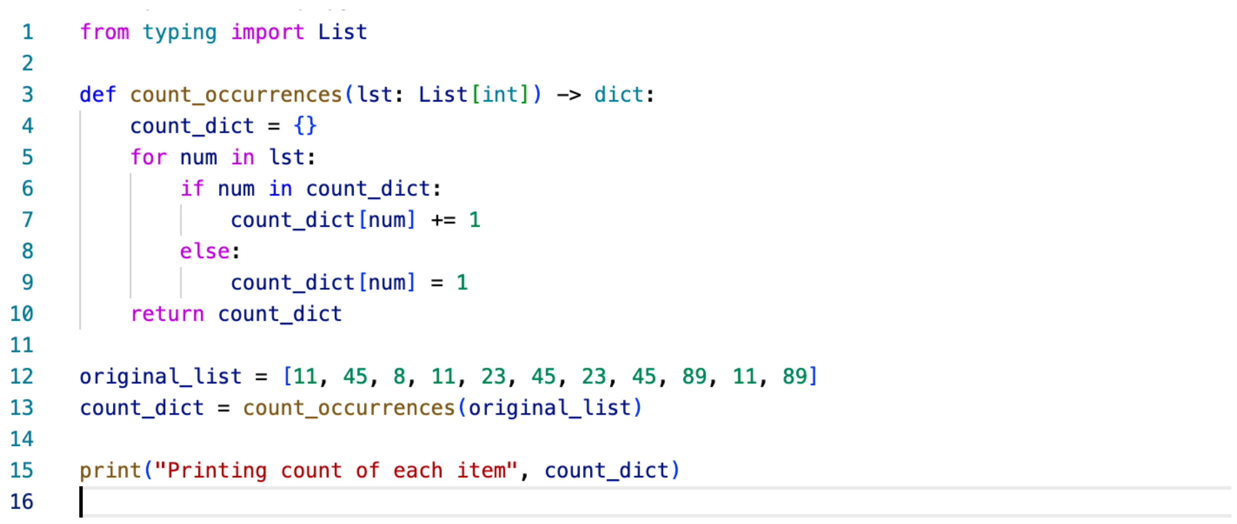
```python
def shift_left(lst):
    """Shifts the elements in a list left by one position."""
    return lst[1:] + [lst[0]]

# example usage
my_list = [1, 2, 3]
shifted_list = shift_left(my_list)
print(shifted_list)  # [2, 3, 1]

my_list = [11, 12, 13]
shifted_list = shift_left(my_list)
print(shifted_list)  # [12, 13, 11]

```



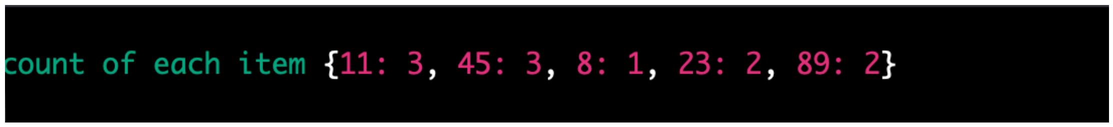
```
[2, 3, 1]
[12, 13, 11]
```

c) Iterate a given list and count the occurrence of each element and create a dictionary to show the count of each element. Original list [11, 45, 8, 11, 23, 45, 23, 45, 89, 11, 89] And expected output is: Printing count of each item {11: 3, 45: 3, 8: 1, 23: 2, 89: 2}

Program + Output



```python
from typing import List

def count_occurrences(lst: List[int]) -> dict:
    count_dict = {}
    for num in lst:
        if num in count_dict:
            count_dict[num] += 1
        else:
            count_dict[num] = 1
    return count_dict

original_list = [11, 45, 8, 11, 23, 45, 23, 45, 89, 11, 89]
count_dict = count_occurrences(original_list)

print("Printing count of each item", count_dict)

```



```
count of each item {11: 3, 45: 3, 8: 1, 23: 2, 89: 2}
```