You can view this report online at : https://www.hackerrank.com/x/tests/1257195/candidates/33952646/report

Full Name:	Rohan Raj
Email:	rr07656@st.habib.edu.pk
Test Name:	**CS 101 - Lab# 12 - Fall 2021 [Dictionary and Tuples]**
Taken On:	20 Dec 2021 15:23:54 PKT
Time Taken:	8547 min 48 sec/ 13890 min
Work Experience:	< 1 years
Invited by:	Aisha
Skills Score:	
Tags Score:	CS101 100/100 Lists 100/100 NestedLists 100/100 Tuples 100/100

100%

490/490

scored in **CS 101 - Lab# 12 - Fall 2021 [Dictionary and Tuples]** in 8547 min 48 sec on 20 Dec 2021 15:23:54 PKT

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Don't pick a lock, pick a key > **Coding**	53 min 37 sec	40/ 40	✓
Q2	Say my date, say my date > **Coding**	15 min 22 sec	80/ 80	✓
Q3	You get a birthday dictionary, and you, and you, and you! > **Coding**	2 hour 15 min	80/ 80	✓
Q4	Merge by Key > **Coding**	40 min 36 sec	20/ 20	✓
Q5	Merge by Value > **Coding**	27 min 43 sec	20/ 20	✓
Q6	Count Words > **Coding**	3 hour 5 min 12 sec	40/ 40	✓
Q7	Last name first > **Coding**	2 hour 15 min 2 sec	100/ 100	✓
Q8	Loan Repayment Strategy > **Coding**	2 hour 2 min 1 sec	10/ 10	✓
Q9	Get Positions > **Coding**	42 min 45 sec	100/ 100	✓

QUESTION 1

✓

Correct Answer

Don't pick a lock, pick a key > Coding

QUESTION DESCRIPTION

Score 40

Challenge

Write a function called `pick` that accepts a key `k` and a list of dictionaries `t` as a parameter, and *returns* a list of values corresponding to the key `k` in each of the dictionaries in list `t`.

Note

Order of values in the returned list is preserved. Key `k` can be of any type acceptable as a key in a `dict`.

Sample

```
>>> pick('year', [{'year': 1995, 'month': 8, 'day': 3}, {'year': 1994,
'month': 7, 'day': 15}, {'year': 1997, 'month': 3, 'day': 17}, {'year': 1995,
'month': 10, 'day': 17}, {'year': 1999, 'month': 3, 'day': 7}, {'year': 1995,
'month': 6, 'day': 4}, {'year': 1994, 'month': 4, 'day': 29}, {'year': 1999,
'month': 5, 'day': 18}, {'year': 1994, 'month': 7, 'day': 3}, {'year': 1994,
'month': 8, 'day': 7}, {'year': 1999, 'month': 4, 'day': 5}, {'year': 1998,
'month': 9, 'day': 30}])
[1995, 1994, 1997, 1995, 1999, 1995, 1994, 1999, 1994, 1994, 1999, 1998]
>>> pick('day', [{'year': 1995, 'month': 8, 'day': 3}, {'year': 1994,
'month': 7, 'day': 15}, {'year': 1997, 'month': 3, 'day': 17}, {'year': 1995,
'month': 10, 'day': 17}, {'year': 1999, 'month': 3, 'day': 7}, {'year': 1995,
'month': 6, 'day': 4}, {'year': 1994, 'month': 4, 'day': 29}, {'year': 1999,
'month': 5, 'day': 18}, {'year': 1994, 'month': 7, 'day': 3}, {'year': 1994,
'month': 8, 'day': 7}, {'year': 1999, 'month': 4, 'day': 5}, {'year': 1998,
'month': 9, 'day': 30}])
[3, 15, 17, 17, 7, 4, 29, 18, 3, 7, 5, 30]
```

Input/Output

Input consists of a key `k` on the first line, and a list literal on the second line, which HackerRank will read in as `t`. HackerRank will pass the two arguments to your function, and then output the result.

Constraints

- `t` will contain at least one date.

INTERVIEWER GUIDELINES

```
# Using a list comprehension:
def pick(k, t):
    return [d[k] for d in t]

# Using a for loop:
def pick(k, t):
    result = []
    for d in t:
        result.append(d[k])
    return result
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def pick(k, t):
    if len(t) > 0:
        l = []
        for d in t:
            l.append(d[k])
        return l
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0362 sec	8.01 KB
Testcase 1	Easy	Sample case	✓ Success	10	0.0283 sec	8.01 KB

Testcase 1	Easy	Sample case	✓ Success	10	0.0283 sec	8.01 KB
Testcase 2	Easy	Hidden case	✓ Success	10	0.041 sec	8.01 KB
Testcase 3	Easy	Hidden case	✓ Success	10	0.0641 sec	8 KB

No Comments

QUESTION 2

✓ Correct Answer

Score 80

Say my date, say my date › Coding

QUESTION DESCRIPTION

Background

In Python, a dictionary can be used to avoid a sequence of if-elif-else statements.

Challenge

Write a function called `print_dates_in_long_form` that accepts a list of date dictionaries `t` as a parameter, and prints dates in "*month dd*, *yyyy*" format, each date on a separate line. A single date dictionary object contains the keys `'year'`, `'month'`, and `'day'`, with associated numeric values.

Note

Dates should not be checked for validity. Dates should be printed in the same order as in the list. Your code must use the provided dictionary called `month_names` that contains a translation from the month number to the month name.

Sample

```
>>> print_dates_in_long_form([{'day': 12, 'month': 12, 'year': 1996}, {'day':
8, 'month': 12, 'year': 1995}, {'day': 30, 'month': 4, 'year': 1999}, {'day':
30, 'month': 7, 'year': 1998}])
December 12, 1996
December 8, 1995
April 30, 1999
July 30, 1998
```

Input/Output

Input consists of a list literal that HackerRank will read in as `t` and pass to your function.

Constraints

- `t` will contain at least one date.

INTERVIEWER GUIDELINES

```
def date_to_long_form(date):
    return month_names[date['month']] + ' ' + str(date['day']) + ', ' +
str(date['year'])

def print_dates_in_long_form(dates):
    for date in dates:
        print(date_to_long_form(date))
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def print_dates_in_long_form(t):
    if len(t) > 0:
        for d in t:
```

```
        print (month_names[d['month']], str(d['day']) + ",", d['year'])
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0385 sec	7.92 KB
Testcase 1	Easy	Sample case	✓ Success	10	0.0362 sec	7.97 KB
Testcase 2	Easy	Sample case	✓ Success	10	0.0327 sec	7.98 KB
Testcase 3	Easy	Sample case	✓ Success	10	0.0257 sec	7.88 KB
Testcase 4	Easy	Hidden case	✓ Success	10	0.0418 sec	7.91 KB
Testcase 5	Easy	Hidden case	✓ Success	10	0.0358 sec	7.98 KB
Testcase 6	Easy	Hidden case	✓ Success	10	0.0392 sec	8.02 KB
Testcase 7	Easy	Hidden case	✓ Success	10	0.0377 sec	7.81 KB

No Comments

QUESTION 3

✓ Correct Answer

Score 80

You get a birthday dictionary, and you, and you, and you! › Coding

QUESTION DESCRIPTION

Background

In Python, a dictionary is a mapping from keys to values. This mapping could be used to represent attributes of an object, no matter if it is real-world or abstract.

A calendar date (in particular, from the proleptic Gregorian calendar in the common era) can be represented by a dictionary containing three keys--namely the year, month, and day--and their associated numeric values.

Challenge

Write a function called `split_dates` that accepts a string `s` as a parameter, and *returns* a list of dictionaries that each represent a date given in `s`. Each date in `s` is written in *yyyy-mm-dd* format, and is separated from other dates by whitespace.

Note

Each dictionary object in the list will contain three keys, `'year'`, `'month'`, and `'day'`, and their associated numeric (`int` type) values. Dates should not be checked for validity. Keys in `dict` do not necessarily preserve a particular order, so your output may display keys in a different order than the sample interaction shown below.

Sample

```
>>> split_dates('1996-12-12 1995-12-08 1999-04-30 1998-07-30')
[{'year': 1996, 'month': 12, 'day': 12}, {'year': 1995, 'month': 12, 'day':
8}, {'year': 1999, 'month': 4, 'day': 30}, {'year': 1998, 'month': 7, 'day':
30}]
>>> split_dates('1995-08-03 1994-07-15 1997-03-17 1995-10-17 1999-03-07 1995-
06-04 1994-04-29 1999-05-18 1994-07-03 1994-08-07 1999-04-05 1998-09-30')
[{'year': 1995, 'month': 8, 'day': 3}, {'year': 1994, 'month': 7, 'day': 15},
{'year': 1997, 'month': 3, 'day': 17}, {'year': 1995, 'month': 10, 'day':
17}, {'year': 1999, 'month': 3, 'day': 7}, {'year': 1995, 'month': 6, 'day':
4}, {'year': 1994, 'month': 4, 'day': 29}, {'year': 1999, 'month': 5, 'day':
18}, {'year': 1994, 'month': 7, 'day': 3}, {'year': 1994, 'month': 8, 'day':
7}, {'year': 1999, 'month': 4, 'day': 5}, {'year': 1998, 'month': 9, 'day':
30}]
```

Input/Output

Input consists of `s` as whitespace-separated dates in *yyyy-mm-dd* format on a single line. HackerRank will read in `s` and pass it to your function, and then output the dictionary, with the keys (not the values) sorted in a natural order.

Constraints

- `s` will contain at least one date.

INTERVIEWER GUIDELINES

```
# One-liner using list comprehension and tuple assignment:
def split_dates(dates):
    return [{ 'year': int(year), 'month': int(month), 'day': int(day) }
for date in dates.split() for year, month, day in [date.split('-')]]

# Using two list comprehensions:
def split_dates(dates):
    dates = [date.split('-') for date in dates.split()]
    return [{ 'year': int(date[0]), 'month': int(date[1]), 'day':
int(date[2]) } for date in dates]

# Using a for loop:
def split_dates(dates):
    result = []
    for date in dates.split():
        date = date.split('-')
        result.append({ 'year': int(date[0]), 'month': int(date[1]),
'day': int(date[2]) })
    return result
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def split_dates(s):
    x = s.split()
    l = []
    if len(s) > 0:
        for i in x:
            di = {'year':0, 'month': 0, 'day':0}
            di['year'] = int(i[0:4])
            di['month'] = int(i[5:7])
            di['day'] = int(i[8:10])
            l.append(di)
        return (l)


```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0623 sec	8.74 KB
Testcase 1	Easy	Sample case	✓ Success	10	0.0549 sec	8.84 KB
Testcase 2	Easy	Sample case	✓ Success	10	0.0431 sec	8.75 KB
Testcase 3	Easy	Sample case	✓ Success	10	0.0425 sec	8.86 KB
Testcase 4	Easy	Hidden case	✓ Success	10	0.0398 sec	9.02 KB
Testcase 5	Easy	Hidden case	✓ Success	10	0.037 sec	8.79 KB
Testcase 6	Easy	Hidden case	✓ Success	10	0.0406 sec	8.86 KB
Testcase 7	Easy	Hidden case	✓ Success	10	0.0598 sec	8.96 KB

No Comments

QUESTION 4

Correct Answer

Score 20

Merge by Key › Coding

QUESTION DESCRIPTION

Problem

Write a function named `merge_key` that takes two dictionaries `d1` and `d2` as parameters and builds a dictionary that contains every key from `d1` and `d2` with the corresponding value. If a key appears in both `d1` and `d2`, the value in the merged dictionary is a list containing the value from `d1` and from `d2`. The function *returns* a sorted list of the (key, value) pairs in the merged dictionary.

Hint: Use Python Dictionary get method.

Sample

```
>>> d1 = {i:chr(96+i) for i in range(1,11)}
>>> d2 = {i:chr(64+i) for i in range(1,11)}
>>> d1
{1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f', 7: 'g', 8: 'h', 9: 'i',
10: 'j'}
>>> d2
{1: 'A', 2: 'B', 3: 'C', 4: 'D', 5: 'E', 6: 'F', 7: 'G', 8: 'H', 9: 'I',
10: 'J'}
>>> merge_key(d1,d2)
{1: ['a', 'A'], 2: ['b', 'B'], 3: ['c', 'C'], 4: ['d', 'D'], 5: ['e',
'E'], 6: ['f', 'F'], 7: ['g', 'G'], 8: ['h', 'H'], 9: ['i', 'I'], 10:
['j', 'J']}
```

Input Format

The input contains `d1` and `d2` on separate lines.

Output Format

The output should be a sorted list of the (key, value) pairs in the merged dictionary.

INTERVIEWER GUIDELINES

Solution

```
def merge_key(d1, d2):
    d = {}
    # Iterate over the items (k, v) of d1 and d2. Insert every newly
encountered
    # k into a new dictionary as a key with [v] as the value. If k is
    # encountered again, store the corresponing value in the new
dictionary in
    # the previously created list.
    for k,v in list(d1.items()) + list(d2.items()):
        # dict.get() eliminates the need for if-else
        d[k] = d.get(k, []) + [v]
    return d

print(sorted(merge_key(d1,d2).items()))
```

CANDIDATE ANSWER

Language used: **Python 3**

```python
def merge_key(d1,d2):
    l = []
    d = {}
    if len (d1) == len (d2):
        for i in range (1, len(d1) + 1):
            l = []
            x = d1.get(i)
            y = d2.get(i)
            l.append(x)
            l.append(y)
            d[i] = l
        return d
# Enter your code here.
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	20	0.0304 sec	7.93 KB

No Comments

QUESTION 5

✓ Correct Answer

Score 20

Merge by Value › Coding

QUESTION DESCRIPTION

Problem

Write a function named `merge_value` that takes two dictionaries `d1` and `d2` as parameters and builds a dictionary that contains every value from `d1` and `d2` as key. The corresponding key in `d1` and `d2` becomes the value in the merged dictionary. For multiple values in the merged dictionary for the same key, the values are put in a list. The function *returns* a sorted list of the (key, value) pairs in the merged dictionary.

Sample

```
>>> d2 = {i:chr(64+i) for i in range(1,11)}
>>> d2
{1: 'A', 2: 'B', 3: 'C', 4: 'D', 5: 'E', 6: 'F', 7: 'G', 8: 'H', 9: 'I',
10: 'J'}
>>> d3 = {i-1:chr(64+i) for i in range(1,11)}
>>> d3
{0: 'A', 1: 'B', 2: 'C', 3: 'D', 4: 'E', 5: 'F', 6: 'G', 7: 'H', 8: 'I',
9: 'J'}
>>> merge_val(d2,d3)
{'B': [2, 1], 'C': [3, 2], 'D': [4, 3], 'I': [9, 8], 'A': [1, 0], 'G': [7,
6], 'E': [5, 4], 'F': [6, 5], 'J': [10, 9], 'H': [8, 7]}
```

Input Format

The input contains `d1` and `d2` on separate lines.

Output Format

The output should be a sorted list of the (key, value) pairs in the merged dictionary.

INTERVIEWER GUIDELINES

Solution

```python
def merge_value(d1, d2):
    d = {}
    # Iterate over the items (k, v) of d1 and d2. Insert every newly
encountered
    # v into a new dictionary as a key with [k] as the value. If v is
    # encountered again, store the corresponing key in the new dictionary
in the
    # previously created list.
    for k,v in list(d1.items()) + list(d2.items()):
        # dict.get() eliminates the need for if-else
        d[v] = d.get(v,[]) + [k]
    return d

print(sorted(merge_value(d1,d2).items()))
```

CANDIDATE ANSWER

Language used: **Python 3**

```python
def merge_value(d1,d2):
    d = {}
    l = []
    if len (d1) == len (d2):
        for i in range (len(d1) + 1):
            l = []
            if i in d1:
                x = d1.get(i)
                for k in d2:
                    if d2[k] == x:
                        l.append(i)
                        l.append(k)
                        d[x] = l
        return d
# Enter your code here.
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	20	0.0258 sec	8.05 KB

No Comments

QUESTION 6

✓ Correct Answer

Score 40

Count Words › Coding

QUESTION DESCRIPTION

Problem

Write a function named `count_words` that uses a dictionary to count the words in its parameter named `s` and of type `str`. It then *prints* the identified words in ascending order along with their frequency in `s` as shown in the sample below. Space, case, and special characters must be ignored when counting.

Sample

```
>>> count_words("Python is a widely used high-level programming language
for general-purpose programming, created by Guido van Rossum and first
released in 1991. An interpreted language, Python has a design philosophy
that emphasizes code readability (notably using whitespace indentation to
delimit code blocks rather than curly brackets or keywords), and a syntax
that allows programmers to express concepts in fewer lines of code than
might be used in languages such as C++ or Java. It provides constructs
that enable clear programming on both small and large scales.")
1991 = 1
a = 3
allows = 1
an = 1
and = 3
as = 1
be = 1
blocks = 1
both = 1
brackets = 1
by = 1
c = 1
clear = 1
code = 3
concepts = 1
constructs = 1
created = 1
curly = 1
delimit = 1
design = 1
emphasizes = 1
enable = 1
express = 1
fewer = 1
first = 1
for = 1
generalpurpose = 1
guido = 1
has = 1
highlevel = 1
in = 3
indentation = 1
interpreted = 1
is = 1
it = 1
java = 1
keywords = 1
language = 2
languages = 1
large = 1
lines = 1
might = 1
notably = 1
of = 1
on = 1
or = 2
philosophy = 1
programmers = 1
programming = 3
provides = 1
python = 2
rather = 1
readability = 1
released = 1
rossum = 1
scales = 1
small = 1
such = 1
syntax = 1
than = 2
that = 3
to = 2
used = 2
using = 1
van = 1
whitespace = 1
widely = 1
```

INTERVIEWER GUIDELINES

Solution

```
def count_words(s):
    new_s = ''
    for letter in s.lower():
        if  ord('a') <= ord(letter) <= ord('z') or letter in '01234567890
':
            new_s += letter
    d = {}
    for word in new_s.split():
        d[word] = d.get(word, 0) + 1
    for k,v in sorted(d.items()):
        print(k, '=', v)
```

CANDIDATE ANSWER

Language used: **Python 3**

```
# Enter your code here.
def count_words(s):
    n = []
    d = {}
    l = s.split()
    for i in l:
        p = ""
        if type (i) == str:
            i = i.lower()
        for k in i:
            if ord (k) > 96 and ord (k) < 123 or (ord (k) > 47 and ord (k) <
58):
                p += k
        if p != "":
            n.append(p)
        n.sort()
    for m in n:
        if m not in d:
            d[m] = 1
        else:
            d[m] += 1
    for o in d:
        print (o, "=", d[o])
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	20	0.023 sec	8.01 KB
Testcase 1	Easy	Sample case	✓ Success	10	0.0303 sec	8.06 KB
Testcase 2	Easy	Sample case	✓ Success	10	0.0349 sec	7.84 KB

No Comments

QUESTION 7 ✓

Last name first › Coding

Correct Answer

Score 100

QUESTION DESCRIPTION

Challenge

Write a function named `last_name_first` that accepts a single parameter, `t`, which is passed a list of tuples. Each tuple contains a name in parts (first name, middle name, last name). Your function should modify each name so that the last name appears first in the tuple.

Note

This function modifies a list in place and, as such, should not return any useful value.

Sample interaction

```
>>> t = [('Ahmed', 'Dawood'), ('Haroon', 'Hussain', 'Fawad', 'Rasheed'),
('Muhammad', 'Faisal', 'Amin')]
>>> last_name_first(t)
>>> print(t)
[('Dawood', 'Ahmed'), ('Rasheed', 'Haroon', 'Hussain', 'Fawad'), ('Amin',
'Muhammad', 'Faisal')]
```

Input/Output

Input and output will be handled by HackerRank.

Constraints

`t` is a list of tuples, where each tuple has one or more strings in it.

INTERVIEWER GUIDELINES

```
def last_name_first(names):
    for p in range(len(names)):
        name = names[p]
        name = (name[-1], ) + name[:-1]
        names[p] = name
```

CANDIDATE ANSWER

Language used: **Python 3**

```
def last_name_first(t):
    for j in range(len(t)):
        t[j] = (t[j][-1],) + (t[j][:-1])

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0414 sec	7.88 KB
Testcase 1	Easy	Sample case	✓ Success	10	0.0317 sec	7.92 KB
Testcase 2	Easy	Sample case	✓ Success	10	0.0364 sec	7.91 KB
Testcase 3	Easy	Sample case	✓ Success	10	0.0325 sec	7.87 KB
Testcase 4	Easy	Sample case	✓ Success	10	0.0254 sec	7.88 KB
Testcase 5	Easy	Hidden case	✓ Success	10	0.0325 sec	8.01 KB
Testcase 6	Easy	Hidden case	✓ Success	10	0.0363 sec	8 KB
Testcase 7	Easy	Hidden case	✓ Success	10	0.0318 sec	7.96 KB
Testcase 8	Easy	Hidden case	✓ Success	10	0.0387 sec	7.95 KB
Testcase 9	Easy	Hidden case	✓ Success	10	0.0245 sec	8 KB

No Comments

QUESTION 8

Correct Answer

Score 10

Loan Repayment Strategy › Coding

QUESTION DESCRIPTION

On the advice of your relative from the stick market, you have invested in stock in the hope to eventually pay off your Habib loan. Your relative sends you daily updates on your stocks in the following form.

Purchase Date	Purchase Price	Shares	Symbol	Current Price
26 Aug 2019	43.50	100	HU	47.02
27 Aug 2019	22.07	500	PTI	19.11
30 Oct 2019	51.98	200	JHR	50.14
28 Nov 2019	137.92	50	WTF	150.28

You want to find out your current earnings from this information.

In the table above, each share of *HU* is at a *profit* of `47.02 - 43.50 = 3.52`. As you have 100 shares of *HU*, your profit is `100 * 3.52 = 352`. Similarly, with *PTI* you are at a *loss* of `500 * (22.07-19.11) = 1480`. Your JHR stocks are at a *loss* of `200 * (51.98-50.14) = 268` and and your WTF stocks are at a *profit* of `50 * (150.28-137.92) = 618`. Your total profit is `352 -1480 - 268 + 618 = -778`.

Function Description

Complete the function *compute_profit* in the editor below. It returns the total profit from given stock information which is provided as a list of tuples. Each tuple contains the following items in the given order. The type of each item is shown.

- <purchase_date> : int
- <purchase_price> : float
- <shares> : int
- <symbol> : str
- <current_price> : float

Constraints

- The argument contains at least 1 tuple.
- All tuples in the list follow the format described above.

▼ Input Format For Custom Testing

The input consists of a single line which contains all information of all stocks in a single line delimited by a space character. The output is a single numeric value indicating the total profit. The input is read and passed to your function and your function's return value is printed by the program.

▼ Sample Case 0

Sample Input For Custom Testing

```
25-Jan-2001 43.50 25 CAT 92.45 25-Jan-2001 42.80 50 DD 51.19 25-Jan-2001
42.10 75 EK 34.87 25-Jan-2001 37.58 100 GM 37.58
```

Sample Output

```
1101
```

Explanation

The input contains information of 4 stocks. The name and corresponding profit from each are as follows.

- CAT: `25 * (92.45 - 43.5) = 1223.75`
- DD: `50 * (51.19 - 42.8) = 419.5`
- EK: `75 * (34.87 - 42.1) = -542.25`
- GM `100 * (37.58 - 37.58) = 0`

The total profit is therefore `1223.75 + 419.5 - 542.5 + 0 = 1101`

▼ Sample Case 1

Sample Input For Custom Testing

```
TODO: ADD_SAMPLE_INPUT
```

Sample Output

```
TODO: ADD_SAMPLE_OUTPUT
```

Explanation

TODO: ADD_EXPLANATION

INTERVIEWER GUIDELINES

Solution

```
import math
def compute_profit(stock_info):
    profit = 0
    for _, cost, qty, _, price in stock_info:
        profit += qty * (price - cost)
    return profit
```

CANDIDATE ANSWER

Language used: **Python 3**

```
# Enter your code here.
def compute_profit(stock_info):
    x = 0
    for i in stock_info:
        p = i[2] * (i[4] - i[1])
        x += p
    return (round(x))
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0375 sec	8.1 KB

No Comments

QUESTION 9

✓

Correct Answer

Score 100

Get Positions › Coding

CS101 NestedLists Lists Tuples

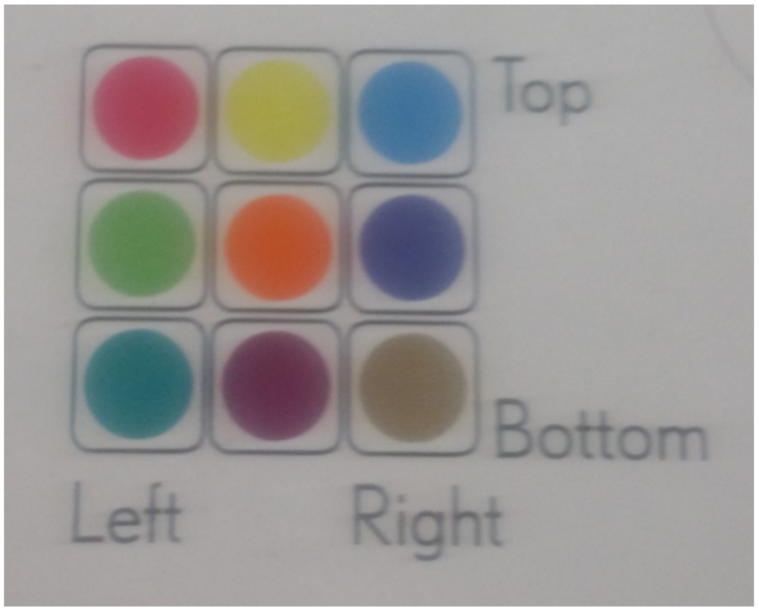
QUESTION DESCRIPTION

Problem

Define a function named `get_positions` which takes the `board` and `color` as arguments

Define a function named `get_positions` which takes the `board` and `color` as arguments and returns a list of all positions (tuple containing the row,col) at which this color is on in the board.

Sample



```
>> board=[["Pink", "Yellow", "LightBlue"],
["Green", "Orange", "DarkBlue"], ["Teal",
"Purple", "Gold"]]
>> get_positions(board, "Yellow")
   [(0,1)]
>> get_positions(board, "Red")
   []
```

Input Format

The input consists of a board on the first line. The second line contains value for `color`

Output Format

The output should be a list of all positions (tuple containing the row,col) at which this color is on in the board.

INTERVIEWER GUIDELINES

```
def read_board():
    '''read_board() -> list of lists.

    Reads a sequence of 9 space separated colors from console and
    returns them arranged as a board.
    '''
    board = input().strip().split()
    return [board[:3], board[3:6], board[6:]]

def get_positions(board, color):
    '''get_positions(list, str) -> list of pairs

    Returns all positions of color on board. Each position is
    represented as a pair (row,col) with row and col 0-indexed row
    and column numbers.
    '''
    pos = []
    for i in range(3):
        for j in range(3):
            if board[i][j] == color:
                pos.append((i,j))
    return pos

board = read_board()
color = input()
print(get_positions(board, color))
```

CANDIDATE ANSWER

Language used: **Python 3**

```
# Enter your code here.
def get_positions(board, color):
    l = board.split()
    p = []
    if color not in board:
        return []
    else:
        for i in range(len(l)):
            if l[i] == color:
                t = (i // 3),
                t = t + (i % 3,)
                p.append(t)
        return p
board = input()
color = input()
print(get_positions(board, color))
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	✓ Success	10	0.0264 sec	8 KB
Testcase 1	Easy	Sample case	✓ Success	10	0.0366 sec	7.93 KB
Testcase 2	Easy	Hidden case	✓ Success	10	0.0416 sec	7.88 KB
Testcase 3	Easy	Sample case	✓ Success	10	0.0255 sec	7.8 KB
Testcase 4	Easy	Hidden case	✓ Success	10	0.0293 sec	7.79 KB
Testcase 5	Easy	Hidden case	✓ Success	10	0.0262 sec	7.92 KB
Testcase 6	Easy	Sample case	✓ Success	10	0.0351 sec	7.88 KB
Testcase 7	Easy	Hidden case	✓ Success	10	0.0303 sec	7.86 KB
Testcase 8	Easy	Hidden case	✓ Success	10	0.0528 sec	8.08 KB
Testcase 9	Easy	Hidden case	✓ Success	10	0.0379 sec	7.91 KB

No Comments