

IS111: Introduction to Programming

Lab Exercise 06: Tuples and Parameter Passing

Optional submission. Deadline: the coming Sun @ 2359 hrs (G12 and G13)

INSTRUCTIONS



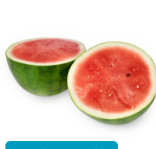



You are expected to read through the accompanying lesson 06 notes before attempting lab exercise 05. You should attempt the questions using Visual Studio Code or Pycharm.

If Level 1 questions are used as in-class exercises, it is optional for you to submit them in your Lab Assignments. Please do use the discussion forum if you have any difficulties in doing Level 1 questions. You are also encouraged to post any related questions in the forum. :)

Level 1 Questions (no submission required; use the discussion forum)

1.1 Fruits

Redmart sells different types of fruits, such as bananas, rock melons, and so on.

 <p>any 4 save 10%</p> <p>Prime Asia Cavendish Snack Pack Banana 415 g</p> <p>4D</p> <p>\$1.95</p> <p>★★★★☆ (91)</p> <p>ADD TO CART</p>	 <p>Rock Melon (Large)</p> <p>1.5 kg</p> <p>8D</p> <p>\$8.50</p> <p>★★★★☆ (95)</p> <p>ADD TO CART</p>	 <p>any 4 save 10%</p> <p>Global Seasons Red Flesh Seedless Watermelon 3 kg</p> <p>4D</p> <p>\$6.25</p> <p>★★★★☆ (47)</p> <p>ADD TO CART</p>	 <p>GIVVO Honeydew Melon</p> <p>1 per pack</p> <p>8D</p> <p>\$4.50</p> <p>★★★★☆ (46)</p> <p>ADD TO CART</p>	 <p>NEW</p> <p>USA White Peaches 4s</p> <p>500 g</p> <p>3D</p> <p>\$5.90</p> <p>★★★★☆ (13)</p> <p>ADD TO CART</p>	 <p>YUVVO Large Limes</p> <p>200 g</p> <p>6D</p> <p>\$1.10</p> <p>★★★★☆ (77)</p> <p>ADD TO CART</p>
--	--	---	---	--	--

Write a Python function `get_cheapest_fruit` that takes in a single list parameter `price_list` that contains a list of tuples of the price of each fruit. The tuple is in the form `(fruit_name, price)`, for example `('banana', 1.95)`.

The function should return the fruit that has the cheapest price. You may assume that all the fruits have different prices and hence, you only need to return one fruit.

You can use the following code snippet to test your code:

```
# the following statements should return True
print(get_cheapest_fruit([('banana',1.95), ('rockmelon',8.50),
('watermelon',6.25), ('melon',4.50), ('peach',5.90), ('limes',1.10)])
== 'limes')
print(get_cheapest_fruit([('lemons',4.05), ('apples',4.00),
('oranges',3.75)]) == 'oranges')
```

1.2 Grab Rewards

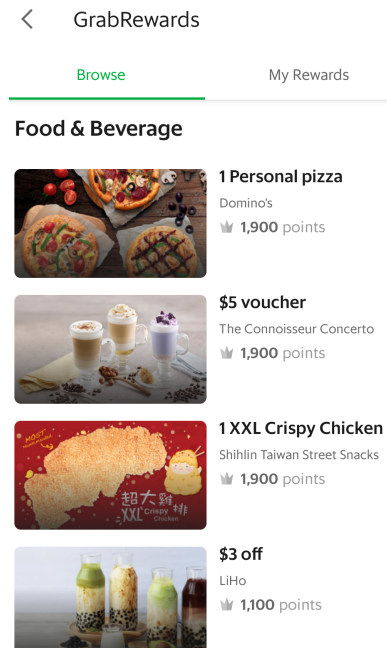
After considerable effort, you have now managed to earn enough Grab points to redeem some items from GrabRewards. You open the Grab app and decide that you want to use *all* your points to redeem the '\$3 off' vouchers from LiHo, so that you can buy lots of bubble tea to drink while working on your IS111 lab exercises.

Write a Python function `get_rewards` that takes in two integer parameters:

`total_points`, and `points_required`,

which represent the total points that you currently have, and the points required to redeem the item that you want, respectively.

Your function should return a tuple in the form `(max_rewards, remaining_points)`, which represent the maximum number of rewards that you can redeem, and the remaining points that you have after redeeming those rewards.



You can use the following code snippet to test your code:

```
# the following statements should return True
print(get_rewards(4300,1100) == (3,1000))
print(get_rewards(20540,360) == (57,20))
print(get_rewards(2300,3000) == (0,2300))
```

Level 2 Questions

2.1 Shopping Cart

You have decided to purchase some fruits on Redmart. The fruits that you have decided to purchase are given as a list of tuples, where each tuple is in the format (fruit_name, price, quantity). For example, the tuple ('banana', 1.95, 4) means that you have decided to purchase 4 bananas, each of which costs \$1.95.

Write a Python function `compute_shopping_cart`, which takes in a single list parameter `shopping_list` that contains the list of tuples of the fruits that you have decided to purchase. The function returns the total amount of the shopping cart (i.e., total amount of money you have to pay for all the fruits that you have decided to purchase).

You can use the following code snippet to test your code:

```
# the following statements should return True
print(compute_shopping_cart([('banana',1.95,1), ('rockmelon',8.50,3),
('watermelon',6.25,2), ('melon',4.50,1), ('peach',5.90,0),
('limes',1.10,6)]) == 51.05)
print(compute_shopping_cart([('lemons',4.05,4), ('apples',4.00,2),
('oranges',3.75,3)]) == 35.45)
```

3 Brownie Points 🍪

3.1 Tic-Tac-Toe Checker

In the game of Tic-Tac-Toe, two players (X and O) take turns to mark spaces in a 3×3 grid. The player who succeeds in placing three of his/her marks in a horizontal, vertical, or diagonal row wins the game. Each of the 9 spaces in the Tic-Tac-Toe may be marked with X , O or $-$ (which means un-marked). Write a Python function `tic_tac_toe_checker` which takes in a single input parameter `marked_play`, and returns either:

- X , if player X has won;
- O , if player O has won; or
- D , if the game is a *Draw*.

You may assume that the list `marked_play` contains 9 elements, which can be X , O or $-$. For instance, if `marked_list = ['X', 'X', 'O', '-', 'X', 'O', '-', 'O', 'O']`, this represents:

X	X	O
-	X	O
-	O	O

You may also assume that the usual rules of Tic-Tac-Toe apply. For instance, there can only be one winner at any one time. You can use the following code snippet to test your code:

```
# the following statements should print True
print(tic_tac_toe_checker(['X', 'X', 'O', '-', 'X', 'O', '-', 'O', 'O']) == 'O')
print(tic_tac_toe_checker(['X', 'X', 'O', 'O', 'X', 'O', '-', 'O', 'X']) == 'X')
print(tic_tac_toe_checker(['X', 'X', 'O', 'O', 'O', 'X', 'X', 'X', 'O']) == 'D')
```

💬 **This question was part of an old question in previous lab test 2.**