Coding exercise process goes like this:

1. Thoroughly read the exercise below, if you have any questions, email debbie.alvarez@eargo.com or vasant.patel@eargo.com
2. Complete the exercise within 3 days of receiving this document using the programming language of your choice
3. When you are finished, send us a link to the code repository (­Github or BitBucket)
4. Please provide instruction on how to execute your code with different test cases
5. We will contact you to set up a time to chat about your submission.

For your solution, we are interested in seeing a couple of things:

* How do you approach a problem?
* How efficient is your code?
* How readable and documented is your code?

Please solve the following challenges.

**Exercise 1:**

Write a function to calculate change for a given number (number represents cents)

The function will return list, vector or collection of numbers representing the number of

quarters(25), dimes(10), nickels (5), pennies (1) that would yield the correct change.

# Example:

# For 83¢, the function will return 3 quarters, 0 dimes, 1 nickel, and 3 pennies.

**Solution:**

**def change(amount):**

**amount = int(amount)**

**quarters, amount = divmod(amount, 25)**

**dimes, amount = divmod(amount, 10)**

**nickels, amount = divmod(amount, 5)**

**pennies = round(amount / 1,0)**

**return list(map(int, [quarters, dimes, nickels, pennies]))**

**Exercise 2:**

Write an SQL Statement to transform data from Input to output format. Input and output table are provided below.

Under the output table format you need to replace ?? with actual record counts while submitting your answers.

**Input\_table format**

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Home\_Page** | **Product\_Page** | **Warranty\_Page** |
| 1 | 1 | 1 | 1 |
| 2 | 1 | 1 | 0 |
| 3 | 1 | 0 | 1 |
| 4 | 1 | 0 | 0 |
| 5 | 0 | 1 | 1 |
| 6 | 0 | 1 | 0 |
| 7 | 0 | 0 | 1 |
| 8 | 0 | 0 | 0 |

**Output table format**

|  |  |
| --- | --- |
| **Page** | **counts** |
| Home\_Page | ?? |
| Product\_Page | ?? |
| Warranty\_Pagge | ?? |

Note: replace ?? with actual record counts when you submit your answer

**Solution:**

Created a table called Page\_count which stores the input table information.

**select 'Home\_Page' as PAGE,count(home\_page) as count**

**from page\_count**

**where home\_page = 1**

**group by home\_page**

**union**

**select 'Product\_Page' as PAGE,count(Product\_Page) as count**

**from page\_count**

**where Product\_Page = 1**

**group by Product\_Page**

**union**

**select 'Warranty\_Page' as PAGE,count(Warranty\_Page) as count**

**from page\_count**

**where Warranty\_Page = 1**

**group by Warranty\_Page;**

\*\*This query can be used as in Insert statement to create the output table (was not sure if output table was to be created or just output should be in the format above)

**Exercise 3:**

This is a 2 part exercise.  If you can do part A or Part B or Both.

Write a function to find out the best Buying and Selling day for maximum gain from daily stock prices of the last 10 days. Following are 2 rules.

**Part A:**

1. Buy first
2. You can only buy once and sell once

**Part B:**

1. You buy first and then sell
2. Buy and sell as many times as possible. Goal is to maximize profit.

For both parts, you can not buy and sell on the same day.

**Use Case for Testing:**

Below table represents stock\_price for 10 days price. Please do provide at least 2 more use cases of your own along with the solution apart from the use case provided below.

|  |  |
| --- | --- |
| **Day** | **Stock Price** |
| Day 1 | $3 |
| Day 2 | $8 |
| Day 3 | $8 |
| Day 4 | $55 |
| Day 5 | $38 |
| Day 6 | $1 |
| Day 7 | $7 |
| Day 8 | $42 |
| Day 9 | $54 |
| Day 10 | $53 |

**Solution:**

**Need to create a cvs (stock.csv) with Day1 to Day10 stock values to run the solution.**

Created a stocks.py file for the above question, which accepts the Day1 to Day10 file as input from csv file. The solution I have separately added to the github. I have added the solution.

The other 2 different use cases are:

|  |  |
| --- | --- |
| Day | Stock Price |
| Day 1 | 1 |
| Day 2 | 23 |
| Day 3 | 5 |
| Day 4 | 67 |
| Day 5 | 1 |
| Day 6 | 1 |
| Day 7 | 65 |
| Day 8 | 65 |
| Day 9 | 23 |
| Day 10 | 54 |

**The output of the above test case is Day 1 we buy the stock and Day 4 we sell the stock.**

|  |  |
| --- | --- |
| Day | Stock Price |
| Day 1 | 321 |
| Day 2 | 123 |
| Day 3 | 432 |
| Day 4 | 223 |
| Day 5 | 244 |
| Day 6 | 522 |
| Day 7 | 122 |
| Day 8 | 110 |
| Day 9 | 123 |
| Day 10 | 422 |

**The output of the above test case is Day 2 we buy the stock and Day 6 we sell the stock.**