First AML

Back End Technical Test

The purpose of this test is to give a basis for discussion about the way in which you approach work, and your technical knowledge around back end development. You can choose any mainstream OO language you prefer, though C# is preferable if you have good experience in that.

Please don't take more than 2 hours

Things we are looking for:

- Well structured code
- Tests
- A record of any assumptions you make as you go
- The next steps you would take and what things you wish you had done better
- Code in a repository we can access, with well structured commits as you go

Things we are not looking for:

- A solution to every step. Seriously. We don't expect you to complete the task within the time limit. Rushing through tasks to get further is more likely to do you a disfavor. Just think through what you might do for the steps you haven't completed so we can use it as a basis for discussion.
- Code tests are stressful, so we don't expect a perfect solution. If you think something you've done is bad, just let us know along with your submission. Being able to criticize your own work is a skill we value!

We would rather see a nicely structured solution with tests to the first couple of steps, than a compromise to complete more steps.

Task Introduction

You work for a courier company and have been tasked with creating a code library to **calculate the cost of sending an order of parcels**.

- You are required to build a library that can be consumed by other code. **Do not** build a CLI, HTTP, or any other interaction layer. The library being consumed by your tests is all we want to see

- The input can be in any form you choose
- Output should be a collection of items with their individual cost and type, as well as the total cost
- In all circumstances **the cheapest option for sending each parcel should be selected**
- Try not to peek ahead at future steps and **commit your working as you go.** The test is designed to introduce changing requirements, and we like to see how the code deals with that

Implementation Steps

1) The initial implementation just needs to calculate cost based on a parcel's size. For each size category there is a fixed delivery cost

- ○ Small parcel: all dimensions < 10cm. Cost $3
- ○ Medium parcel: all dimensions < 50cm. Cost $8
- ○ Large parcel: all dimensions < 100cm. Cost $15
- ○ XL parcel: any dimension >= 100cm. Cost $25

2) Thanks to logistics improvements we can deliver parcels faster. This means we can charge more money. Speedy shipping can be selected by the user to take advantage of our improvements.

- ○ Speedy shipping doubles the cost of the entire order
- ○ Speedy shipping should be listed as a separate item in the output, with its associated cost
- ○ Speedy shipping should not impact the price of individual parcels, i.e. their individual cost should remain the same as it was before

3) There have been complaints from delivery drivers that people are taking advantage of our dimension only shipping costs. A new weight limit has been added for each parcel type, over which a charge per kg applies

+$2/kg over weight limit for parcel size:

- Small parcel: 1kg
- Medium parcel: 3kg
- Large parcel: 6kg
- XL parcel: 10kg

4) Some of the extra weight charges for certain goods were excessive. A new parcel type has been added to try and address overweight parcels

Heavy parcel, $50 up to 50kg +$1/kg over 50kg

5) In order to award those who send multiple parcels, special discounts have been introduced.

- Small parcel mania! Every 4th small parcel in an order is free!
- Medium parcel mania! Every 3rd medium parcel in an order is free!
- Mixed parcel mania! Every 5th parcel in an order is free!

- Each parcel can only be used in a discount once
- Within each discount, the cheapest parcel is the free one
- The combination of discounts which saves the most money should be selected every time

Example:
6x medium parcels. 3x $8, 3 x $10. 1st discount should include all 3 $8 parcels and save $8. 2nd discount should include all 3 $10 parcels and save $10.

- Just like speedy shipping, discounts should be listed as a separate item in the output, with associated saving, e.g. “-$2”
- Discounts should not impact the price of individual parcels, i.e. their individual cost should remain the same as it was before
- Speedy shipping applies after discounts are taken into account