Developer Challenge

Instructions:

- Attempt **any one of the two problems** given below.
- Develop the solution using Java, Python, Ruby, C, C++.
- Console interface is enough.
- Even though these are really small problems, try to write production quality code instead of write-once and throw kind of code (refactor into small, recognisable modules, use frameworks where needed to save code, write tests, etc. No need for CI, or scale-out in mind, though.)
- Please use object oriented design when designing your code.
- Share your code via Github. Make sure the code is accessible publicly, so it is easier to review.
- The deadline for the submission of this assignment is mentioned in the email. For further questions, email us.

All the best!

Problem 1: Sales Tax

Basic sales tax is applicable at a rate of 10% on all goods, except books, food, and medical products that are exempt. Import duty is an additional sales tax applicable on all imported goods at a rate of 5%, with no exemptions.

When I purchase items I receive a receipt which lists the name of all the items and their price (including tax), finishing with the total cost of the items, and the total amounts of sales taxes paid. The rounding rules for sales tax are that for a tax rate of n%, a shelf price of p contains (np/100 rounded up to the nearest 0.05) amount of sales tax.

Write an application in **Python** that prints out the receipt details for these shopping baskets.
INPUT:
Input 1:
1 book at 12.49
1 music CD at 14.99
1 chocolate bar at 0.85

Input 2:
1 imported box of chocolates at 10.00 1 imported bottle of perfume at 47.50

Input 3:
 1 imported bottle of perfume at 27.99 1 bottle of perfume at 18.99
 1 packet of headache pills at 9.75
 1 box of imported chocolates at 11.25

OUTPUT:
 Output 1:
 1 book: 12.49
 1 music CD: 16.49
 1 chocolate bar: 0.85 Sales Taxes: 1.50 Total: 29.83

Output 2:
 1 imported box of chocolates: 10.50 1 imported bottle of perfume: 54.65 Sales Taxes: 7.65
 Total: 65.15

Output 3:
 1 imported bottle of perfume: 32.19 1 bottle of perfume: 20.89
 1 packet of headache pills: 9.75
 1 imported box of chocolates: 11.85 Sales Taxes: 6.70
 Total: 74.68

Problem 2: Game of Life

The universe of the Game of Life is an infinite two-dimensional orthogonal grid of square cells, each of which is in one of two possible states, live or dead. Every cell interacts with its eight neighbours, which are the cells that are directly horizontally, vertically, or diagonally adjacent. At each step in time, the following transitions occur:

1. Any live cell with fewer than two live neighbours dies, as if by loneliness.
2. Any live cell with more than three live neighbours dies, as if by overcrowding.
3. Any live cell with two or three live neighbours lives, unchanged, to the next generation.
4. Any dead cell with exactly three live neighbours comes to life.

The initial pattern constitutes the 'seed' of the system. The first generation is created by applying the above rules simultaneously to every cell in the seed - births and deaths happen simultaneously, and the discrete moment at which this happens is sometimes called a tick. (In other words, each generation is a pure function of the one before.) The rules continue to be applied repeatedly to create further generations.

Problem:
The below inputs provide the pattern or initial cells in the universe, especially their (x,y) co-ordinates.
The output is the state of the system in the next tick (one run of the application of all the rules) , represented in the same format - i.e. x,y coordinates of all the alive cells after one tick.

Input A:
(Block pattern - Still life)
1, 1
1, 2
2, 1
2, 2
Output A:
1, 1
1, 2
2, 1
2, 2
--- -------------------

Input B
(Boat pattern - Still life)
0, 1
1, 0
2, 1
0, 2
1, 2
Output B
0, 1
1, 0
2, 1
0, 2
1, 2
--- -------------------

Input C
(Blinker pattern - oscillator)
1, 1
1, 0
1, 2
Output C

```
 1, 1
 0, 1
 2, 1
----------------------------------------------------------------------------------------------- ---------------------
 Input D
 (Toad pattern - two phase oscillator)
 1, 1
 1, 2
 1, 3
 2, 2
 2, 3
 2, 4
 Output D
 0, 2
 1, 1
 1, 4
 2, 1
 2, 4
 3, 3
==========
```