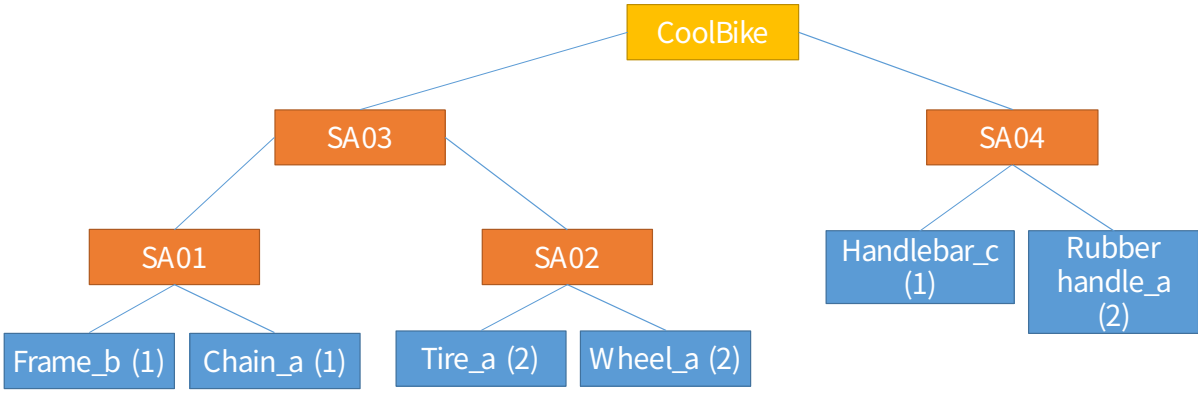
DSP Assignment 02 (Groups of 1 or 2 students)

MRP - Material Requirements Planning

In a manufacturing company, Material Requirements Planning (MRP) concerns making sure to have enough materials, i.e., parts and/or sub-assemblies, to assemble the goods required by the clients. In this assignment, your task is to implement a simple MRP application.

The Bill of Materials (BOM) identifies, for each product that a company manufactures, the list of parts and sub-assemblies that are required to assemble it. The BOM normally has a multi-level (or tree[1]) structure. The figure below, for instance, shows a simple BOM of a bicycle named "CoolBike". Assembling a CoolBike requires 2 tires of type *tire_a*, 1 frame of type *frame_b*, 1 chain (type a), 2 wheels (a), 1 handlebar (c) and 2 rubber handles (a). Note that a CoolBike can be assembled using, for instance: (i) only parts (i.e., only the "leaves" of the BOM), (ii) 1 unit of SA03 and 1 unit of SA04, (iii) 1 unit of SA04, 1 Frame (b), 1 Chain (a) and 1 unit of SA02, etc.



Companies normally maintain a list of the parts they have in stock, storing the part id, the part name, quantity available in stock and unit price[2]. A part list in the bicycle example looks like this:

Part name	ID	Quantity in stock	Unit price
Frame_a	FR-01	20	20000
Frame_b	FR-02	26	35000
Tire_a	TI-01	8	10000
Wheel_a	WH-01	23	17000
Etc.			

Companies also maintain a list of sub-assemblies in stock. Customers, in fact, may change their orders, often at the last minute, leaving some assemblies unused, but of course reusable for other orders.

---

1 Note that in this assignment, for simplicity, we only consider binary tree structures

2 we assume that the supplier of each part is fixed, so there is only one possible price for each part

Example:

Sub-assembly type	Quantity in stock
SA01	10
SA03	5
Etc.	

The company manufactures two types of bicycles, i.e, CoolBike and BoringBike.

Orders from a customer assume the form <product, quantity required>, e.g., <BoringBike, 17> to order 17 boring bikes.

Given a customer order, your MRP application should allow (i) to calculate and verify the material requirements, (ii) to procure the spare parts required if not available (procurement) and (iii) to execute the order. More in detail:

Calculate and verify the material requirements: given an order, MRP should assess which parts and/or sub-assemblies required to manufacture an order are available, which are not available (and therefore have to be ordered from suppliers) and the budget required to buy the necessary parts from suppliers. In doing so, MRP should consider that priority should be given to utilising first the unused sub-assemblies in stock. That is (ref. the CoolBike BOM example), if one unit of SA03 and one unit of SA04 are available, then the next CoolBike has to be manufactured using these sub-assemblies. We say that an order can be "honoured" if it can be manufactured using the parts and/or sub-assemblies currently in stock at the company.

Procurement: MRP also provides functionality to manage the procurement of parts required for an order. When an order cannot be honoured with parts and sub-assemblies in stock, parts are bought from suppliers. In your MRP application, procurement updates the list of parts in stock. Assume that, to avoid surprises, parts are always procured in excess of 2 units if 7 or more units have to be ordered or in excess of 1 unit if 6 or less units must be ordered (example: if you need 9 units of part P to honour order, your MRP will order 11 = 9 + 2 units of P; if you need 5 units of part Q to honour an order, your MRP will order 5 + 1 = 6 units of Q). Assume also that there is always budget available to buy the required parts (= you do not have to worry about "money" in your MRP).

Execute an order: if an order is "honourable" or after procurement, the products in the order are manufactured. As a result, the MRP should update the list of parts and sub-assemblies after manufacturing is completed.

Given a list of orders as input, after each order is executed, MRP should do the following:

- print the current list of parts and work in progress **on console**

- print the outcome of order execution **in submit.txt.** (see detailed example later)

To avoid the number of assemblies decreasing to 0, every time an order is executed, MRP will always add X/6 units of all the sub-assemblies required to manufacture type of bike that was ordered, where X is the number of bikes ordered. Example: for the order <CoolBike, 20>, after the order is executed your MRP will add 20/6=3 (round down to the closest integer) units of sub-assemblies SA01, …SA04 to the stock[3]. If X = 6 or less, then you will add 1 unit of all sub-assemblies.

---

3 This is a totally artificial "trick", but it is required to maintain sub-assemblies in stock without the need of simulating the fact that some sub-assemblies

Files required:

Everything that you need in assignment02-student_2.zip, which includes:
mrp_solution.py: this is the file in which you have to put your solution. There is a class "MRP" that you have to complete
tree.py, ***binary_tree.py***, ***linked_binary_tree.py***, ***exceptions***, ***decision_tree.py***: You need the "decision tree" to store the BOMs of bicycles. These are all the files that you need to import to use a Decision Tree. For your convenience they are copied in the same package of the mrp_solution.py file
boringbike.txt, ***coolbike.txt***: BOM of the 2 types of bicycle.
parts.txt, ***subassemblies.txt***: these files contain the initial lists of parts and subassemblies
order.txt, ***test.py***, ***submit.txt***: these files are required to run automated tests of your solutions (see explanation below).

How to develop and test your MRP:

1) Develop your solution in the file mrp_solution.py

2) To test your solution during development, you can always use the "main" function in mrp_solution.py

3) Once you have a complete solution, you can run the file test.py: this file runs a number of automated tests of your solution and checks the content of the file submit.txt produced by execution of mrp_solution.py

Examples can be found in the folder "test-cases".

For instance, given the list of orders in file in 0-input.txt, the content of submit.txt produced by the execution of mrp_solution.py should be EXACTLY THE SAME as the file 0-output.txt (note that the outcome written in submit.txt is different whether an order can be honoured without procurement or not)

Submission Instructions:

You have to conduct this project in **the same group as assignment 1**.
DEADLINE: Monday, November 26th at 10pm.
Please zip the folder "assignment02-student_2" on your computer and upload the zip file.

IMPORTANT: BEFORE SUBMITTING, MAKE SURE TO FILL OUT GROUP MEMBER INFORMATION IN THE FILE mrp_solution.py (for each student, name in English and student id). The zip file must also include a txt file containing the link to a video demo of your MRP

As you did with assignment 1, you have to record a **video demo** of your MRP. The demo must show the execution of all the test cases. Comments in your video (in speaking or as subtitles) are welcome. The video should not exceed 3 minutes.

You have to **give a demo** of your code to the TA. Details about the schedule will be provided in due time. You are not allowed to change your code between the submission deadline and

---

may remain unused (because customers change orders or issues with the manufacturing process).

the demo (we will check!). "Giving a demo" means to show the functionality of your software. So, you have to develop appropriate code in the main() of your application to showcase the implemented functionality. Failing to demonstrates the implementation of (some of the) functionality will lead to point deductions.

Late submission policy: *Up to 6 hours late:* the grade will decrease of 5 points for each ½ hour (e.g., your submission is 1.5 hour late, your grade will decrease of 15 points). *More than 6 hours late:* these cases are critical and a decision will be taken on a case-by-case basis (note that all assignments are graded on a scale of 100 points)