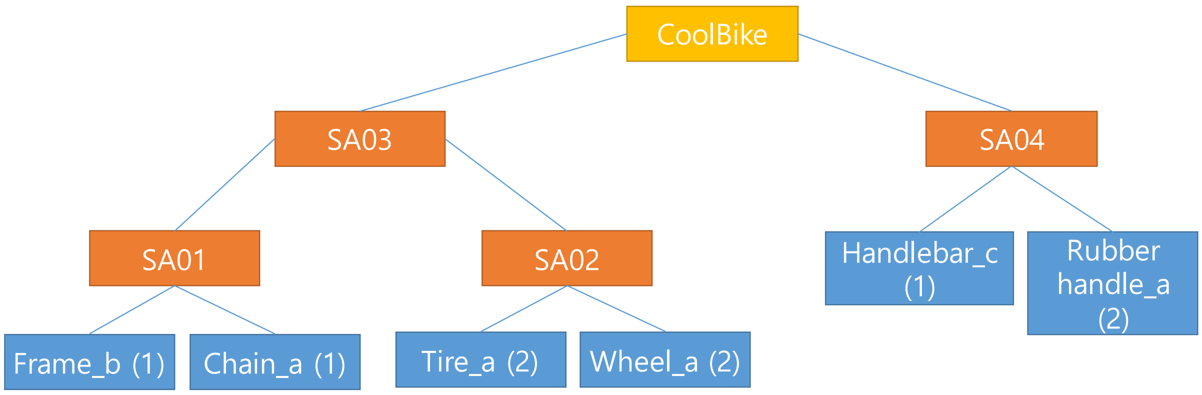
ADSA Assignment 02

MRP - Material Requirements Planning

In a manufacturing company, Material Requirements Planning (MRP) concerns making sure that we 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. Figure 1, for instance, shows a simple BOM for a bicycle named "CoolBike". Assembling a CoolBike requires 2 tyres of type *tires_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 CoolBicycle 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 SA03, 1 Frame (b), 1 Chain (a) and 1 unit of SA02, (iv) 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. Example:

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

[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

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 check 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 provide 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 a random [1%-5%]. Assume that there is always budget available to buy the required parts.

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. In doing this, assume that any manufacturing process always yields 5% of the sub-assemblies required in a product in excess. Example: for the order <CoolBike, 20>, the manufacturing process will yield 1 (=5% of 20) unit of the assemblies SA01, …SA04 in stock after the order is executed.

MRP should allow to print the current list of parts and the current list of sub-assemblies.

Files required: download the zip file assignment02.zip and unzip it on your computer in the ADSA PyCharm project. The BOM of CoolBike and BoringBike are given in the files “coolbike.dat” and “boringbike.dat”, respectively (note that some parts and sub-assemblies are sheared by the two bikes). Note that that in these files the BOM tree is traversed in postorder (similarly to the Decision Tree example of Week 6); the initial list of parts in stock is given in the file “parts.dat” and the initial list of sub-assemblies in stock in “subassemblies.dat”.

When initialised, the MRP application should read the files mentioned above and store the information in them in appropriate data structures to be ready to process orders.

Notes and suggestions:

- You must implement your MRP using the class “MRP” provided in PyCharm. This class should contain all the methods and data structures that you need.
- Think of the BOM as a tree (or, better, a DecisionTree…) …if a sub-assembly in a node is available, then there is no-need to explore the sub-tree of that node….
- On top of the requirements described above, try to think of a set of features that will facilitate the “user experience”, e.g., printing meaningful and nicely formatted messages

while a method is executing, or even provide additional functionality, such as printing a BOM, visualise the historic list of orders, etc.
In summary, the requirements listed in this document are the “minimal” requirements of your MRP application, you are of course free to impress us with improved and more advanced/user-friendly functionality ☺

Instructions:

- You have to conduct this project in **groups of 2 or 3 people**.
  Once a group is formed, please send an email with names and ids to Aikerim. You have to choose one member of the group to become “the captain”: the captain is the one who will have to make the submission on blackboard.
- You have to **submit the code** that you develop **on blackboard** by the **deadline of Wednesday October 26 at 10pm**.
  Please zip the folder “assignment02” on your computer and upload the zip file.
  The submission area will be available in due time.
- You have to **give a demo** of your code to the TA, Ms Aikerim Orken. This has to take place in the period October 26-31. More instructions about how to notify groups and schedule an appointment with Aikerim will be provided in due time.
  <u>You are not allowed to change your code between the submission deadline and the demo</u> (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.
  <u>All group members must be present at the demo and must demonstrate that they know the code.</u>