PPA Assignment 10

This coursework is designed to help you revise the content from Weeks 1-5 and 7-11.

Do you think there is a problem with any of the content below? Let us know immediately at programming@kcl.ac.uk.

Read through this brief carefully before starting your attempted solution and make sure you follow it as closely as possible, paying particular attention to the names required for elements of your code. Also ensure that you comment your code, as part of the assignment documentation.

You are advised to always access the latest version of this brief through KEATS.

For this week's assessment, consider the following scenario, and then complete the tasks that follow it:

A farmer wishes to harvest her crops.

1. Model this scenario based on the following requirements:
    1. Model a `Crop`
        1. Every crop has a `type` (e.g. "Corn", "Wheat", etc.), and a `value` that would be made in profit if that crop were to be sold.
        2. It should be possible to understand the `value` of a crop.
    2. Model a `Field`
        1. Every field consists of a set of `Crop`s, which are all of the same type and value in our simple implementation.
        2. The maximum number of crops that any field can contain is 10.
        3. When a field is made, it should be possible to specify the `type` and (individual) `value` of the crops that are to be planted in that field.
        4. It should be possible to `plant` the maximum number of crops that can exist in a field. When a field is first made, all crops should be `plant`ed in this way.
        5. It should be possible to `harvest` a field, such that the total `value` of all the crops in the field is calculated and returned. As the result of a `harvest`, all of the `Crop`s in a field are removed.
    3. Model a `Harvester`, which is a small machine used for harvesting crops:
        1. Every harvester has a `fuelTankSize` and a `topSpeed`.

    2. A harvester has a `harvestingCapacity`, which determines how many fields that harvester can harvest in a single harvest (!). `harvestingCapacity` is determined by adding the `fuelTankSize` to the `topSpeed` of the harvester.

4. Model a `CombineHarvester`, which is a larger type of `Harvester`:

    1. A combine harvester differs from a normal harvester in that it has a cutter bar at the front, which has a certain `length`.

    2. This cutter bar allows the combine harvester to harvest more fields, such that its `harvestingCapacity` is the same as the harvesting capacity of a normal `Harvester`, multiplied by the `length` of the cutter bar.

5. Model a `Farm`:

    1. A farm consists of a number of `Field`s.

    2. A farm consists of a number of `Harvester`s.

    3. It should be possible to `add` a `Harvester` to a farm.

    4. It should be possible to `add` a `Field` to a farm, with a certain `type` of crop with a certain (individual) `value`.

    5. It should be possible to understand the `profit` a farm has made.

    6. It should be possible for a `Farm` to have a `harvest`, in which the `Field`s on the `Farm` are `harvest`ed. The total number of fields that are harvested in one harvest should depend on the total `harvestingCapacity` of all the harvesters the farm owns. If this capacity is less than the number of fields, then not all of the fields can be harvested. If this capacity is equal to or greater than the number of fields, then all of the fields can be harvested. After each `Field` is `harvest`ed, we assume the farm sells the crops from that field, such that the farm's `profit` is updated accordingly.

2. Create a class `Harvest`, which can be compiled and run from the command line. Use this class to do the following (in order), using the classes and methods you have created for Question 1.

1. Create a `Farm`.

2. Add a `Harvester` to the farm, with a `fuelTankSize` of 1 and a `topSpeed` of 1.

3. Add a `CombineHarvester` to the farm, with a `fuelTankSize` of 2, a `topSpeed` of 2 and a `length` of 3.

4. Add 20 `Field`s to the `Farm`: 5 of corn, 5 of wheat, 5 of oats and 5 of barley. Each crop should have a value of 20.

5. Run a `harvest` using the `Farm`.

6. Print the `Farm`'s `profit`, which should be £2800.

Once completed, submit your assignment using the link marked `Assignment 10: Nexus Submission Link' on KEATS.

You must complete the plagiarism and collusion training before submitting this assignment.

You must also submit complete documentation of your solution. You will find a sample piece of documentation in the Support section on KEATS marked `Sample Assignment Documentation'. Submit your documentation using the link marked `Assignment 10: Documentation Submission' on KEATS.

Students who do not submit documentation along with their code, or vice-versa, will receive a mark of zero.

Any submitted code or documentation that is found to be unduly similar to the code or documentation submitted by any other student(s), will result in a penalty for those involved.

Provisional mark for your code will be released on KEATS within one week of submission. Final assignment grades will be submitted to the exam board at the end of the semester, and will take into consideration the quality of your documentation and the quality of the comments written into your code directly.

For all other queries, see the Support section on KEATS, specifically the document marked `Introduction'.