## 15.

1. The line of code (or specifically, frozen = True) which makes the class immutable is the decorator:

@dataclass(frozen=True)

1. As written in the python docs on the internet:

It is not possible to create truly immutable Python objects. However, by passing frozen=True to the [dataclass()](https://docs.python.org/3/library/dataclasses.html" \l "dataclasses.dataclass" \o "dataclasses.dataclass) decorator you can emulate immutability. In that case, dataclasses will add [\_\_setattr\_\_()](https://docs.python.org/3/reference/datamodel.html#object.__setattr__) and [\_\_delattr\_\_()](https://docs.python.org/3/reference/datamodel.html#object.__delattr__) methods to the class. These methods will raise a [FrozenInstanceError](https://docs.python.org/3/library/dataclasses.html" \l "dataclasses.FrozenInstanceError" \o "dataclasses.FrozenInstanceError) when invoked.

Additional way to ensure the object is immutable is making its fields (all except of “nr\_matoshim\_on\_ambulance”) an object of type Frozenset. This ensures nothing can be added / deleted from those lists

1. If frozen parameter would be False, a user may have mistakenly copied the reference to the “state\_to\_expand”, (thinking he’s copying an object) and make changes to it. This will result in the “state\_to\_expand” changes. Example:

