Topics:
Classes, Inheritance, Polymorphism, etc.
[*100 points*]

---

1. Qazaq
[*30 points*]

Only the *Qazaq* class should have 3 methods inside it, all the others must have only a constructor (`init`). Complete each class correctly so that the following result is generated.

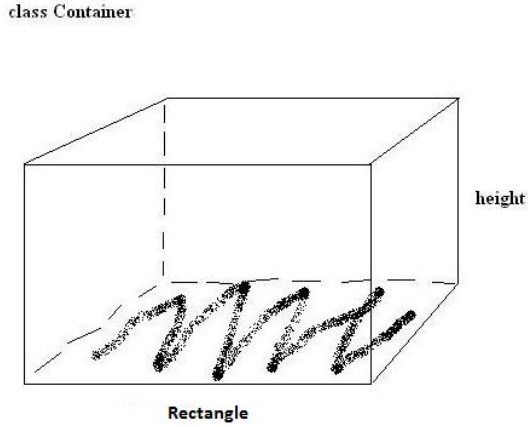
```
qazaq = Qazaq()
print(qazaq)
qazaq.ne_isteidi()

alban = UlyJuz()
qypwaq = OrtaJuz()
aday = KiwiJuz()

alban.ne_isteidi()
qypwaq.ne_isteidi()
aday.ne_isteidi()

halyq = [alban, qypwaq, aday]
for azamat in halyq:
    print(azamat)
```

Result:

```
>>>
Мен Қазақпын!
Бешбармақ жейді...
Ұлы жүзге таяқ беріп, малға қой
Орта жүзге қалам беріп, дауға қой
Кіші жүзге найза беріп, жауға қой
Мен Қазақпын!
Мен Қазақпын!
Мен Қазақпын!
>>>
```

---

2. Refrigerator
[*40 points*]



You should continue the `Shape` class discussed in class to do this task.

`Container` class

Contents:

`__init__`:
Gets 3 arguments:
- constructs a `Rectangle` from `width` and `length` (as attributes);
- and `height`.

volume:
- calculates the overall volume of a container (must use a method of `Rectangle`).

`Refrigerator` class

Derives from `Container` class.

Contents:

`__init__`:
Accepts 5 arguments:
- product model name;
- features, options;
- creates a `Container` from the first three arguments (3 sides).

`__str__`:
- prints as shown in the example.

The program should work as below:

```
c = Container(10,20,30)
print(c)
print(c.volume(),'\n')

r = Refrigerator(10,20,30,'Samsung F007','no frost, 3 year warranty')
print(r)
```

Result:

```
>>>
<__main__.Container object at 0x02A92270>
6000

Samsung F007, 6000 liters:
- no frost
- 3 year warranty
>>>
```

---

3. The Goodies

[*30 points*]

Here you will be asked to do the exercises that appear inside Chapter-19 of the textbook. Refer to the book to get more details and pre-requirements about the problem.

a) Python provides another built-in function, `all`, that returns `True` if every element of the sequence is `True`. As an exercise, use `all` to re-write `uses_all` from Section 9.3. (*page 186*);

b) The <= operator checks whether one set is a subset or another, including the possibility that they are equal, which is true if all the letters in word appear in available. As an exercise, rewrite `avoids` using sets. (*page 187*)

c) My solution to Exercise 18.3, which you can download from http://thinkpython2.com/code/PokerHandSoln.py, uses `setdefault` in the function `has_straightflush`. This solution has the drawback of creating a `Hand` object every time through the loop, whether it is needed or not. As an exercise, rewrite it using a `defaultdict`. (*page 189*)

==========END.