"Pet Caretaker"

In this assignment you will create a menu driven program that lets the user care for a virtual pet. The player is responsible for keeping the pet happy. The player can feed and play with the pet to keep it in a good mood. They can also listen to the pet to find out how it is feeling, which can range from happy to mad. The player can ask the pet to do a trick. If the pet is happy it will perform, at random, one of three tricks that it knows, otherwise it will tell the player it doesn't feel like doing a trick.

Program mock up:

```
*************** Pet Caretaker ****************
*                                            *
* (L) – Listen to pet, check pet’s mood      *
* (F) – Feed pet                             *
* (P) – Play with pet                        *
* (T) – Ask pet to do a trick                *
* (Q) - Quit program                         *
*                                            *
**********************************************
```

Main Menu

```
************ Pet’s Mood ***********
I feel MAD!

Press any key to continue…
```

Mood menu Example

```
************ Trick ***********
I do a backflip

Press any key to continue…
```

Trick Example

```
************ Trick ***********
I don’t feel like it

Press any key to continue…
```

unhappy Trick Example

Requirements:

- Menu choices will work with both upper and lower case characters.
- You must present the menu as shown above.
- You must use object oriented programming techniques.
- The pet **has a** hunger level and a boredom level (**state**)
- Each time the player interacts with the pet, the pet's boredom and hunger levels increase by a small amount.

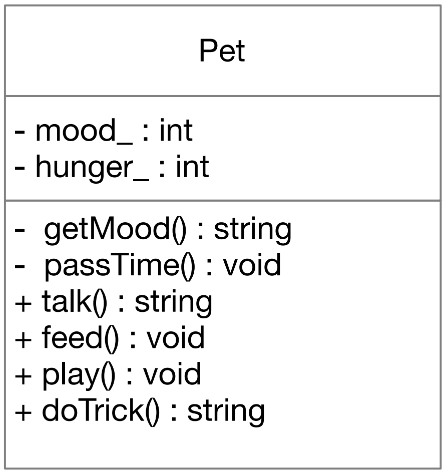
- Hunger and boredom levels start at 0 and go up. A pet is full and can eat no more if their hunger level is 0. A pet is as not bored as possible if their boredom level is 0.
    - Each time the player feeds the pet, pet's hunger level decreases (-3).
    - Each time the player plays with the pet, the pet's boredom level decreases (-3).
    - A pets hunger and/or boredom can never be less than 0.
- The pet will randomly do one of three tricks
    - "Roll over"
    - "Jump",
    - "backflip"
- If the pet is not Happy he/she will say "I don't feel like it" instead of doing a trick.

Create a Pet class that stores it's hunger and boredom levels as internal **state** (private). Create public methods for the
e pet to interact with the player:

- feed() – decreases hunger level (-3)
- play() – decreases boredom level (-3)
- talk() - the pet tells you his mood, one of Mad, Frustrated, OK, or Happy
    - "I'm your pet and I feel _______"
- doTrick() – the pet randomly does one of three tricks, if it is Happy.

You may create some private methods to use in implementation, for example:

- passTime(), adds to hunger and boredom levels. Is called by each of the public methods to simulate passing of time as the user plays the game.
- getMood(), returns a string, based on your pets mood
    - if (mood + hunger > MAD) return "mad"
    - if (mood + hunger > FRUSTRATED) return "frustrated"
    - if (mood + hunger > OKAY) return "okay"
    - otherwise return "happy"



- Your program should demonstrate good programing practices and follow the class style guide.
- Your pet class should never write directly to cout, it should return a string, and the main program (the controller) should write to cout