Quiz #7 **9/24**

Name (please print): _____

Student Number (recommended): _____

This quiz is all about nuclear physics.

DON'T PANIC!

All you need to know about the subject is contained in the questions themselves. Although you have a large amount of space to enter your answer, you may be surprised how few lines are actually required.

1. [50 points] Show the definition of a class called `Particle`. `Particle`s have a symbol, a charge, and a "mass number[1]", all of which are set at instantiation (passed in that order to the constructor) with no defaults and never change. Here are some example instantiations:

   ```
   em = Particle("e-", -1, 0)      # an electron
   ep = Particle("e+", 1, 0)       # a positron
   p = Particle("p", 1, 1)         # a proton
   n = Particle("n", 0, 1)         # a neutron
   nu_e = Particle("nu_e", 0, 0)   # a neutrino
   gamma = Particle("gamma", 0, 0) # a gamma particle
   ```

   When `Particle`s are passed to `print()`, the symbol is what you see, so

   ```
   print(em, ep, p, n, nu_e, gamma)
   ```

   results in

   ```
   e- e+ p n nu_e gamma
   ```

   (Note that the separating spaces are provided by `print()`.)

   When displayed interactively, the user gets a "lossless" output that resembles the original instantiation:

   ```
   >>> gamma
   Particle('gamma', 0, 0)
   ```

   (please enter your answer on the next page)

[1] A particle's mass number is the number of nucleons it contains and differs slightly from its actual mass, but you don't need to know that.

(this page left blank for your answer)

2. [50 points] Create a subclass of `Particle` called `Nucleus`. A `Nucleus` has the same attributes as a `Particle`, except that its symbol is that of a chemical element ("`H`" for hydrogen, "`He`" for helium, etc.[2]). Here are some declarations:

```
d =   Nucleus("H", 1, 2)  # hydrogen
li6 = Nucleus("Li", 3, 6) # lithium
he4 = Nucleus("He", 2, 4) # helium
```

When you print a `Nucleus`, you get the symbol preceded by its mass number in parentheses, so that:

```
print(d)
print(li6)
print(he4)
```

will produce

```
(2)H
(6)Li
(4)He
```

As with `Particles`, an interactive display is lossless:

```
>>> he4
Nucleus("He", 2, 4)
```

You may assume the existence of a working implementation of `Particle` and, for full credit, your answer must make as extensive use of inheritance from it as possible.

(please enter your answer on the next page)

[2]These will be provided by the user of your module. You do not need to check them for legitimacy.

(this page left blank for your answer)