The Gradebook Program

Imagine a gradebook that stores the names and grades of a student in a class. If we print the gradebook it comes out like this:

```
bob:          28   32   11
Hermione: 100  100  100
Hagrid        37   45   89
```

There are 3 things we want to do with the gradebook:

- GetNewClassList: gets a list of all of the students in the class and initializes whatever structures the gradebook uses.
- NewGrade: goes through the class roster and asks the instructor for a grade for each student.
- PrintGrades: prints the entire gradebook.

We will see this program written in 3 ways: with the gradebook stored as a list, as a dictionary, and in an object-oriented format.

The list version stores the gradebook as a list of pairs, where the first entry of each pair is a name and the second entry is a list of grades:

```
[  ("bob", [28,32,11]),
   ("Hermione", [100,100,100]),
   ("Hagrid", [37, 45,89])   ]
```

We can run through this list to print it as

```
for (name, gradeList) in Gradebook:
    ....
```

We can get a new grade with

```
for (name, gradeList) in Gradebook:
    g = eval(input( "grade for %s"%name "))
    gradelist.append(g)
```

The dictionary version stores the grades in a dictionary, where the keys are the names of the students and the value associated with a key is the list of grades for that student.

We can run through this structure to print it as

```
for name in Gradebook.keys():
        for grade in Gradebook[name]:
            ....
```

We can get a new grade with:

```
for name in Gradebook.keys():
    grade = eval(input("Grade for %s: ", %name))
    Gradebook[name].append( grade )
```

Now, what about the object-oriented design?

What are the physical elements corresponding to parts of a gradebook?

I can see three:

- A Gradebook contains a roster of Students
- A Student has a name and a list of Grades
- A Grade has a value, which might be a number or a string.