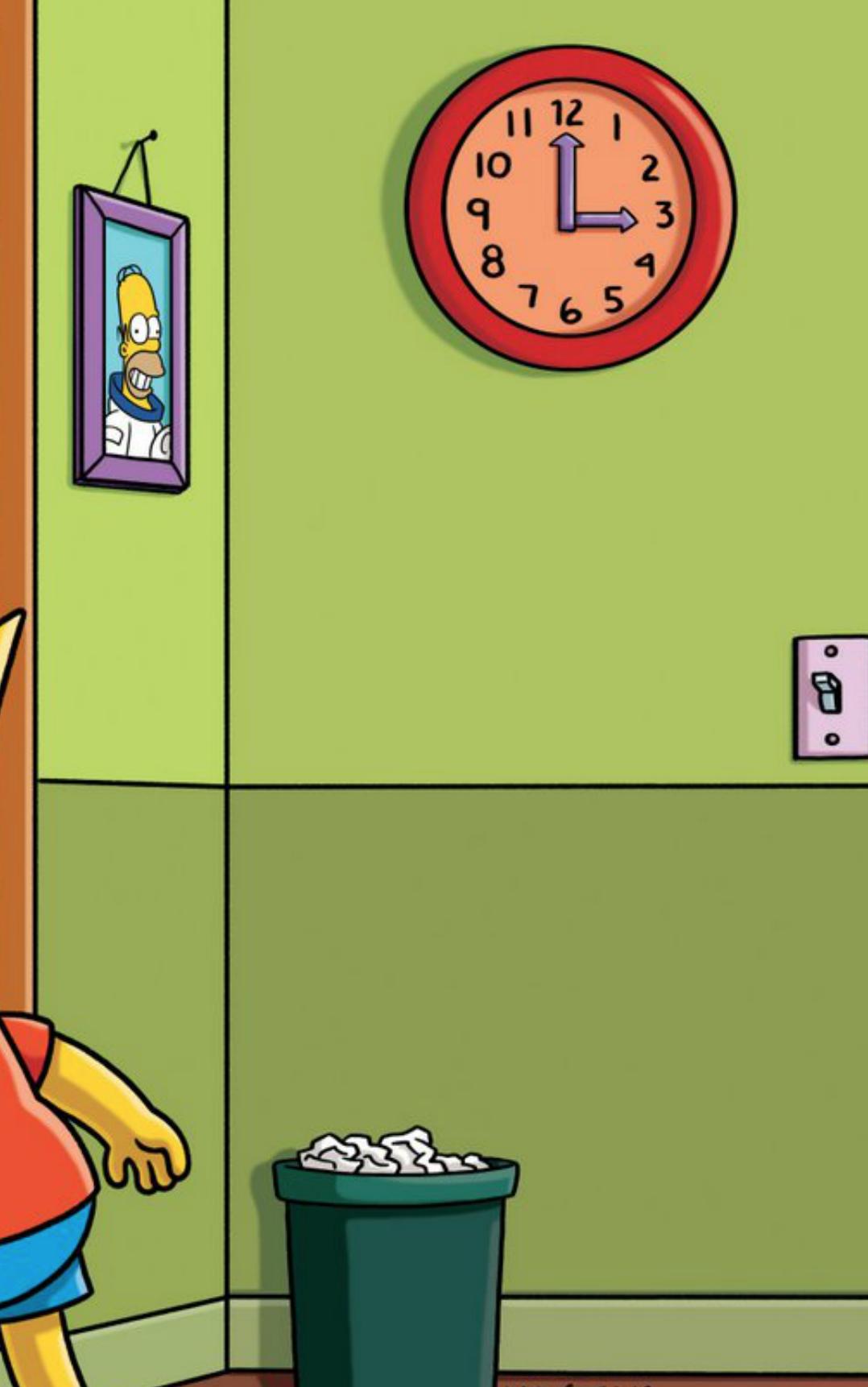




Exercism: Grade School

I will do my Exercism
I will do my Exercism



MATT GROENING

School Roster



School Roster

School Roster



Add a student's name to the roster for a grade

School Roster

- 🎒 *Add a student's name* to the roster for a grade
- 🎒 *Get a list of all students* enrolled in a grade

School Roster

- 🎒 *Add a student's name* to the roster for a grade
- 🎒 *Get a list of all students* enrolled in a grade
- 🎒 Get a *sorted list* of all students in *all grades*



Take 1

Create a School Roster

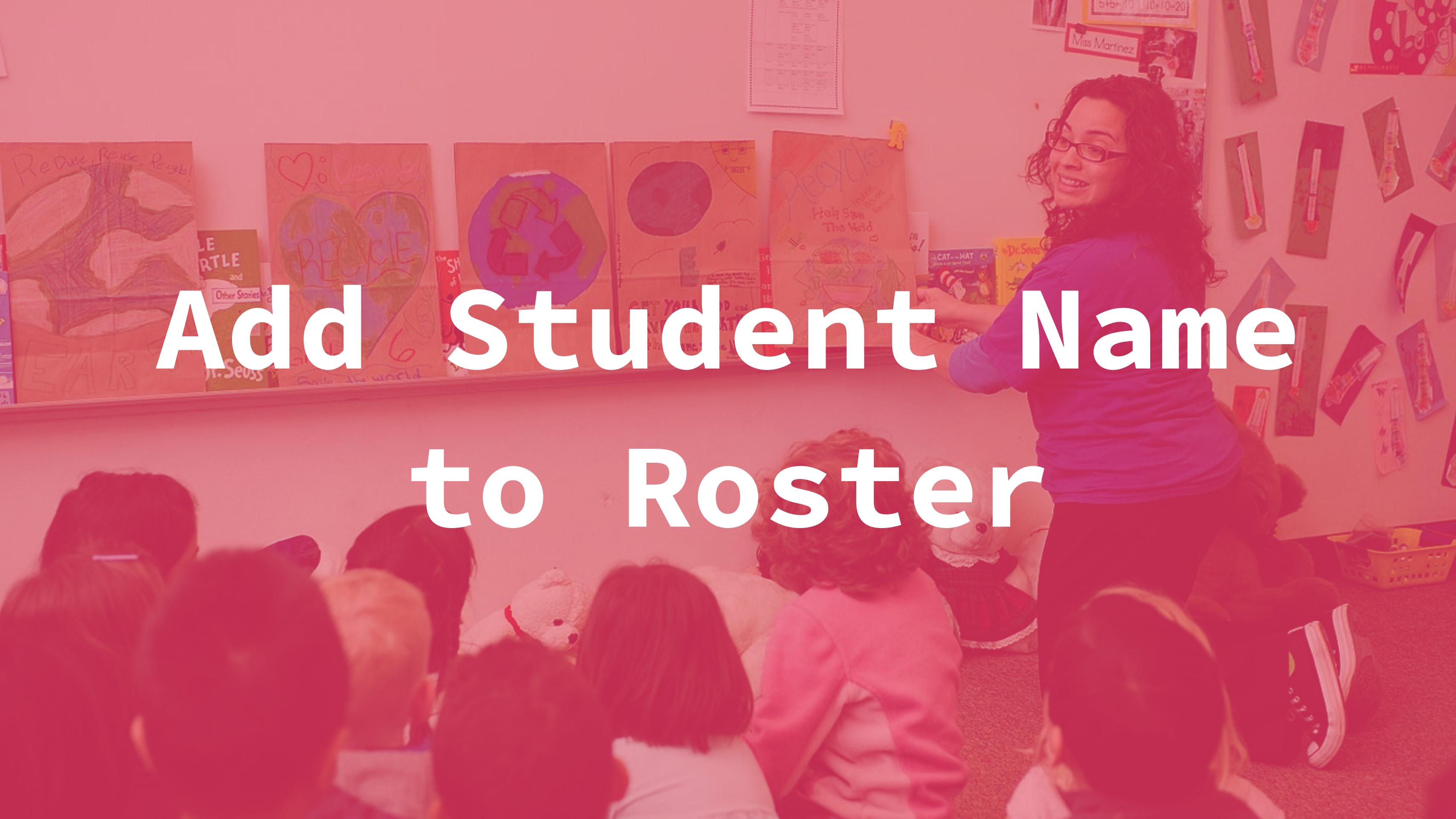
```
class School
  def initialize
    @roster = []
  end

  private

    attr_reader :roster
end
```

A Roster is filled with Enrollments

```
> @roster
=> [
    { grade: 1, students: ["Paul", "RobC"] },
    { grade: 2, students: ["Nick"] ) },
    { grade: 3, students: ["Cath", "RobH"] )
]
```

A photograph of a classroom scene. In the foreground, the backs of several young children are visible, looking towards the front of the room. In the background, a teacher with curly hair and glasses, wearing a purple shirt, stands smiling. The wall behind her is covered with various pieces of children's artwork, including a large recycling symbol, a heart with the word "Dessert", and several "Recycle" signs. There are also book titles like "LE P'TIT L'RTLÉ and Other Stories" and "Dr. Seuss".

Add Student Name
to Roster

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

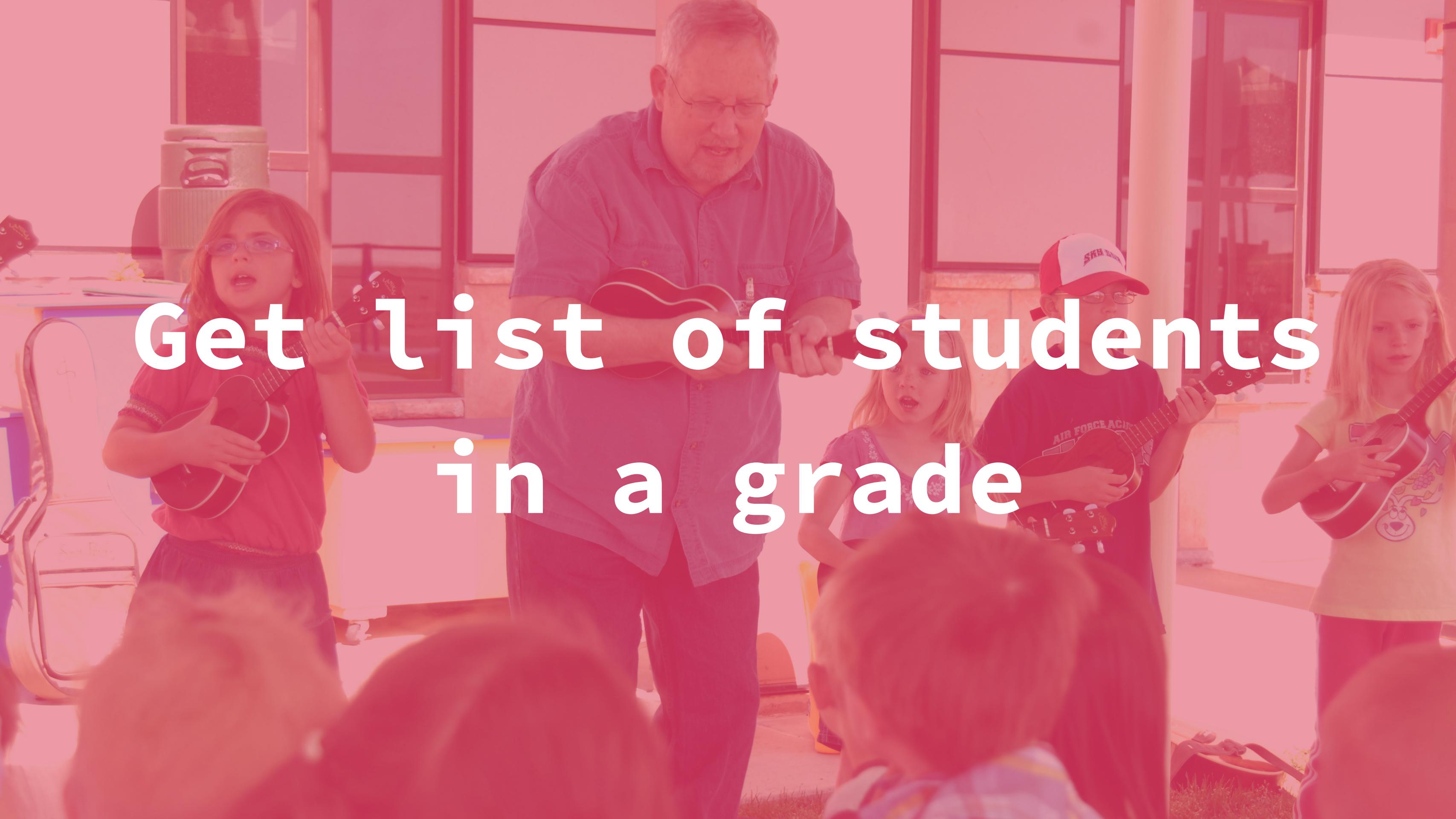
```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

A group of children are playing ukuleles in a classroom setting. A teacher, wearing glasses and a red shirt, is standing in front of them, looking down at one of the students. The students are all playing ukuleles and singing. One student in the foreground is wearing a red cap with "SKH" on it. Another student has "AIR FORCE AC" on their shirt. There are windows in the background.

Get list of students
in a grade

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

A classroom scene with many students of various ages. In the foreground, a young girl in an orange t-shirt with her arms raised is smiling broadly. Behind her, other students are cheering and clapping. The room has bookshelves and educational posters on the walls.

Get sorted list of
students in all grades

Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```

Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```

Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```

Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```

Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```

Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```



exercism submit grade_school.rb



Done !



Done?

what gives?

what gives?



Methods are kind of long...

what gives?

- 🎒 Methods are kind of long...
- 🎒 Is the naming right...?

what gives?

- 🎒 Methods are kind of long...
- 🎒 Is the naming right...?
- 🎒 There's code repetition...

what gives?

- 🎒 Methods are kind of long...
- 🎒 Is the naming right...?
- 🎒 There's code repetition...
- 🎒 Something doesn't *feel* right...



Take 2

Create a School Roster

```
class School
  def initialize
    @roster = []
  end

  private

    attr_reader :roster
end
```

A Roster is filled with Enrollments

```
> @roster
=> [
    { grade: 1, students: ["Paul", "RobC"] },
    { grade: 2, students: ["Nick"] ) },
    { grade: 3, students: ["Cath", "RobH"] )
]
```

Simplify to a Hash of Enrollments

```
> @enrollments
=> {
    1 => ["Paul", "RobC"],
    2 => ["Nick"],
    3 => ["Cath", "RobH"]
}
```

Create a Hash of Enrollments

```
class School
  def initialize
    @enrollments = {}
  end

  private

    attr_reader :enrollments
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

A group of students wearing purple t-shirts with "NCMS" and a logo are sitting at wooden desks in a classroom setting. They are holding clear plastic bags tied at the top. The student in the foreground is looking down at their bag. The student behind them is smiling. The student on the far right has their eyes closed. The background shows other students and classroom elements.

Too much
repetition

Create a Hash of Enrollments

```
class School
  def initialize
    @enrollments = {}
  end

  private

    attr_reader :enrollments
end
```

Create a Hash of Enrollments

```
class School
  def initialize
    @enrollments =
      Hash.new { |hash, key| hash[key] = [] }
  end

  private

    attr_reader :enrollments
end
```

Create a Hash of Enrollments

```
class School
  def initialize
    @enrollments =
      Hash.new { |hash, key| hash[key] = [] }
  end

  private

    attr_reader :enrollments
end
```

Add Student Name to Roster

```
def add(name, grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students] << name
  else
    roster << { grade: grade, students: [name] }
  end
end
```

Add Student Name to Enrollments

```
def add(name, grade)
  enrollments[grade] << name
end
```

Get list of students in a grade

```
def students(grade)
  grade_enrollment =
    roster.find { |enrollment| enrollment[:grade] == grade }
  if grade_enrollment
    grade_enrollment[:students].sort
  else
    []
  end
end
```

Get list of students in a grade

```
def students(grade)
    enrollments[grade].sort
end
```



Get sorted list of students in all grades

```
def students_by_grade
  roster
    .sort_by { |enrollment| enrollment[:grade] }
    .map do |enrollment|
      enrollment[:students] =
        enrollment[:students].sort
      enrollment
    end
  end
```

Get sorted list of students in all grades

```
def students_by_grade
  enrollments.sort.map do |grade, students|
    { grade: grade, students: students.sort }
  end
end
```

Get sorted list of students in all grades

```
def students_by_grade
  enrollments.sort.map do |grade, students|
    { grade: grade, students: students.sort }
  end
end
```

Get sorted list of students in all grades

```
def students_by_grade
  enrollments.sort.map do |grade, students|
    { grade: grade, students: students.sort }
  end
end
```

Get sorted list of students in all grades

```
def students_by_grade
  enrollments.sort.map do |grade, students|
    { grade: grade, students: students.sort }
  end
end
```

Get sorted list of students in all grades

```
> school.students_by_grade  
=> [  
      { grade: 1, students: ["Paul", "RobC"] } ,  
      { grade: 2, students: ["Nick"] ) } ,  
      { grade: 3, students: ["Cath", "RobH"] ) }  
]
```

You'll rarely get it
right the first time

Submit often,
get feedback

Read other
people's code

Refactor until
you're happy



github.com/paulfioravanti/exercism

Thanks!
@paulfioravanti

