In [ ]:
```
# Initialize Otter
import otter
grader = otter.Notebook("demo.ipynb")
```

Otter-Grader Tutorial

This notebook is part of the Otter-Grader tutorial. For more information about Otter, see our documentation (https://otter-grader.rtfd.io).

In [1]:
```
import pandas as pd
import numpy as np
%matplotlib inline
```

Question 1: Write a function `square` that returns the square of its argument.

In [1]:
```
def square(x):
    return x**2 # SOLUTION
```

In [ ]:
```
grader.check("q1")
```

Question 2: Write an infinite generator of the Fibonacci sequence `fiberator` that is *not* recursive.

In [8]:
```
def fiberator():
    # BEGIN SOLUTION
    yield 0
    yield 1
    x, y = 0, 1
    while True:
        x, y = y, x + y
        yield y
    # END SOLUTION
```

In [ ]:
```
grader.check("q2")
```

Question 3: Create a DataFrame mirroring the table below and assign this to `data`. Then group by the `flavor` column and find the mean price for each flavor; assign this **series** to `price_by_flavor`.

flavor	scoops	price
chocolate	1	2
vanilla	1	1.5
chocolate	2	3
strawberry	1	2
strawberry	3	4
vanilla	2	2
mint	1	4
mint	2	5
chocolate	3	5

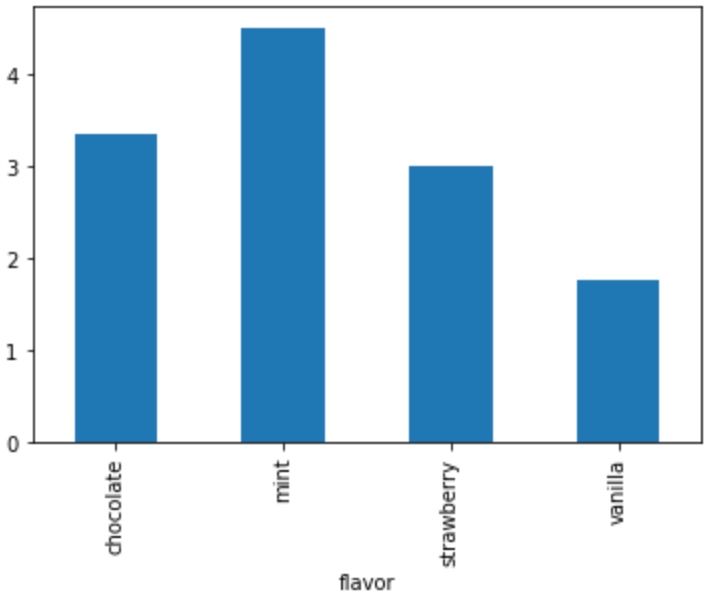
In [13]:
```
# BEGIN SOLUTION NO PROMPT
data = pd.DataFrame({
    "flavor": ["chocolate", "vanilla", "chocolate", "strawberry", "strawberry",
"vanilla", "mint",
               "mint", "chocolate"],
    "scoops": [1, 1, 2, 1, 3, 2, 1, 2, 3],
    "price": [2, 1.5, 3, 2, 4, 2, 4, 5, 5]
})
price_by_flavor = data.groupby("flavor").mean()["price"]
# END SOLUTION
""" # BEGIN PROMPT
data = ...
price_by_flavor = ...
""" # END PROMPT
price_by_flavor
```

Out[13]:
```
flavor
chocolate     3.333333
mint          4.500000
strawberry    3.000000
vanilla       1.750000
Name: price, dtype: float64
```

In [ ]:
```
grader.check("q3")
```

Question 4: Create a barplot of `price_by_flavor`.

In [26]:
```
price_by_flavor.plot.bar(); # SOLUTION
```



Question 5: What do you notice about the bar plot?

Type your answer here, replacing this text.

SOLUTION: mint is the highest...?

Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

These are some submission instructions.

In [ ]:
```
# Save your notebook first, then run this cell to export your submission.
grader.export(run_tests=True)
```