DATA 8

Week 4 – **Histograms, Functions**

Slides by Suraj Rampure

Fall 2017

Administrative

1. Project 1 comes out Friday!

   Start looking for partners in this lab.

2. Feedback!

   I’d really appreciate you giving me feedback on my teaching. Please fill out this form:

   https://goo.gl/forms/YzzThyyzplmDRUKp1

   sometime during lab if you haven’t already.

Histograms

An important, tricky topic! You won’t get much practice with these in lab, so make sure to pay attention and read the textbook.

What is a histogram?

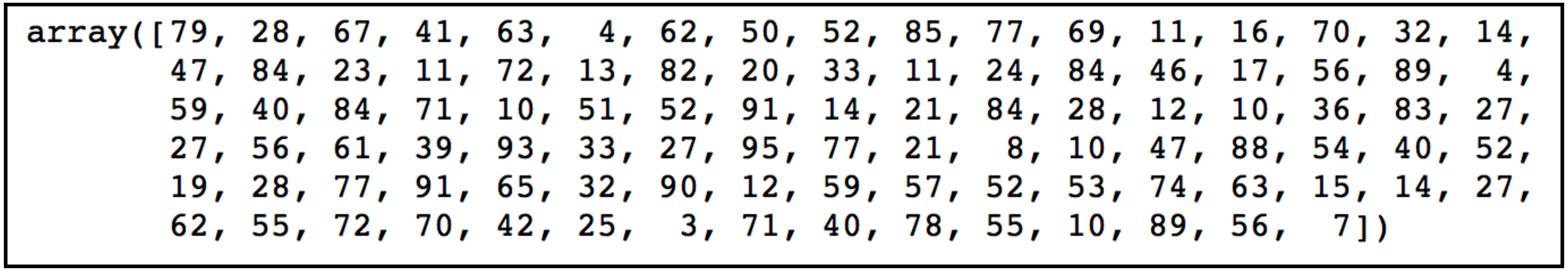
A **histogram** is a visualization that uses rectangles to show the frequency of data points. In a histogram:

- The **widths** of each rectangle are the same, and correspond to the **intervals** to which the data points belong
- The **areas** of each rectangle correspond to the **proportion** of the data that is made up by values in that interval

A **histogram** is a visualization that uses rectangles to show the frequency of data points. In a histogram:

- The **widths** of each rectangle are the same, and correspond to the **intervals** to which the data points belong
- The **areas** of each rectangle correspond to the **proportion** of the data that is made up by values in that interval

Here’s an array of some student grades.



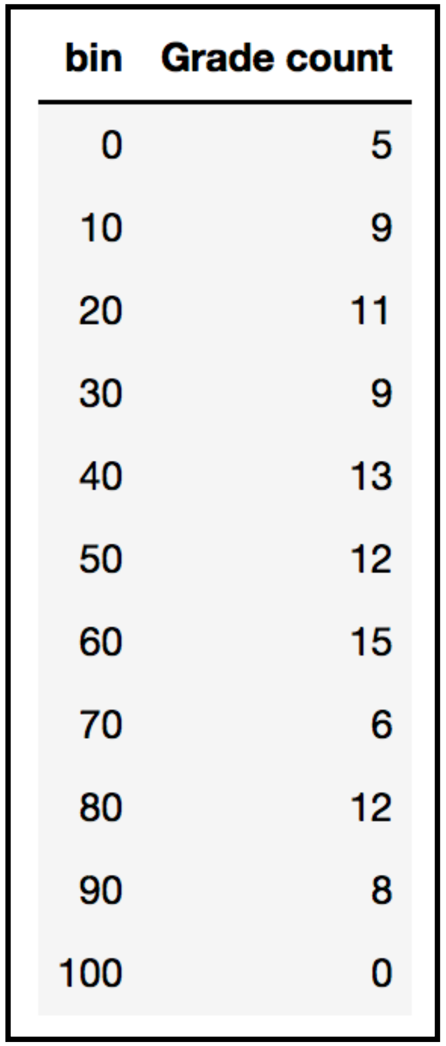
```
array([79, 28, 67, 41, 63,  4, 62, 50, 52, 85, 77, 69, 11, 16, 70, 32, 14,
       47, 84, 23, 11, 72, 13, 82, 20, 33, 11, 24, 84, 46, 17, 56, 89,  4,
       59, 40, 84, 71, 10, 51, 52, 91, 14, 21, 84, 28, 12, 10, 36, 83, 27,
       27, 56, 61, 39, 93, 33, 27, 95, 77, 21,  8, 10, 47, 88, 54, 40, 52,
       19, 28, 77, 91, 65, 32, 90, 12, 59, 57, 52, 53, 74, 63, 15, 14, 27,
       62, 55, 72, 70, 42, 25,  3, 71, 40, 78, 55, 10, 89, 56,  7])
```

As you’ll later see, to make a histogram in iPython, we need our values to be in a table. Ignore that for now.

A **histogram** is a visualization that uses rectangles to show the frequency of data points. In a histogram:

- The **widths** of each rectangle are the same, and correspond to the **intervals** to which the data points belong

- The **areas** of each rectangle correspond to the **proportion** of the data that is made up by values in that interval



bin	Grade count
0	5
10	9
20	11
30	9
40	13
50	12
60	15
70	6
80	12
90	8
100	0

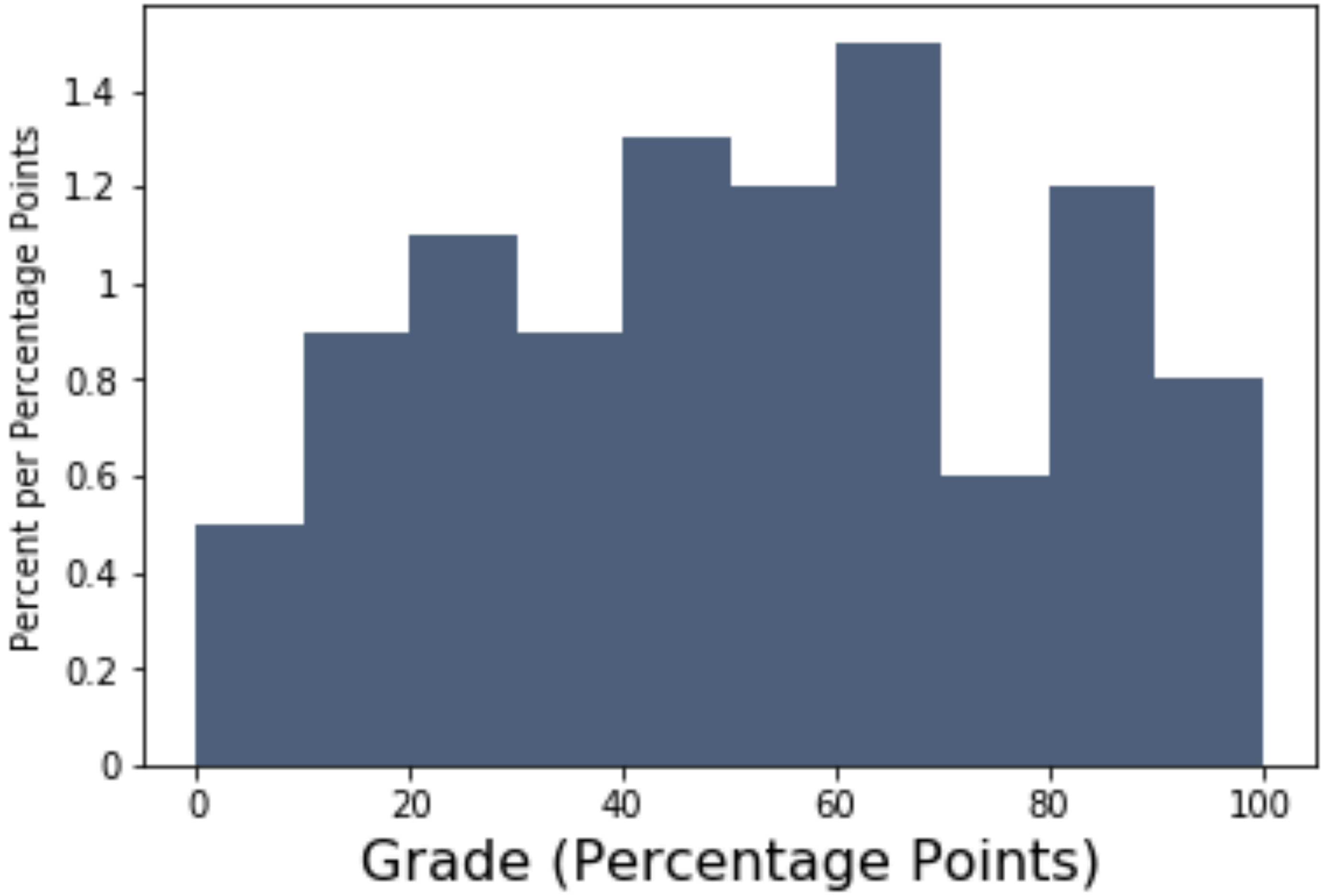
Here, we've "binned" the original data. We now don't know exactly what the original values were, we only know the **intervals** they lie in.

```
bins = np.arange(0, 110, 10)
```

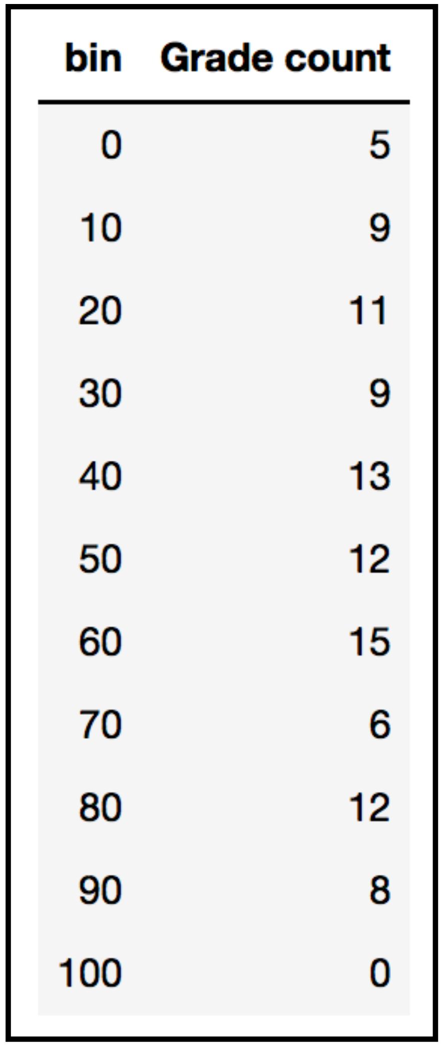
```
0-10, 10-20, 20-30, … 90-100
```

A **histogram** is a visualization that uses rectangles to show the frequency of data points. In a histogram:

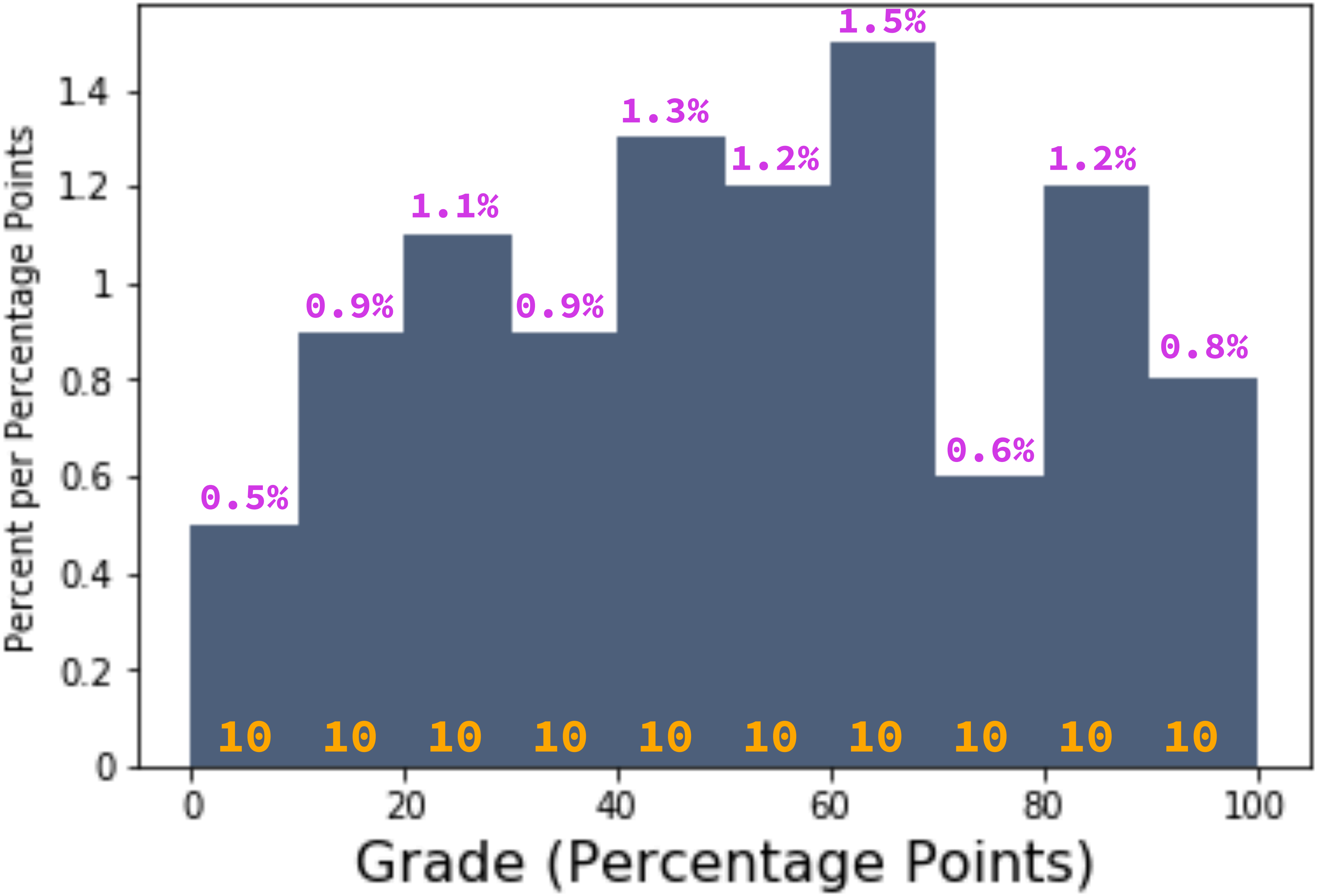
- The **widths** of each rectangle are the same, and correspond to the **intervals** to which the data points belong
- The **areas** of each rectangle correspond to the **proportion** of the data that is made up by values in that interval

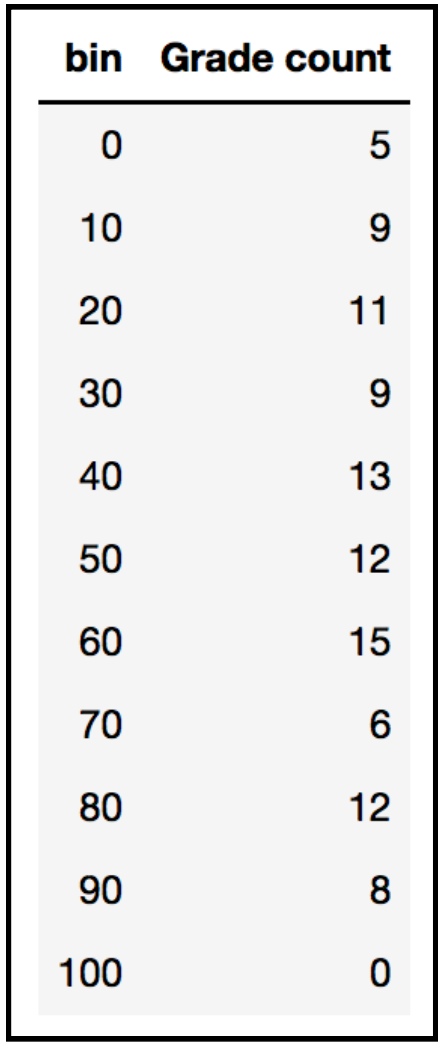


When defining bins, the left endpoint is included in each bin, but the right endpoint is not. For example, the counts for the value **`10`** are in the second bin, even the first bin was defined as **`0-10`**.



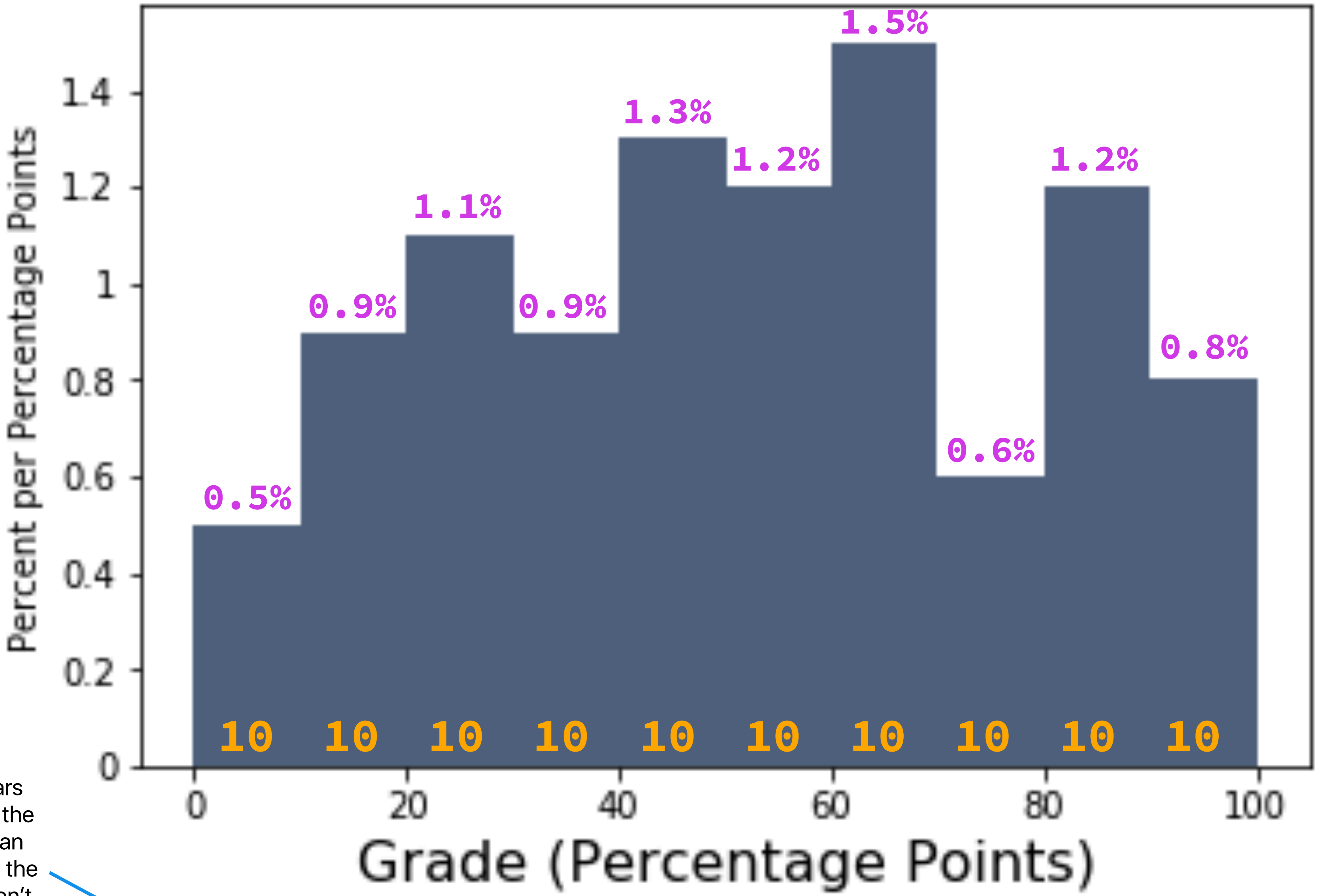
bin	Grade count
0	5
10	9
20	11
30	9
40	13
50	12
60	15
70	6
80	12
90	8
100	0





bin	Grade count
0	5
10	9
20	11
30	9
40	13
50	12
60	15
70	6
80	12
90	8
100	0

Simple, but powerful, formula: **Area = (Height of Bar) * (Width of Bar)**



Since all of the bars happened to have the same width, we can common factor out the width (10). This won't always happen!

Total Area = 10 * (0.5% + 0.9% + 1.1% + 0.9% + 1.3% + 1.2% + 1.5% + 0.6% + 1.2% + 0.8%) = 10 * (10%) = 100% = 1

The **sum** of the areas of the bars in a histogram is always **1** (100%).

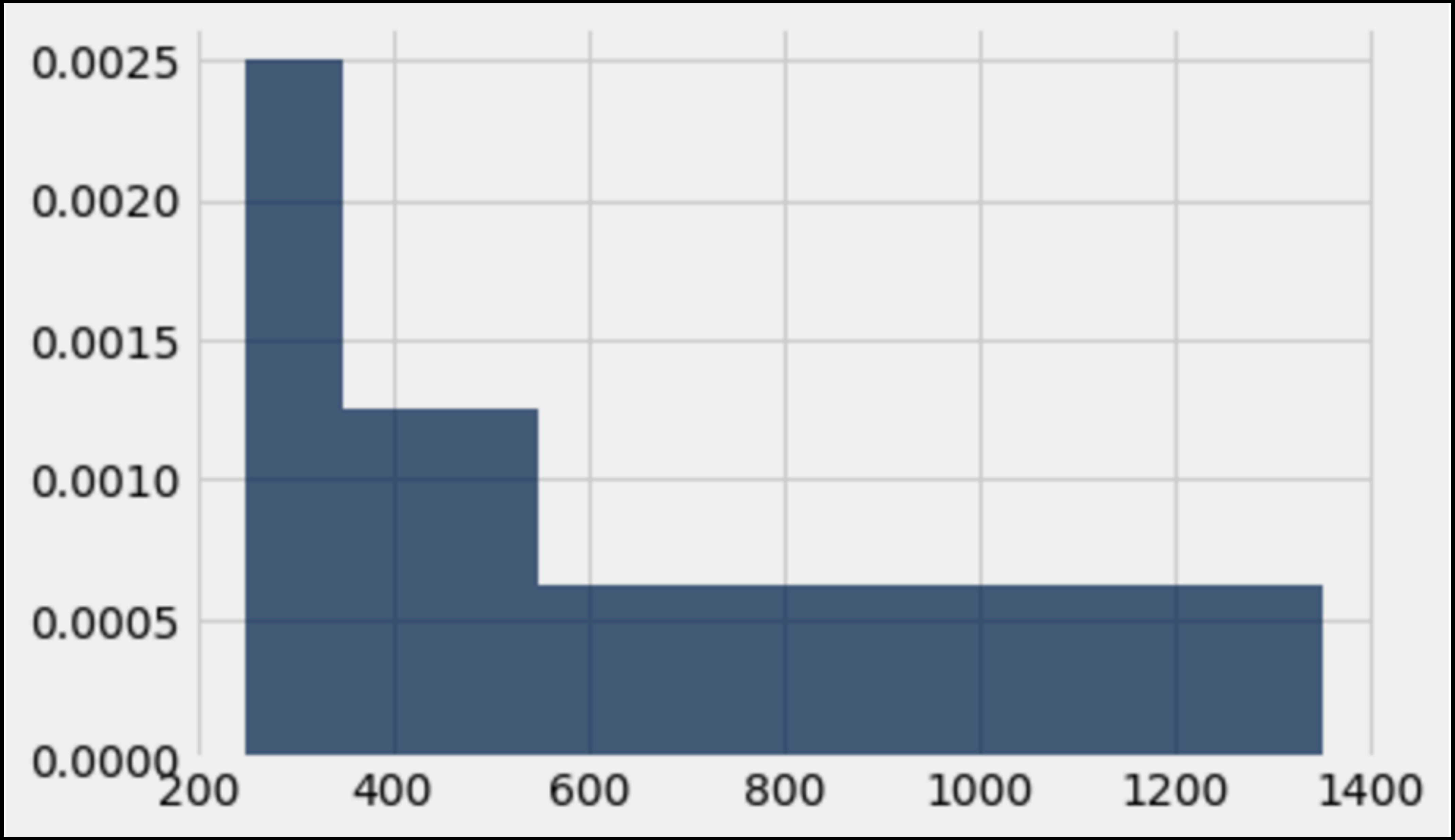
This is the most important fact about histograms there is.
Don’t forget it!

Discussion Worksheet Q1

Dollars	Student (%)
250-350	25
350-550	25
550-950	25
950-1350	25

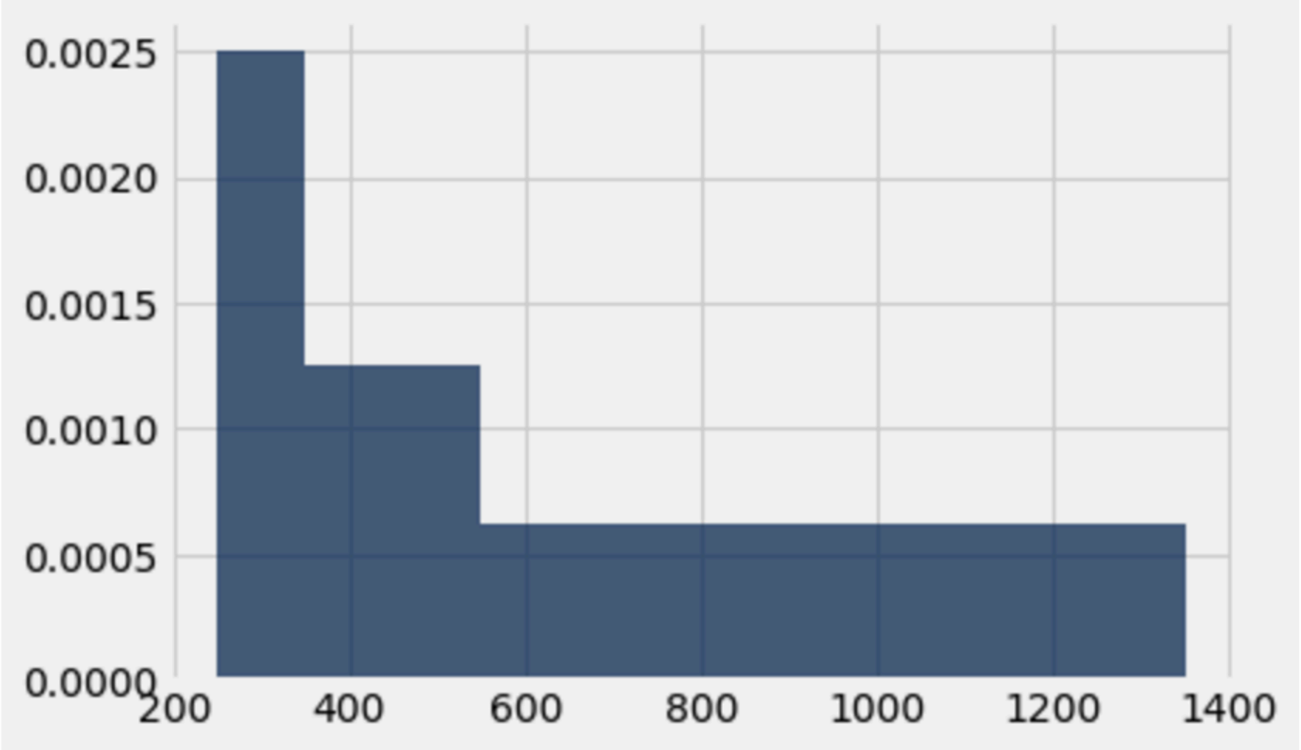
Draw a histogram of the above data.

Discussion Worksheet Q1



Discussion Worksheet Q1

Dollars	Student (%)
250-350	25
350-550	25
550-950	25
950-1350	25

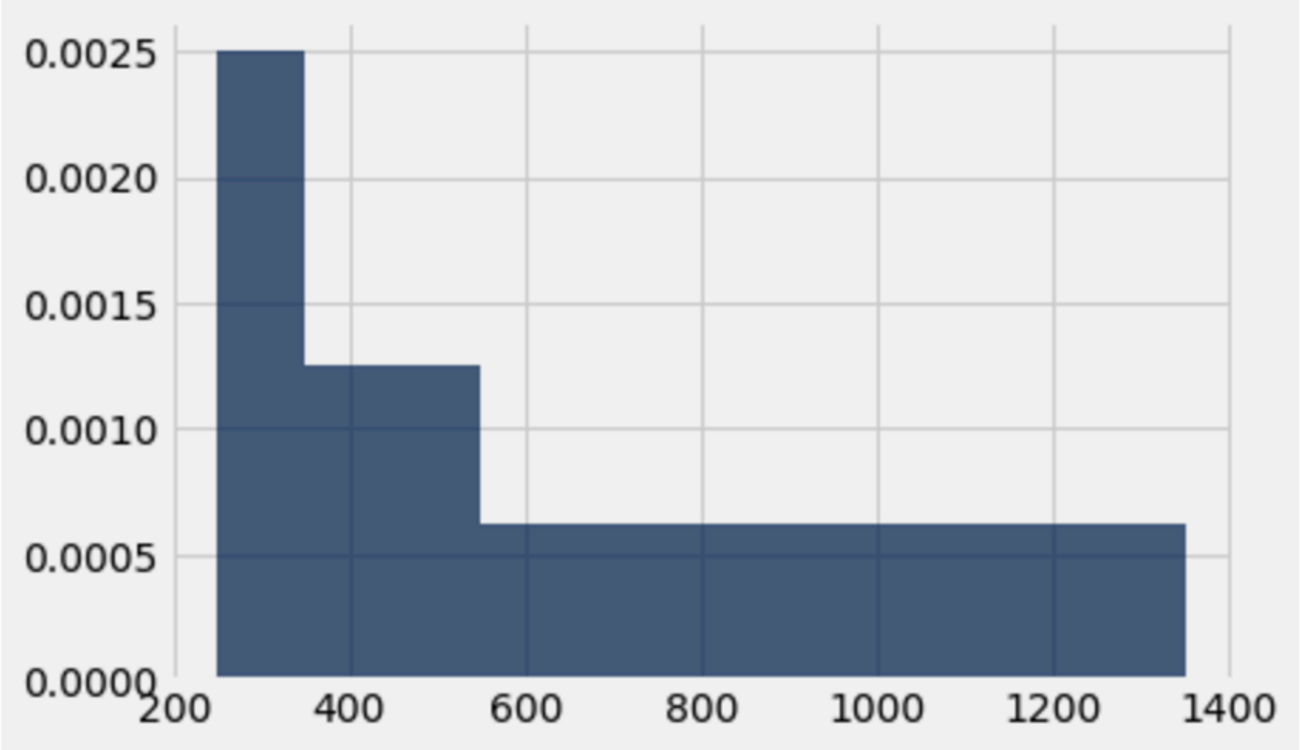


True or false (explain): The data shows that the rents are evenly distributed over the interval 250-1350.

False – Each bin contains 25% of the rents, but the bins aren't all of equal width.

Discussion Worksheet Q1

Dollars	Student (%)
250-350	25
350-550	25
550-950	25
950-1350	25



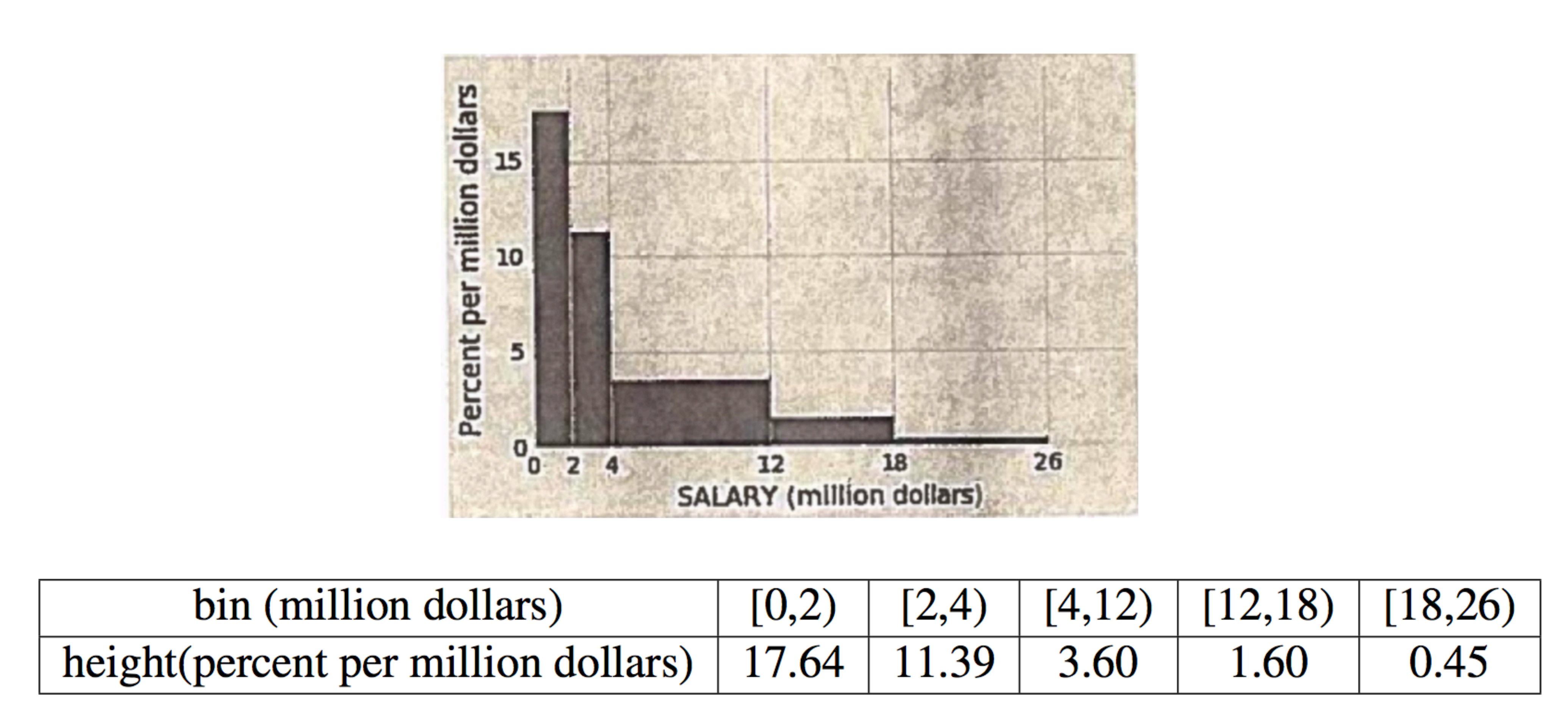
What is the height and correct units of the histogram bar over the bin 350-550 on the density scale?

0.125% per dollar

Since **Area = Height * Width,**

Height = Area/Width = 25%/(550-350) Dollars = 0.00125 per dollar = 0.125% per dollar

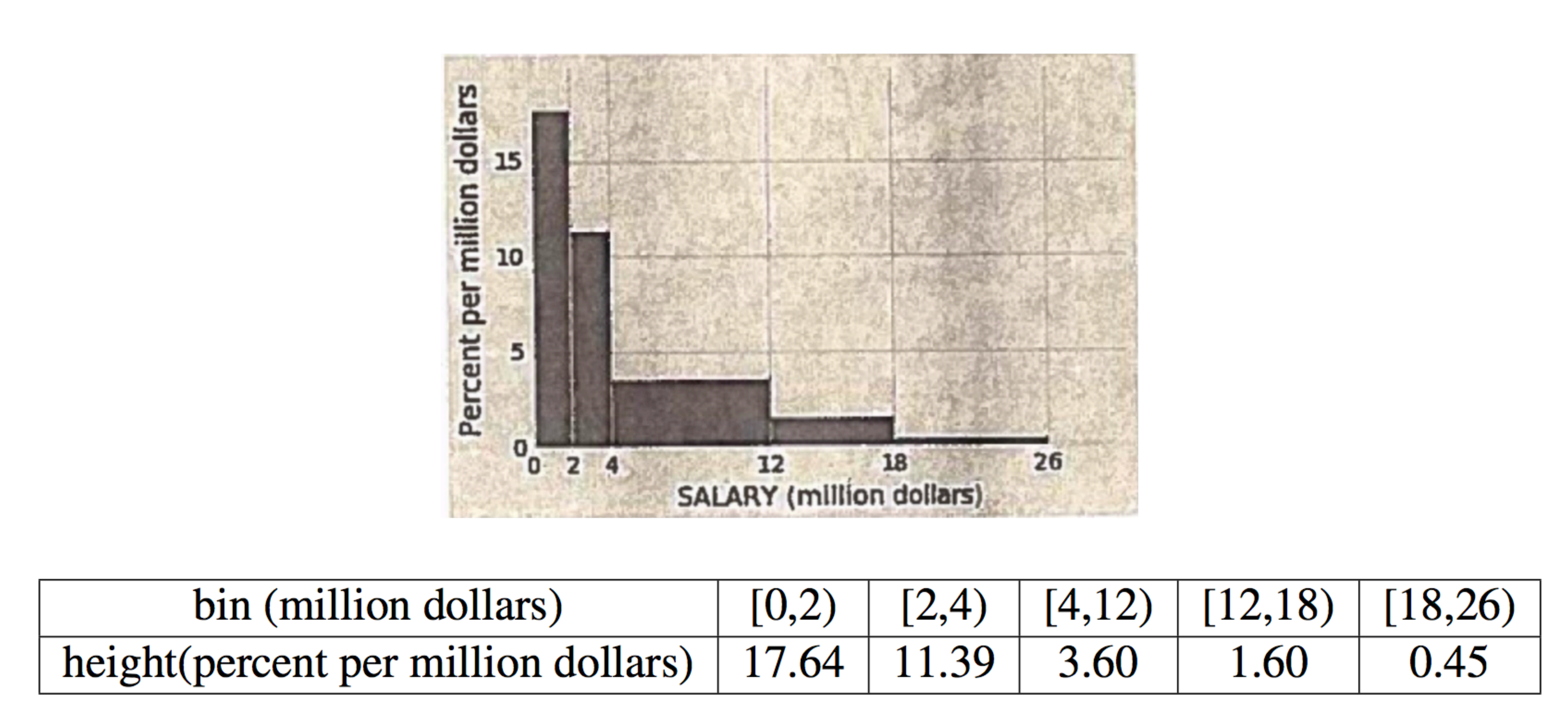
Discussion Worksheet Q2



bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

Which bin contains more players: [2, 4) or [4, 12)?

Discussion Worksheet Q2

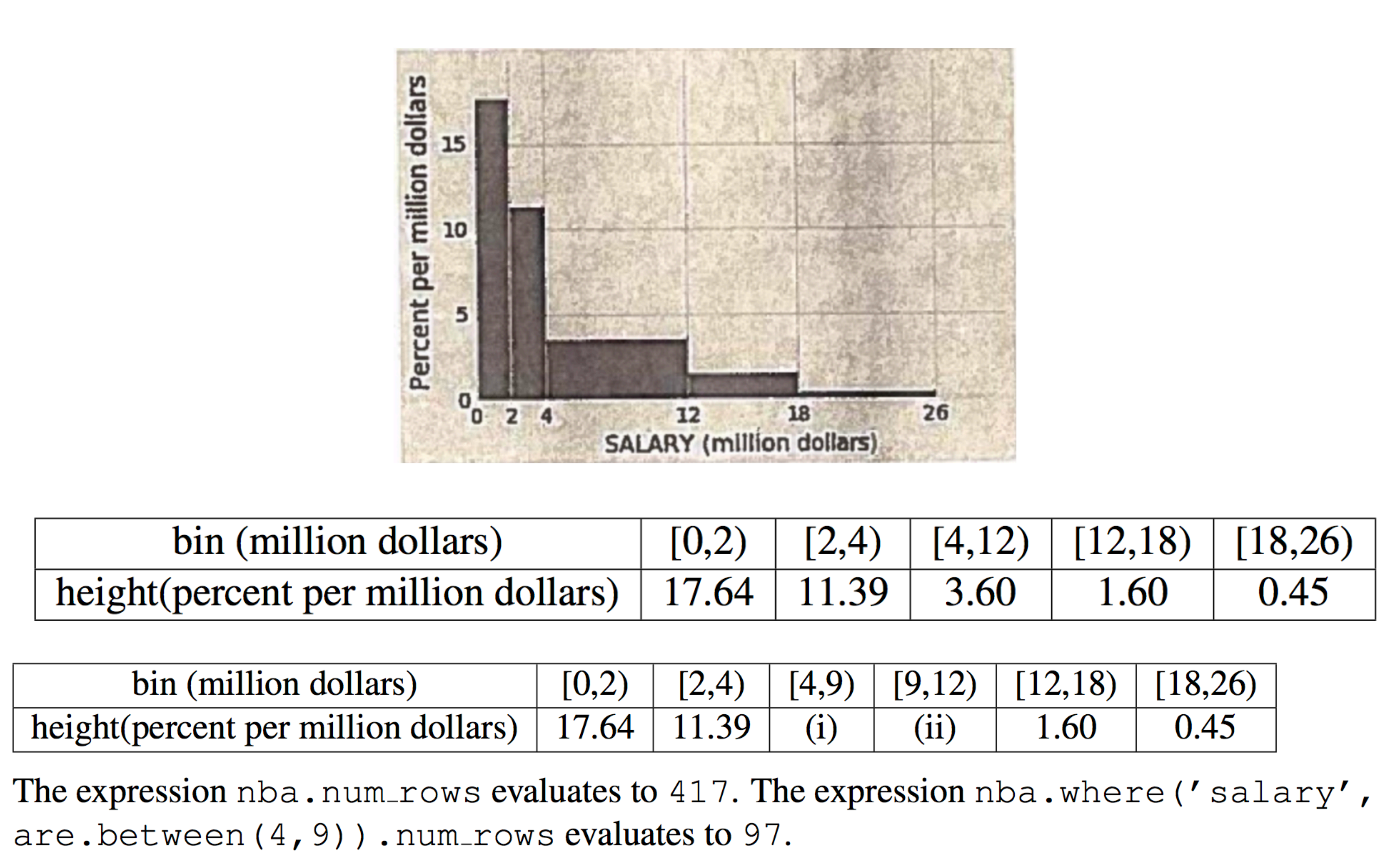


bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

Which bin contains more players: [2, 4) or [4, 12)?

[4, 12) because (2 * 17.64) < (8 * 3.60)
Remember, areas correspond to proportions. Larger area —> more values.

Discussion Worksheet Q2

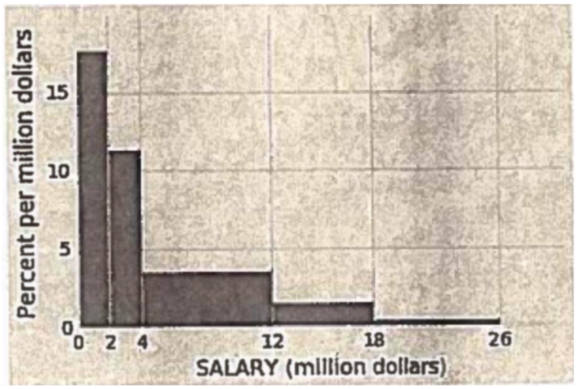

Percent per million dollars
0
5
10
15
0
2
4
12
18
26
SALARY (million dollars)

bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

bin (million dollars)	[0,2)	[2,4)	[4,9)	[9,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	(i)	(ii)	1.60	0.45

The expression `nba.num_rows` evaluates to `417`. The expression `nba.where('salary', are.between(4,9)).num_rows` evaluates to `97`.

Find the missing heights.

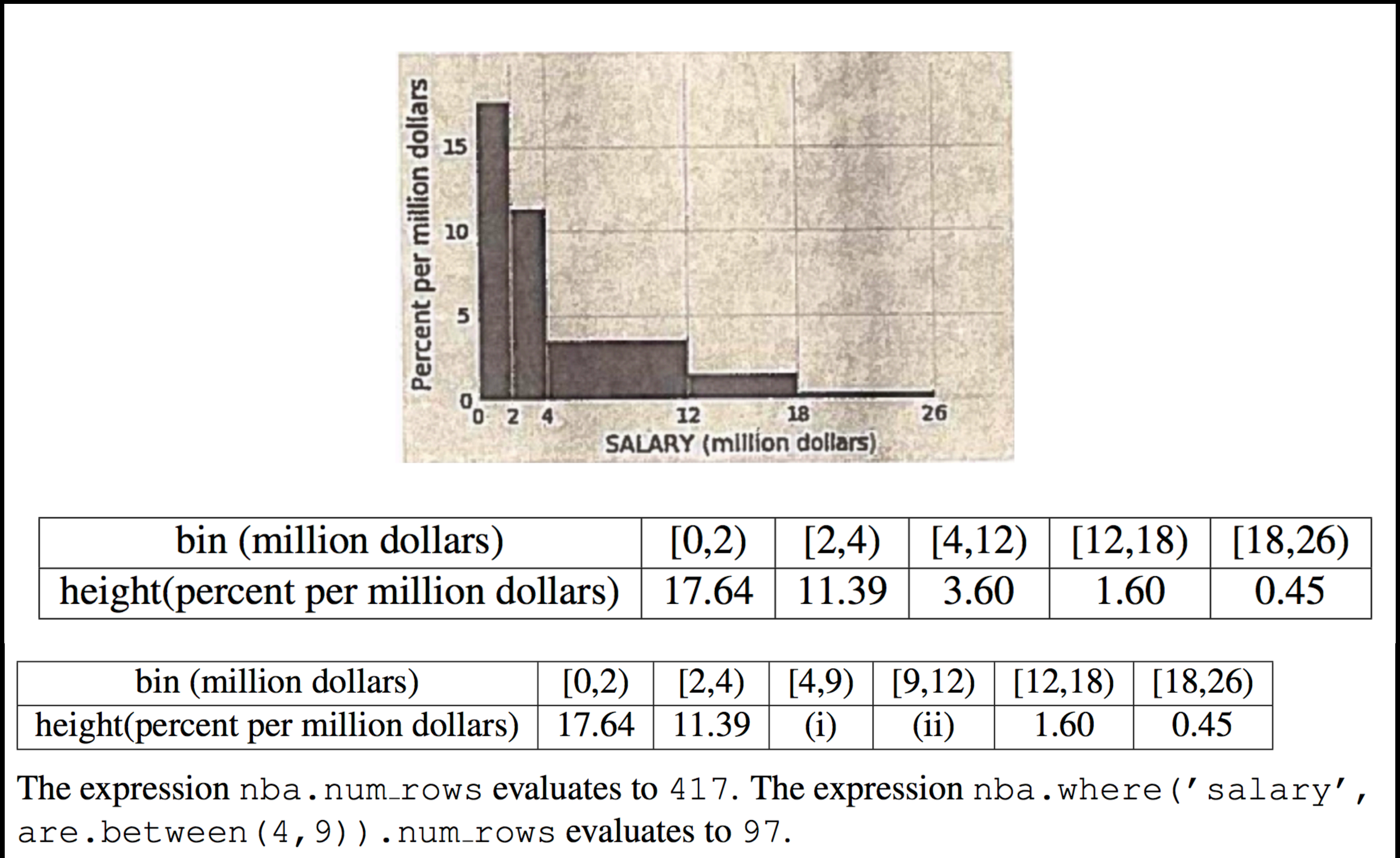


bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

bin (million dollars)	[0,2)	[2,4)	[4,9)	[9,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	(i)	(ii)	1.60	0.45

The expression `nba.num_rows` evaluates to `417`. The expression `nba.where('salary', are.between(4,9)).num_rows` evaluates to `97`.

Discussion Worksheet Q2



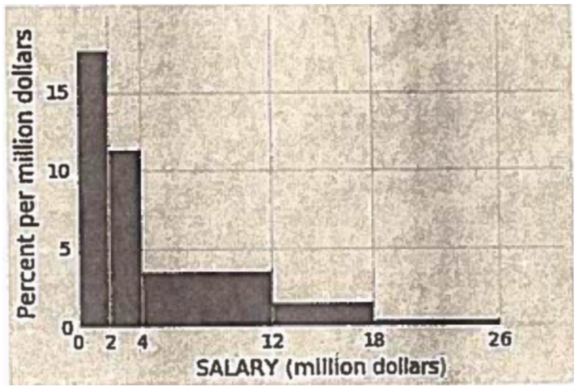
bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

bin (million dollars)	[0,2)	[2,4)	[4,9)	[9,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	(i)	(ii)	1.60	0.45

The expression `nba.num_rows` evaluates to `417`. The expression `nba.where('salary', are.between(4,9)).num_rows` evaluates to `97`.

i. 97 * 100 / (417 * 5)

Re-arranging the area formula for height, we have that **height = area/width**. We are given that 97 out of 417 values are in the 4-9 interval, and since areas correspond to proportions, we know the area of this bar is 97/417. The width of this bar is 9 - 4 = 5. We multiply by 100 since heights in histograms are measured in percentages.

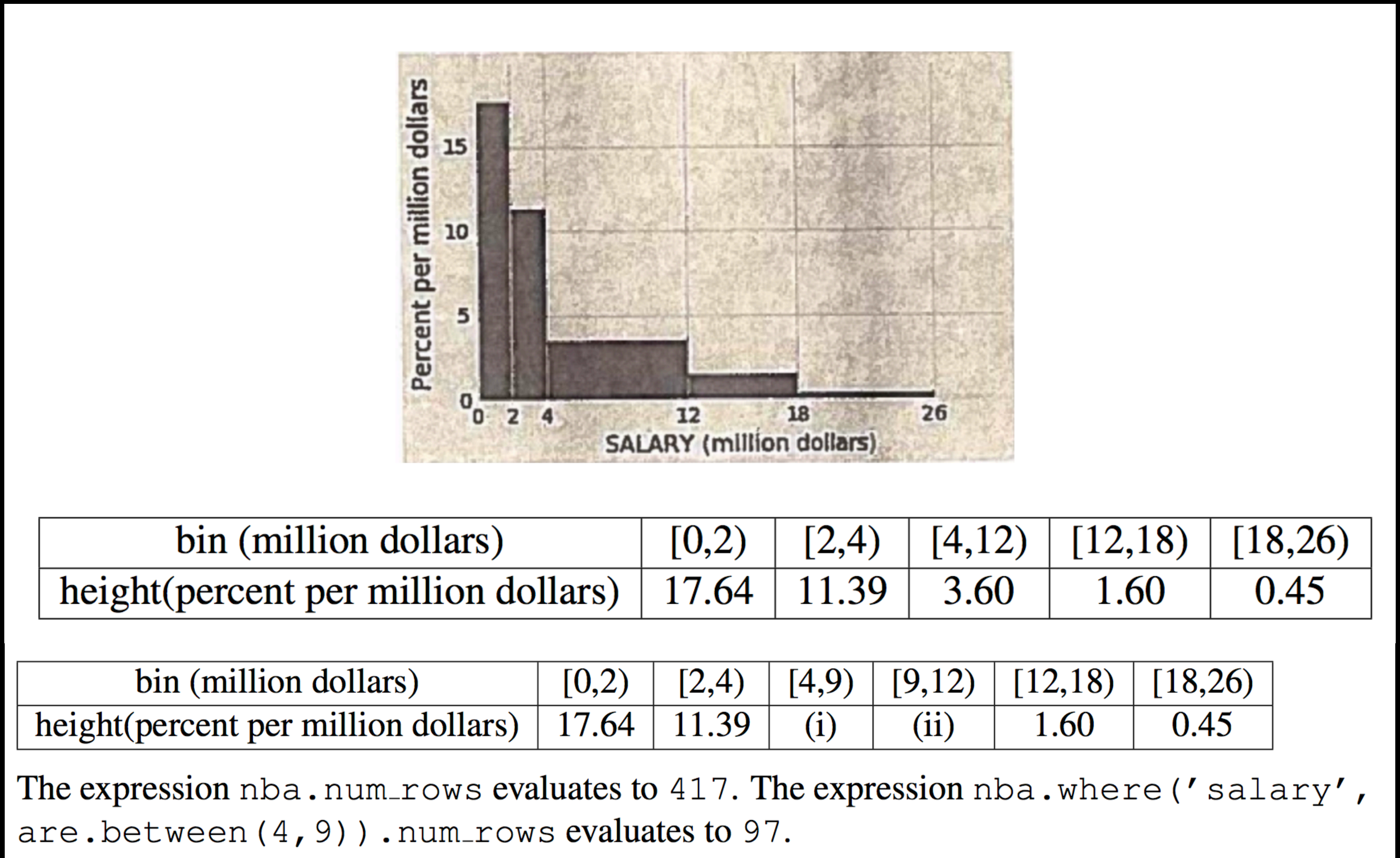


bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

bin (million dollars)	[0,2)	[2,4)	[4,9)	[9,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	(i)	(ii)	1.60	0.45

The expression `nba.num_rows` evaluates to `417`. The expression `nba.where(’salary’, are.between(4,9)).num_rows` evaluates to `97`.

Discussion Worksheet Q2

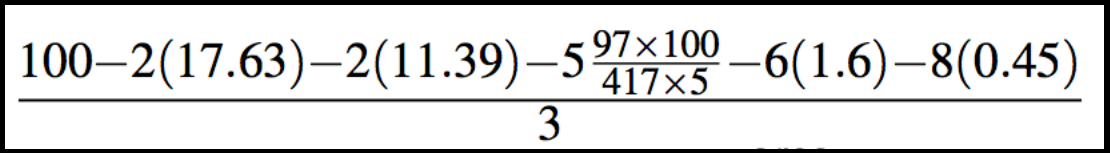


bin (million dollars)	[0,2)	[2,4)	[4,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	3.60	1.60	0.45

bin (million dollars)	[0,2)	[2,4)	[4,9)	[9,12)	[12,18)	[18,26)
height(percent per million dollars)	17.64	11.39	(i)	(ii)	1.60	0.45

The expression `nba.num_rows` evaluates to `417`. The expression `nba.where('salary', are.between(4,9)).num_rows` evaluates to `97`.

ii.



$$\frac{100-2(17.63)-2(11.39)-5\frac{97\times100}{417\times5}-6(1.6)-8(0.45)}{3}$$

We know the total area must sum to 100%. We can then subtract the areas of all other bars from 100% to find the area of this bar. We then divide this quantity by the width of this bar to find its height.

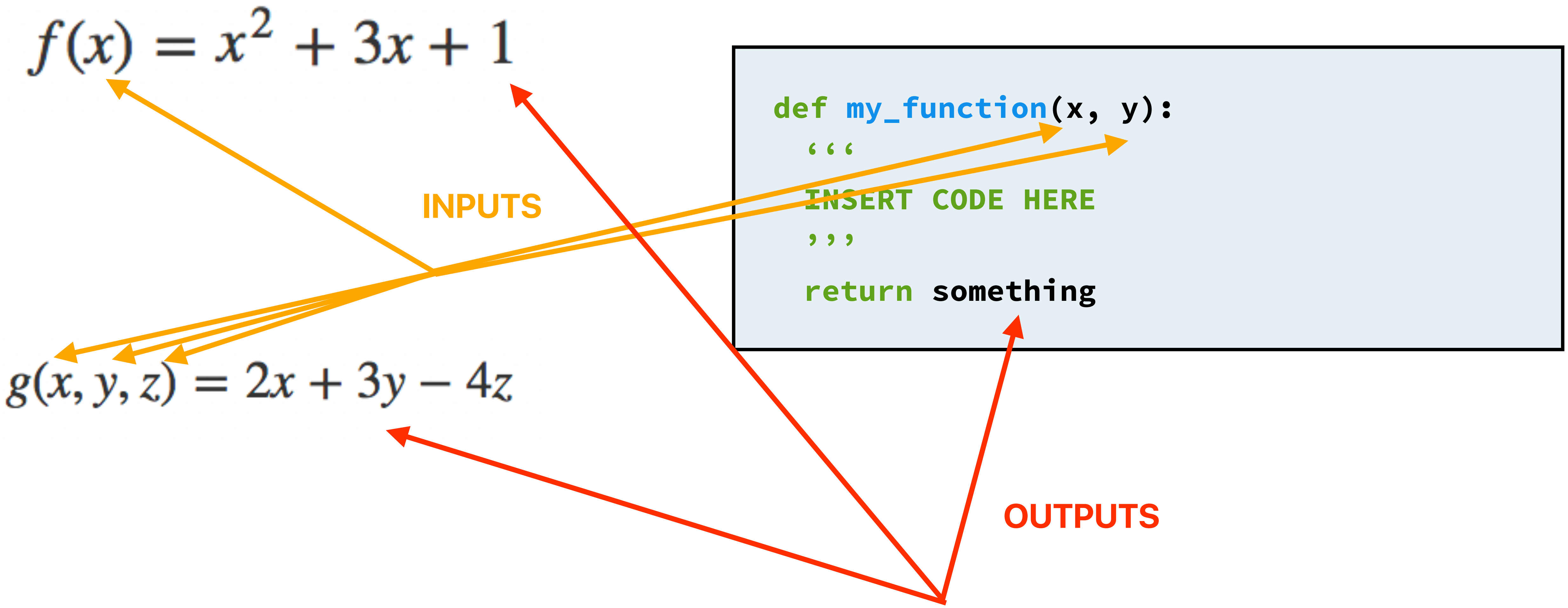
Functions

Functions

Functions give us a way to write code once and use it many times. Functions in programming work similarly to the way they do in mathematics – you provide the input(s), and the function determines the output(s).

Math

Programming



$$f(x) = x^2 + 3x + 1$$

$$g(x, y, z) = 2x + 3y - 4z$$

```
def my_function(x, y):
  ‘‘‘
  INSERT CODE HERE
  ’’’
  return something
```

INPUTS

OUTPUTS

Functions

Defining a function:

```
def compound_interest(PV, rate, t):
  ‘‘‘Calculates the future value of an amount
  with compounded interest.’’’

  multiplier = 1 + rate
  FV = PV * (multiplier ** t)
  return FV
```

Usage:

```
compound_interest(1000, 0.4, 10)
```

→

```
1480.24428491
```

```
compound_interest(1000, 0.4)
```

→

```
TypeError
```

Functions

```
value = 1432315.33
future_value = compound_interest(value, 0.04, 3)
```

This is fine.

Functions

```
value = 1432315.33
future_value = compound_interest(value, 0.04, 3)
multiplier
```

```
def compound_interest(PV, rate, t):
    ‘‘‘Calculates the future value of an amount
    with compounded interest.’’’

    multiplier = 1 + rate
    FV = PV * (multiplier ** t)
    return FV
```

What about if we try to reference **`multiplier`**, the variable we created when defining **`compound_interest`**?

This errors! Variables that are created inside the function definition only exist inside the function.

Functions

You can even make lists of functions, since Python treats functions as values.

```
functions = [max, compound_interest, np.arange]
what_value = functions[2](3, 10, 1)
```

What’s the value of **`what_value`**?

```
[3, 4, 5, 6, 7, 8, 9]
```

Functions

You can even pass in functions as parameters to other functions!

```
def combiner(f, lst):
  total = lst[0]
  for i in range(1, len(lst)):
    total = f(total, lst[i])
  return total
```

Don't worry about what this does, but notice that it calls parameter **f** as a function in the 4th line.

```
def add(a, b): return a + b
s = combiner(add, [3, 4, 12])
s
```

→

```
19
```