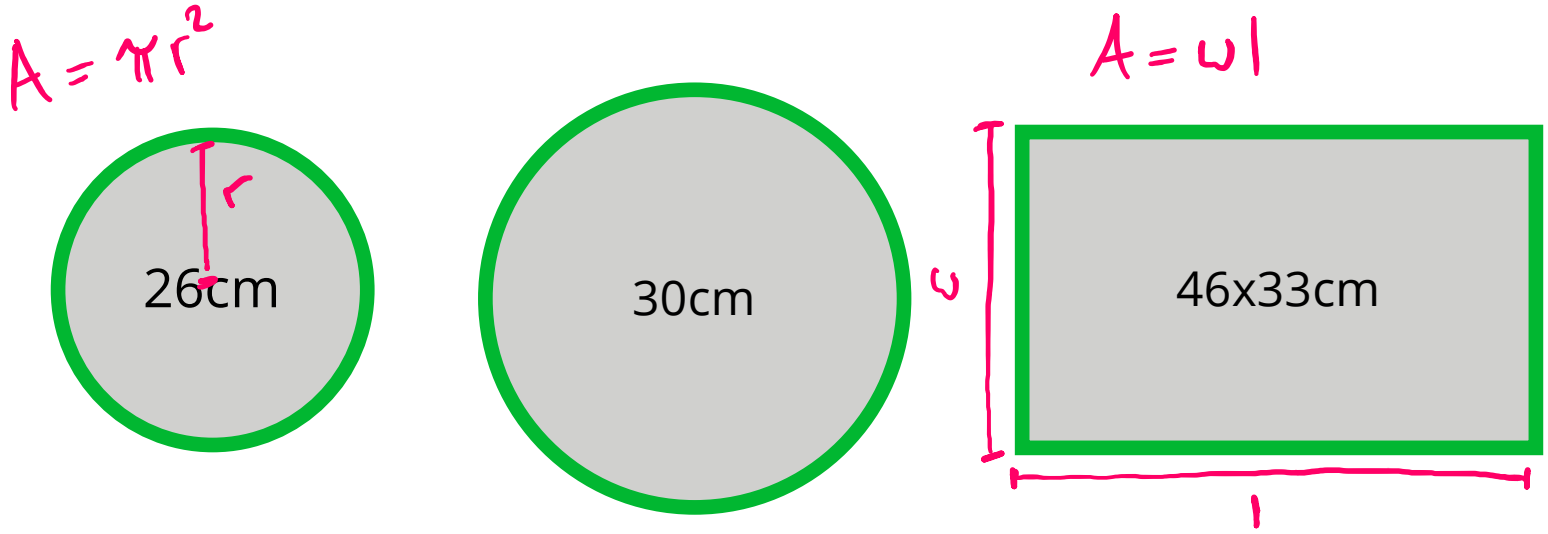
# 01
# The Pizza Problem

*Python Programming for Linguists*

Ingo Kleiber, 2020

# The Pizza Problem

At Sue's Pizza, you can oder **three types of pizza:**




**Small** for 4.80 €

$A \approx 531\ cm^2$

**Large** for 5.50 €

$A \approx 707\ cm^2$

**Party** for 13.00 €

$A \approx 1518\ cm^2$

# Problem

*Which pizza offers the best value?*

**Assumptions**

More pizza is strictly better.

Toppings are equal across sizes.

We have no preference regarding the shape.

...

# Solutions

For every (coding) **problem**, there are various solutions and approaches …

In **programming**, some common measures for **good solutions** are:

*(1)* simplicity *(2)* reusability *(3)* testability *(4)* understandability

*(5)* compliance *(6)* maintainability *(7)* efficiency *(8)* robustness

→ This can also be linked to principles of **good scientific practice**!

→ **Here**, we're aiming for a solution which is **just good enough**!

# A Solution

***Problem:*** *Which pizza offers the best value?*

1. Determine sizes, prices, and shapes of *n* pizzas
2. For each pizza, determine its area *(A)*
3. For each pizza, calculate the pizza to Euro ration *(PTER)*
4. Determine the best PTER

# Modeling Pizzas

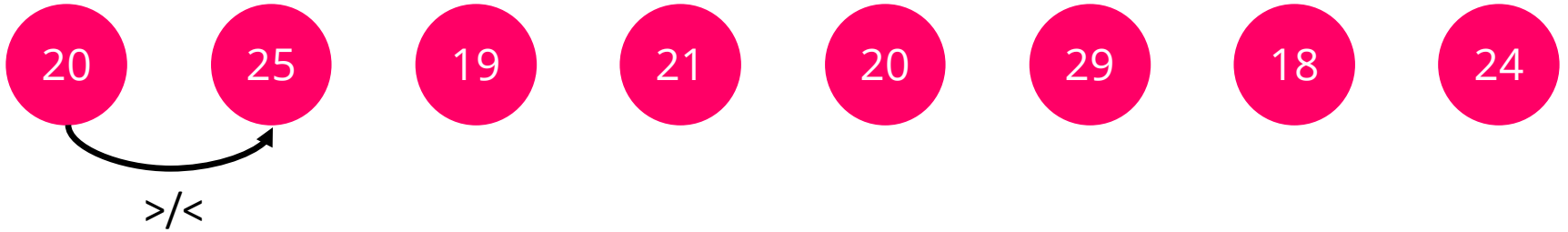
Type Size Price Shape

ps = [‘small’, [26, 0], 4.80, ‘circle’]

Shape is, implicitly,
encoded here as well!

# A Simple Algorithm

Imagine we wanted to **find the youngest and the oldest person** in the room …




**Youngest =** 100 20 20 19 19 19 18 18
**Oldest =** 0 20 25 25 25 29 29 29

*We also need to keep track of the current youngest and oldest person, for example, by using an index.*

# Bonus Exercises

1. How can we find the ideal (i.e. best priced) combination of pizzas for a given area that is being requested?
2. What if we were looking to optimize for as much or as little crust as possible?
3. What about a second or third size dimension (e.g., height)?
4. What about factoring in different types of toppings?