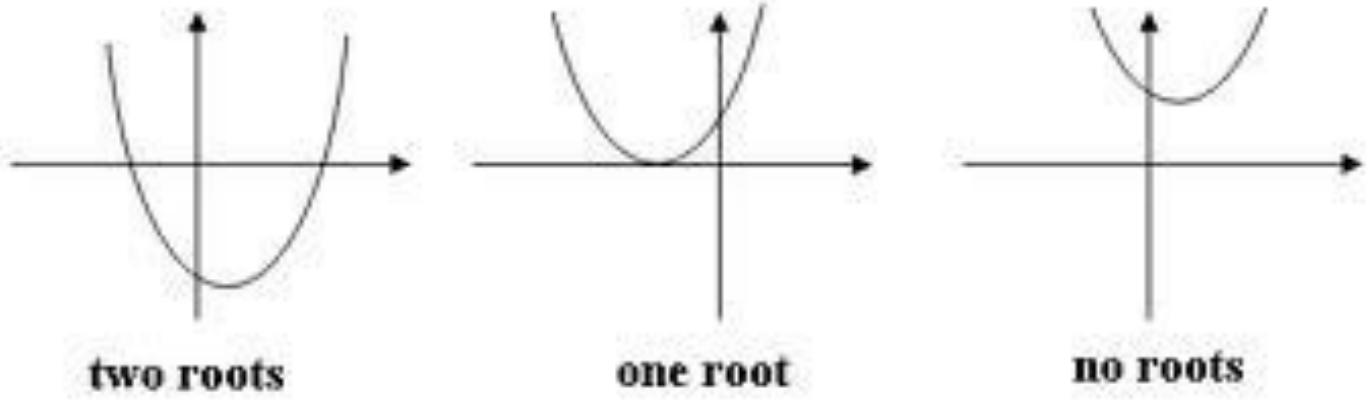
Chapter 7:
Decision Structures : Exceptions

Feb 13, 2020



Today's Outline

- Review:
  - Variables and Functions
  - Decision Structures: if statements
  - Midterm Review
- Exception Handling

A function can modify the value of an actual parameter only if it's

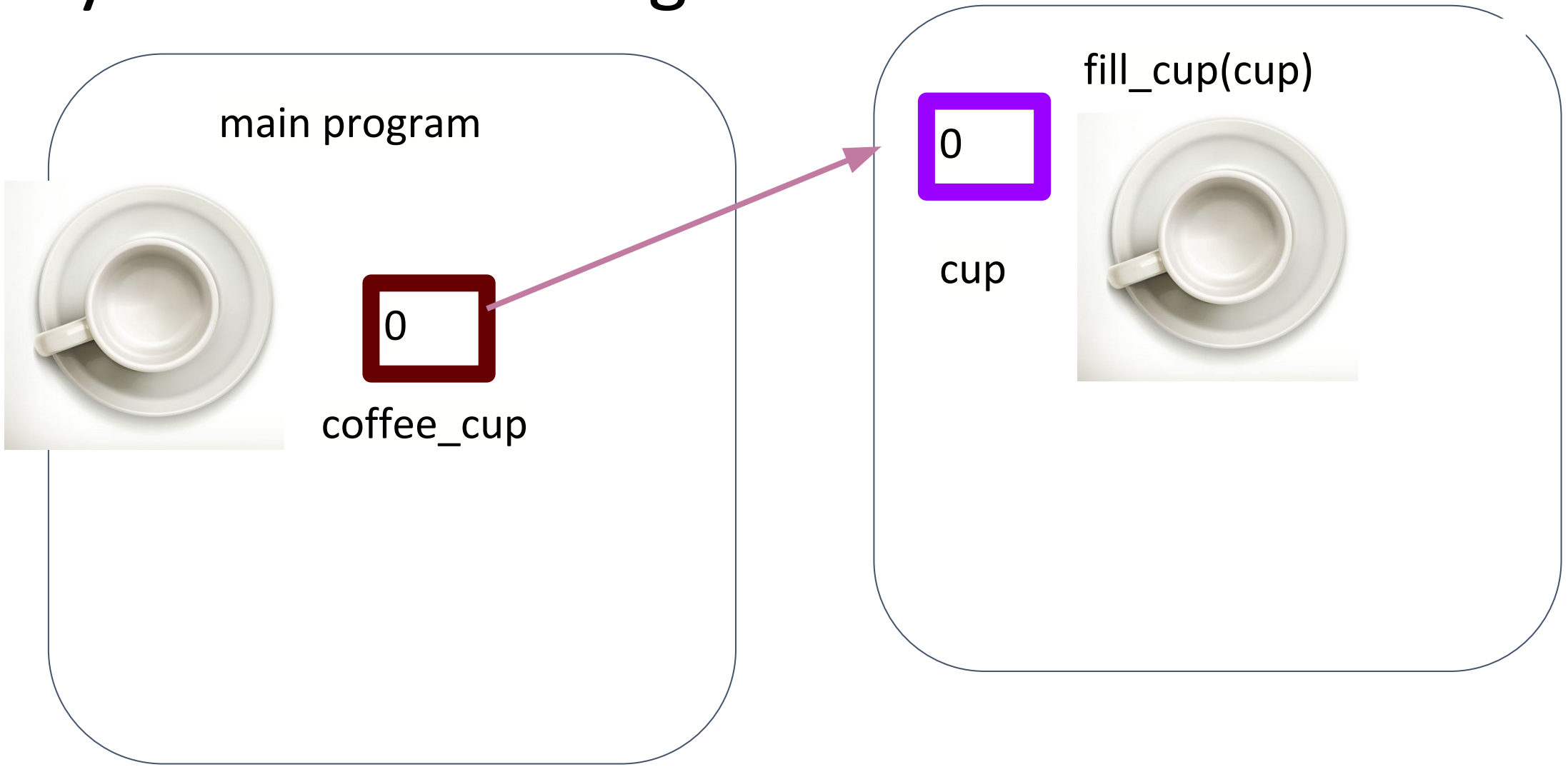
a) mutable
b) a list
c) passed by reference
d) a variable

Pass by Value vs. Pass by Reference

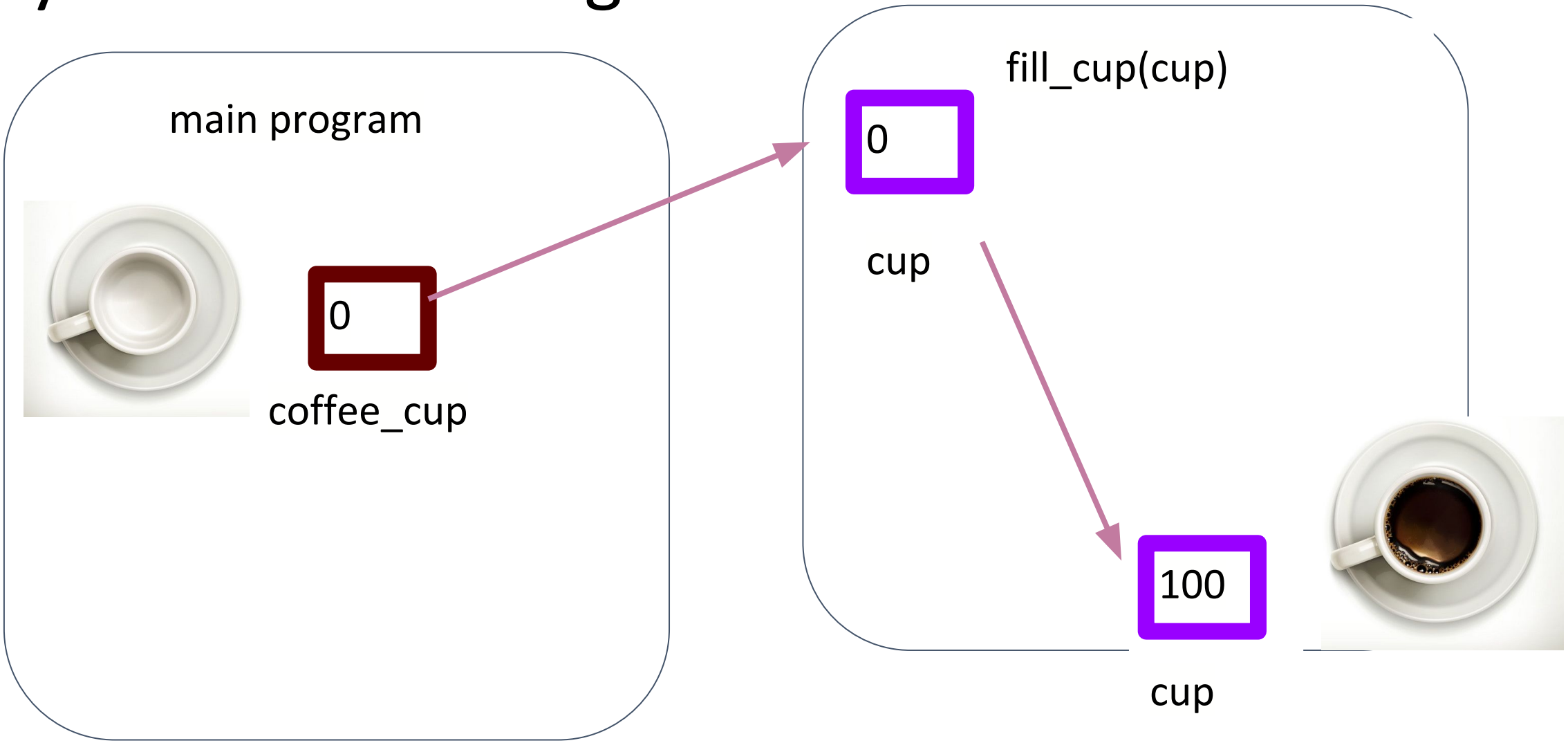
Pass by Value:

- the function copies the **values** of any variables given to it as parameters
- when passing by reference, changes that are made to the variable within the function don’t affect the variable in the main program
- in Python, this is how variables are passed into functions

Python Coffee Program



Python Coffee Program

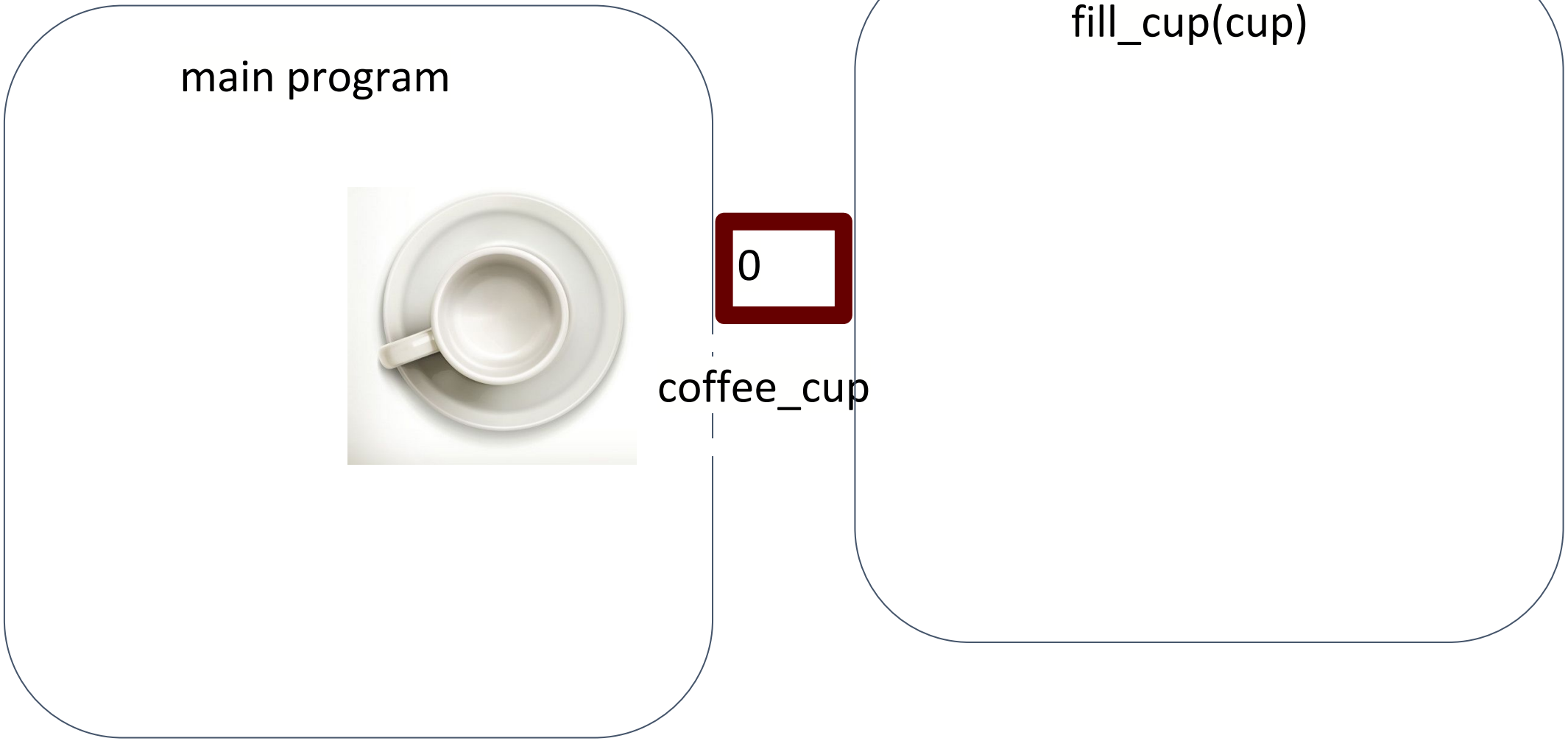


Pass by Value vs. Pass by Reference

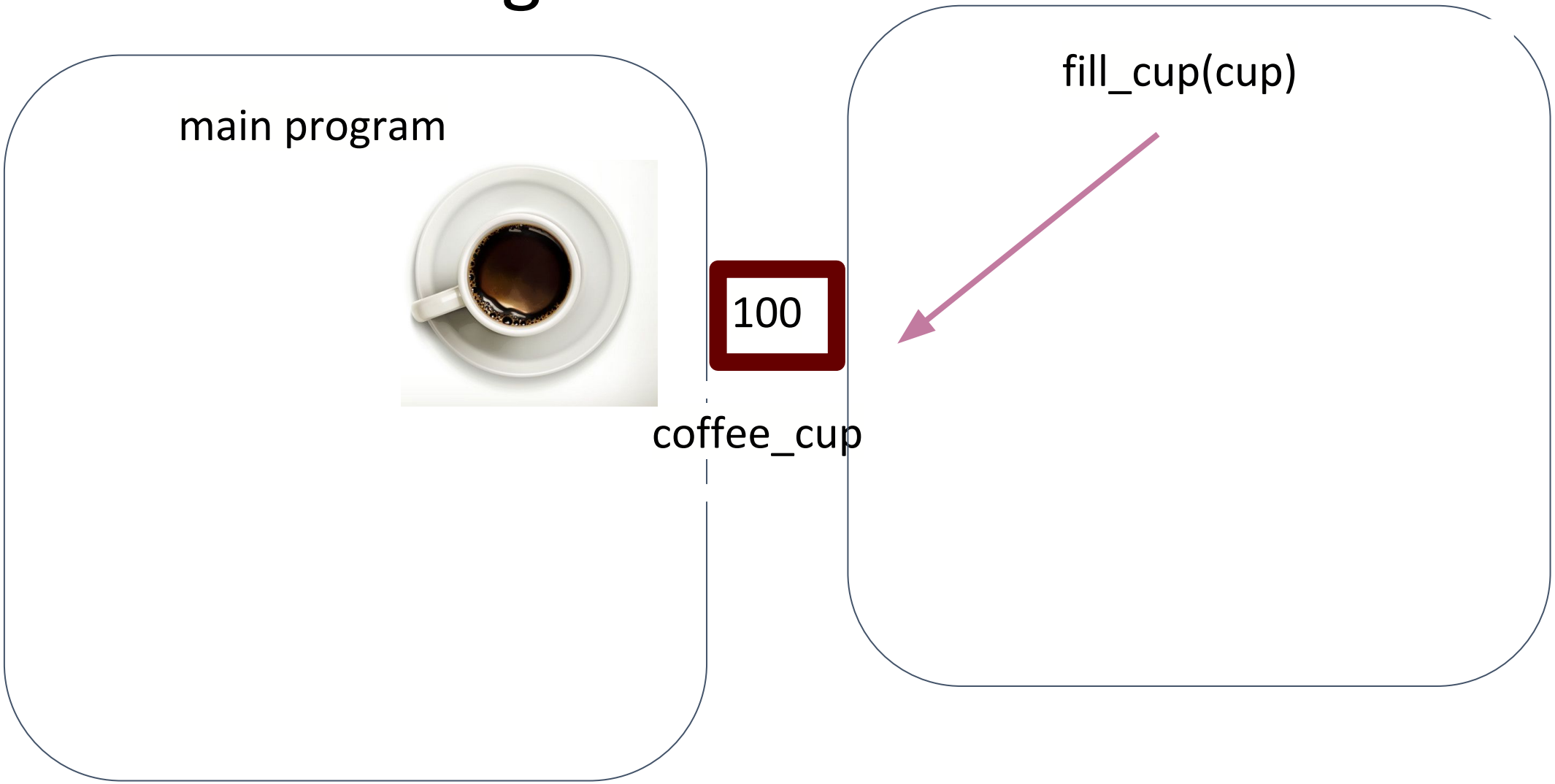
Pass by Reference:

- in some programming languages (ex. C++), you can pass the variable in the main program into the function
- changes that are made to the variable within the function will also affect the variable in the main program

C++ Coffee Program



C++ Coffee Program

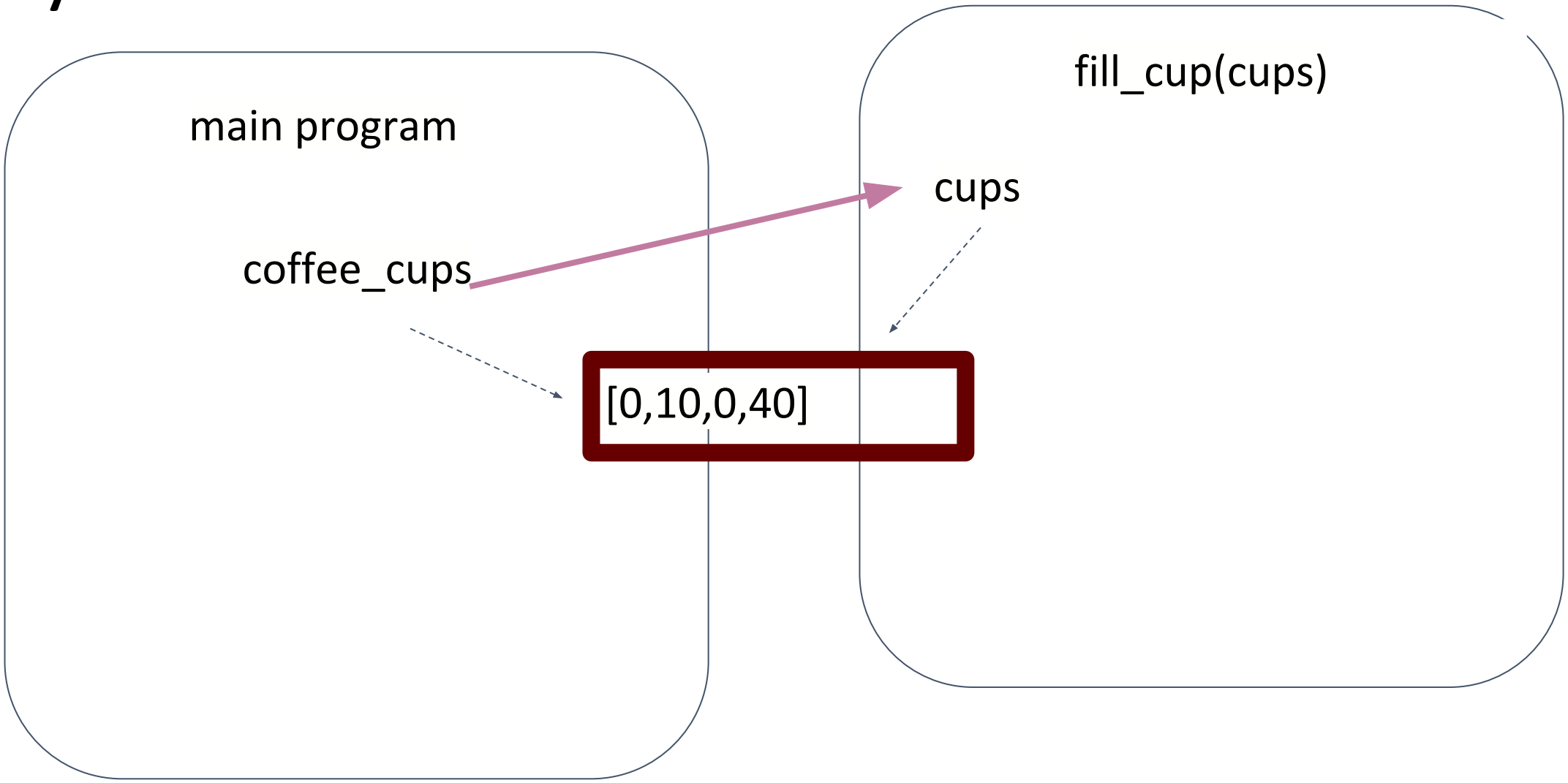


Python: Pass by Object Reference

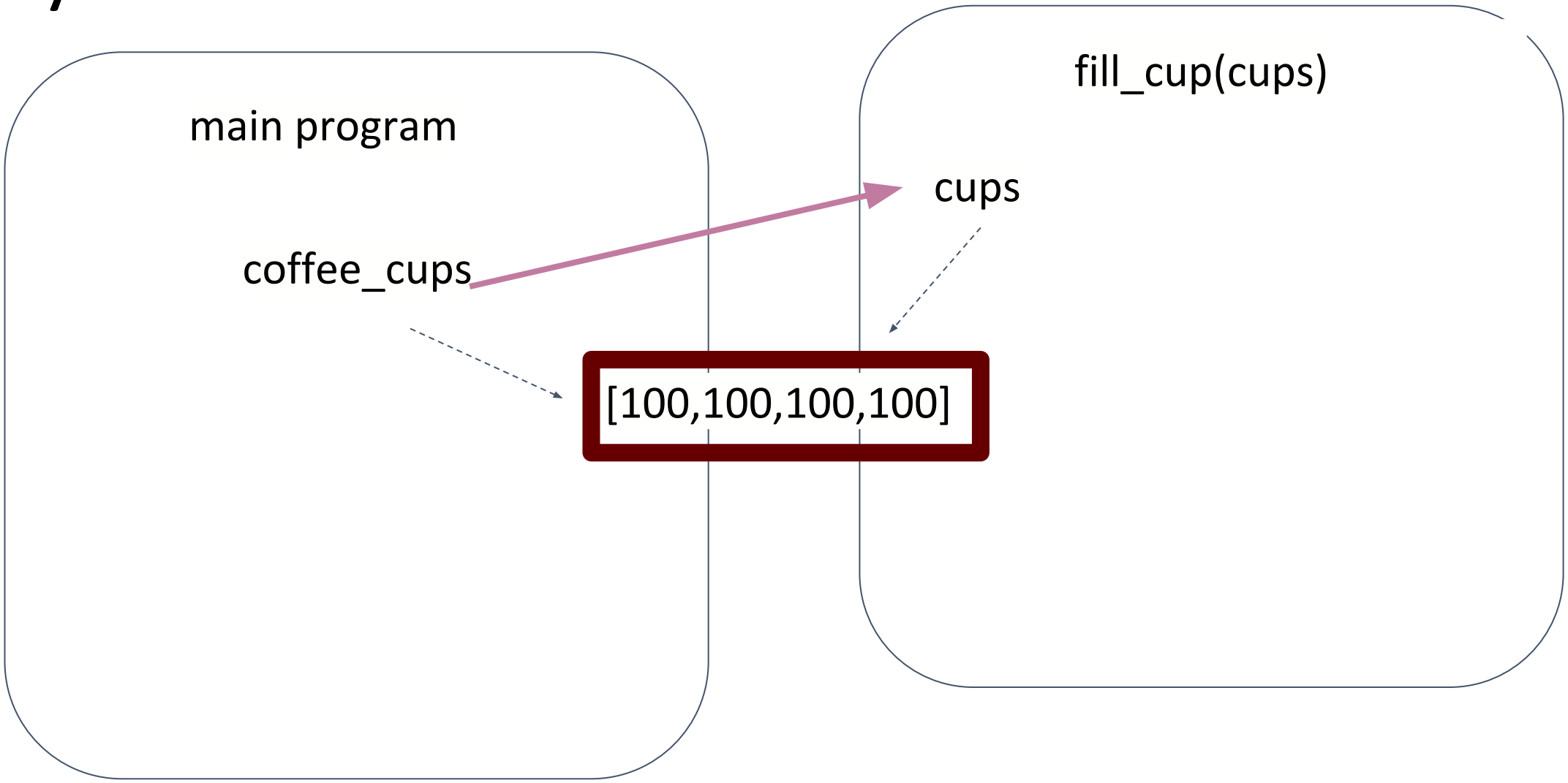
“Object references are passed by value”

- When using a list, as a parameter to a function, the function **copies** the value of the variable that refers to the list in memory
- if an item is appended to the list in the function, the same change occurs to the list outside the function, because they are different names for the same list
- https://robertheaton.com/2014/02/09/pythons-pass-by-object-reference-as-explained-by-philip-k-dick/

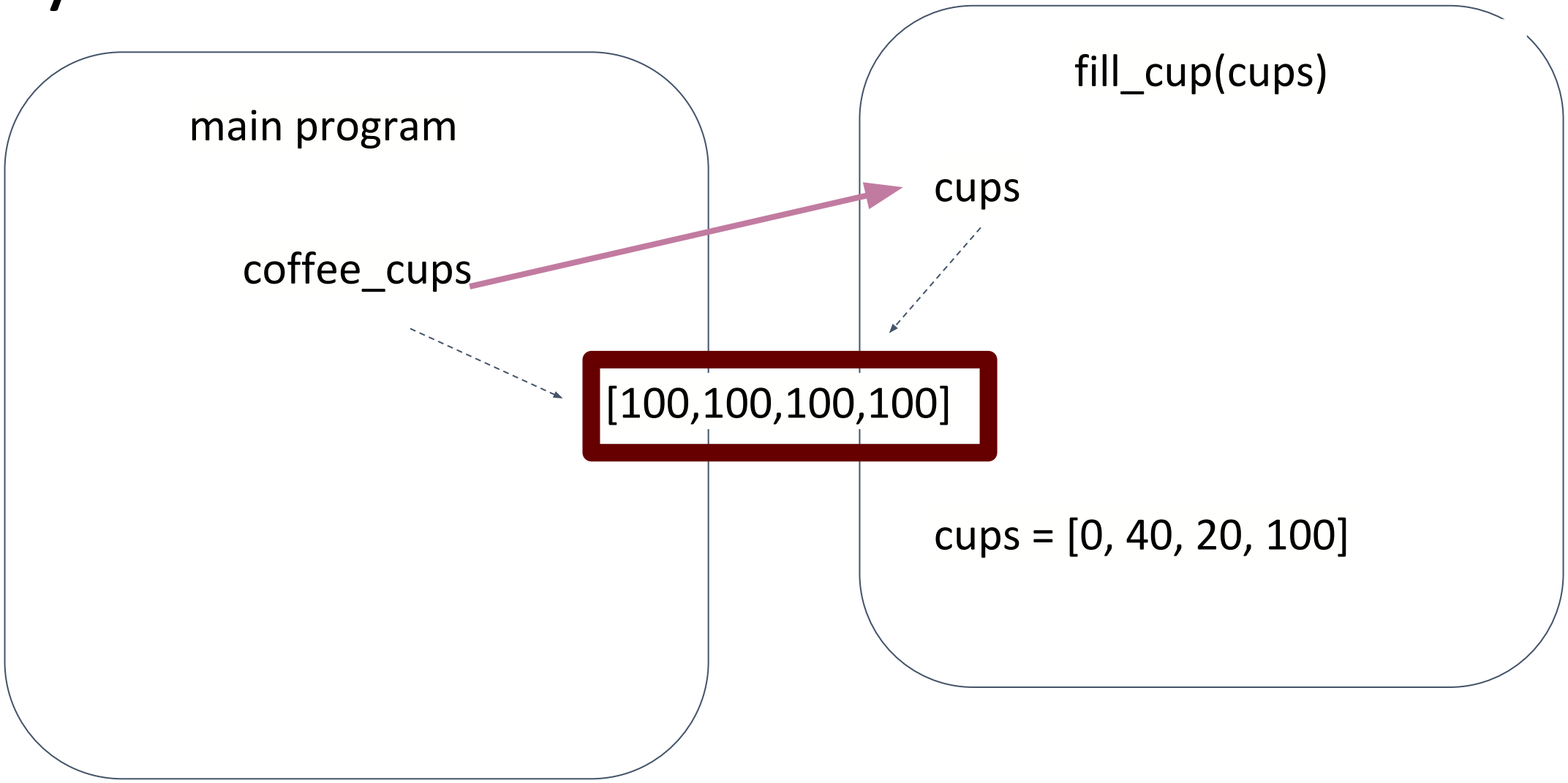
Python Coffee List



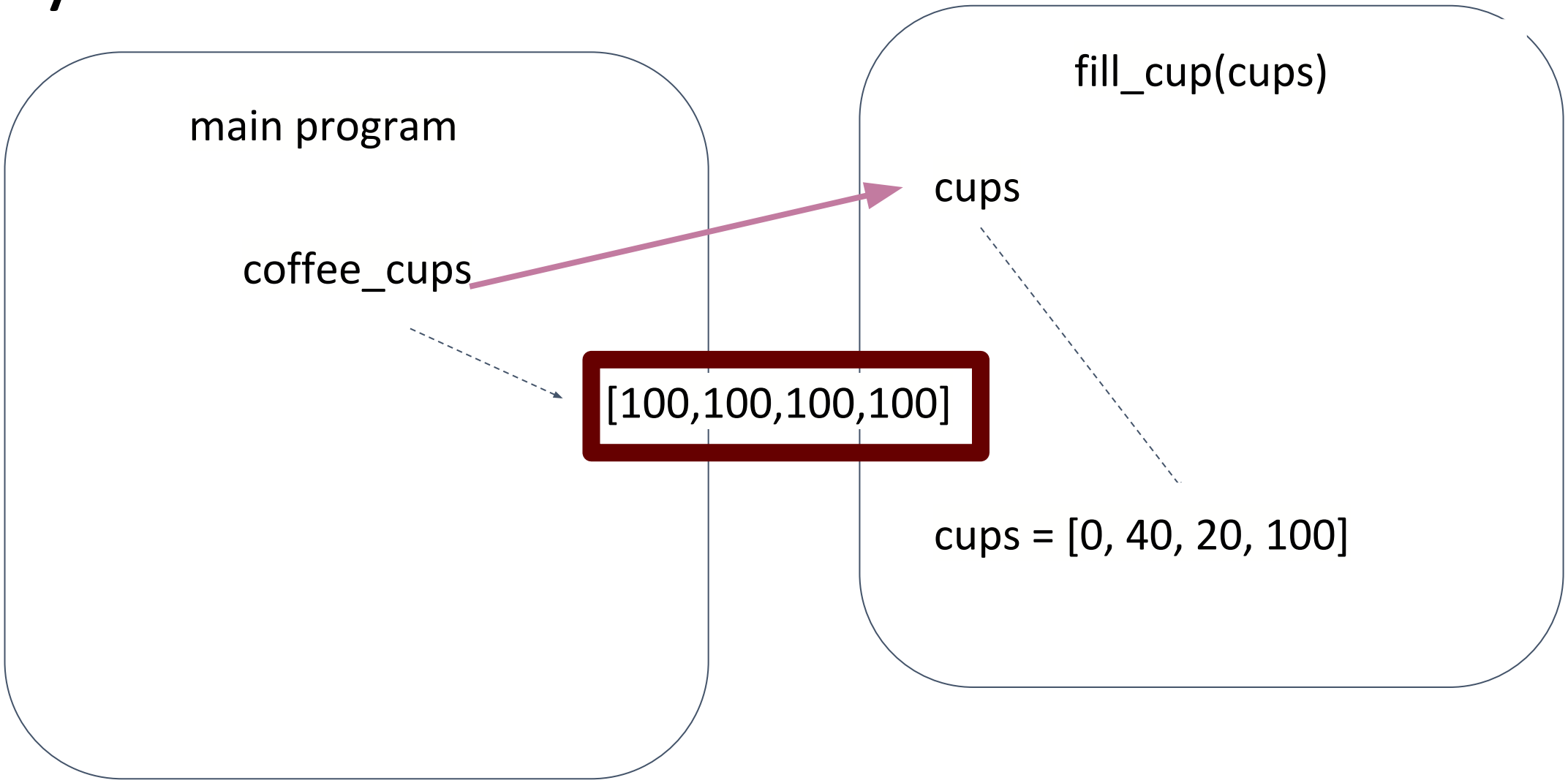
Python Coffee List



Python Coffee List



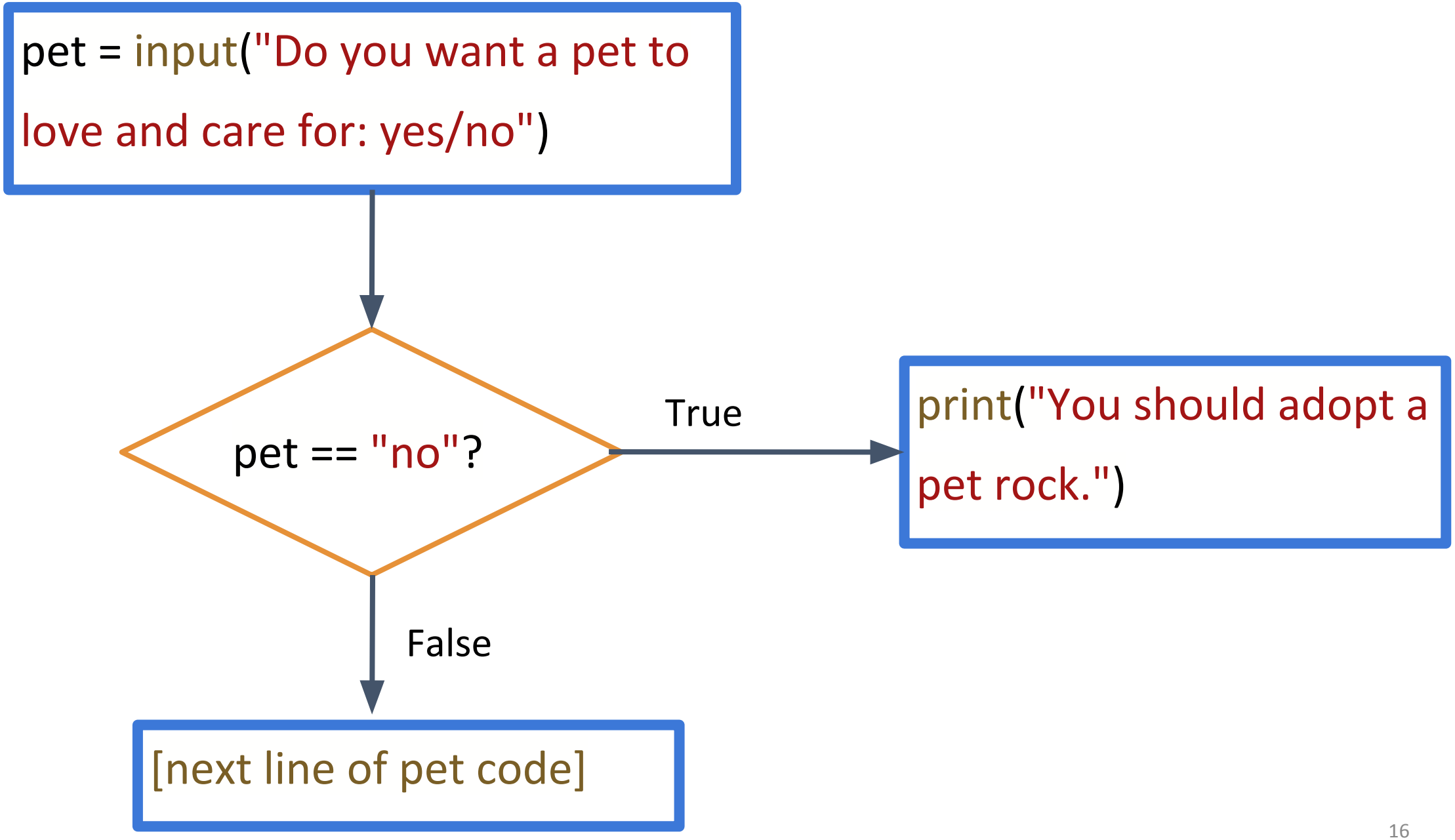
Python Coffee List



Basic Pet Decision

```
#pet
pet = input("Do you want a pet to love and care for?: yes/no")


if pet == "no":
  print("You should adopt a pet rock.")
```


pet = input("Do you want a pet to love and care for: yes/no")
pet == "no"?
True
print("You should adopt a pet rock.")
False
[next line of pet code]

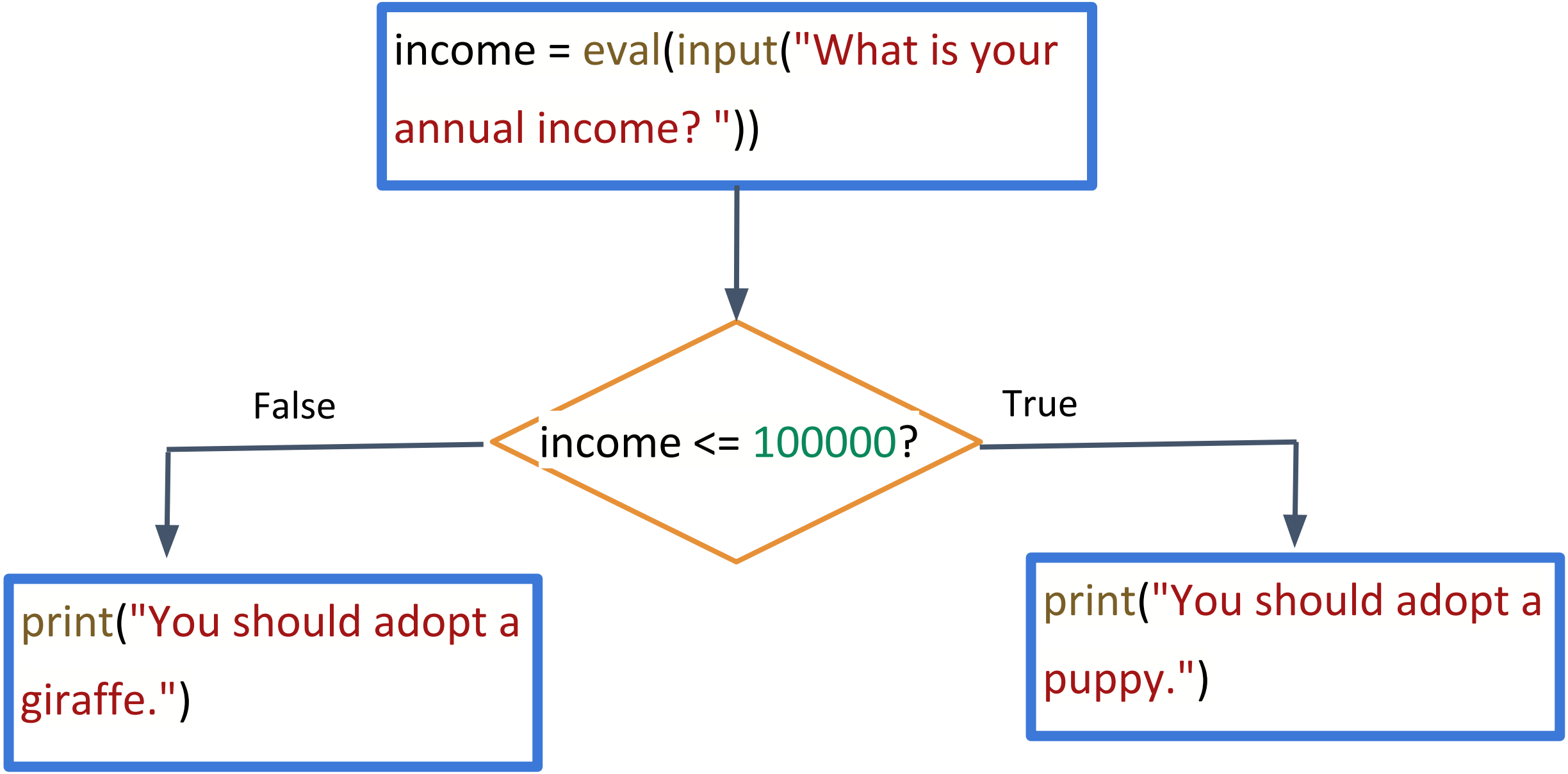
If - else statement: 2 way decision

```
if <condition 1>:
  <case 1 statement>
else:
  <default statement>
```

If - else statement: 2 way decision

```
#pet income
income = eval(input("What is your annual income? "))


if income <= 100000:
  print("You should adopt a puppy.")
else:
  print("You should adopt a giraffe.")
```


income = eval(input("What is your annual income? "))
income <= 100000?
False
True
print("You should adopt a giraffe.")
print("You should adopt a puppy.")

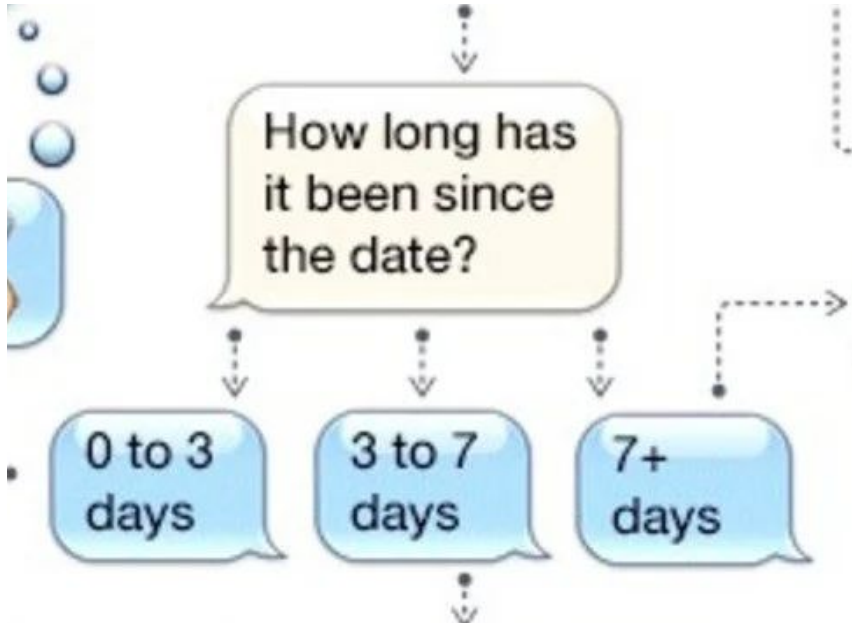
Multi-Way Decision

```
if <condition 1>:
  <case 1 statement>
elif <condition 2>:
  <case 2 statement>
elif <condition 3>:
  <case 3 statement>
else:
  <default statement>
```

Multi-Way Decision

```
#bad dating advice
days = eval(input("How many days has it been since your date? "))


if days <= 3:
  print("Wait to see if he texts.")
elif days <= 7:
  print("Call his mom to ask if he is alive.")
else:
  print("He was most likely abducted by aliens :( ")
```



Conditional Program Execution

Sometimes, it may be necessary to create a hybrid module that can both: 1) run as a stand-alone program or 2) can be imported as a library.

```
if __name__ == '__main__':
  print("program is running directly")
  main()
else:
  print("program is not running directly")
```

Leap year

A year is a leap year if it is divisible by 4, unless it is a century year that is not divisible by 400. (1800 and 1900 are not leap years while 1600 and 2000 are.) Write a program that calculates whether a year is a leap year.

Leap year

```
year = 2020


if year%100==0 and year%400!=0:
  print("not leap year")
elif year%4 ==0:
  print("leap year")
```

Pig Latin Bonus

Napoleon the pig is very clever and wants to read War and Peace.

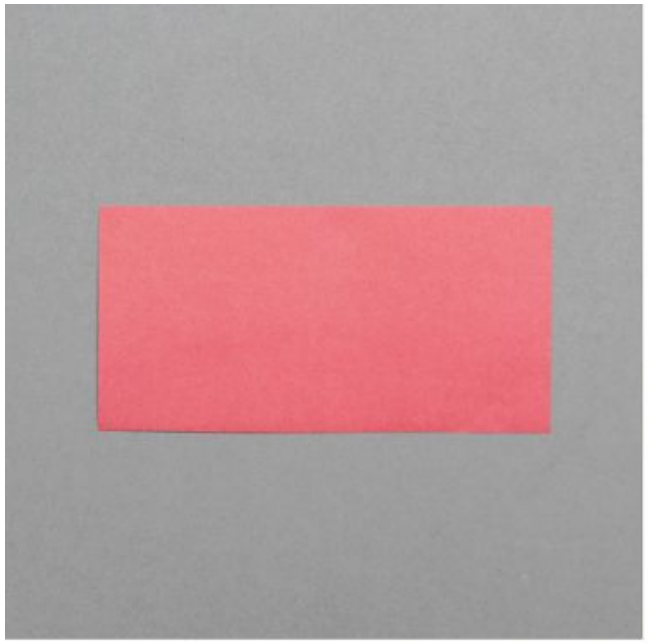
Make a program that translates War and Peace into Pig Latin and saves it into a .txt file so that he can read the book.

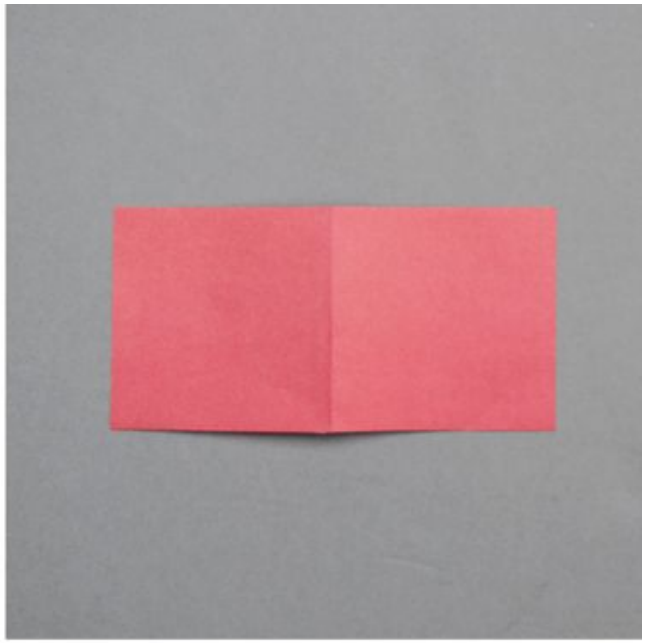
Submit it to him by Feb 14.

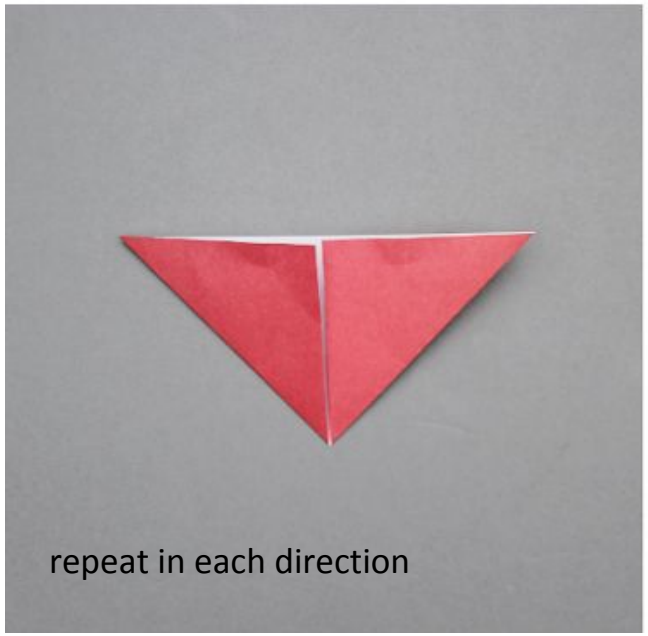
Valentines Day Activity

Make an origami heart

https://www.gatheringbeauty.com/blog/2019/2/how-to-make-origami-blossom-hearts

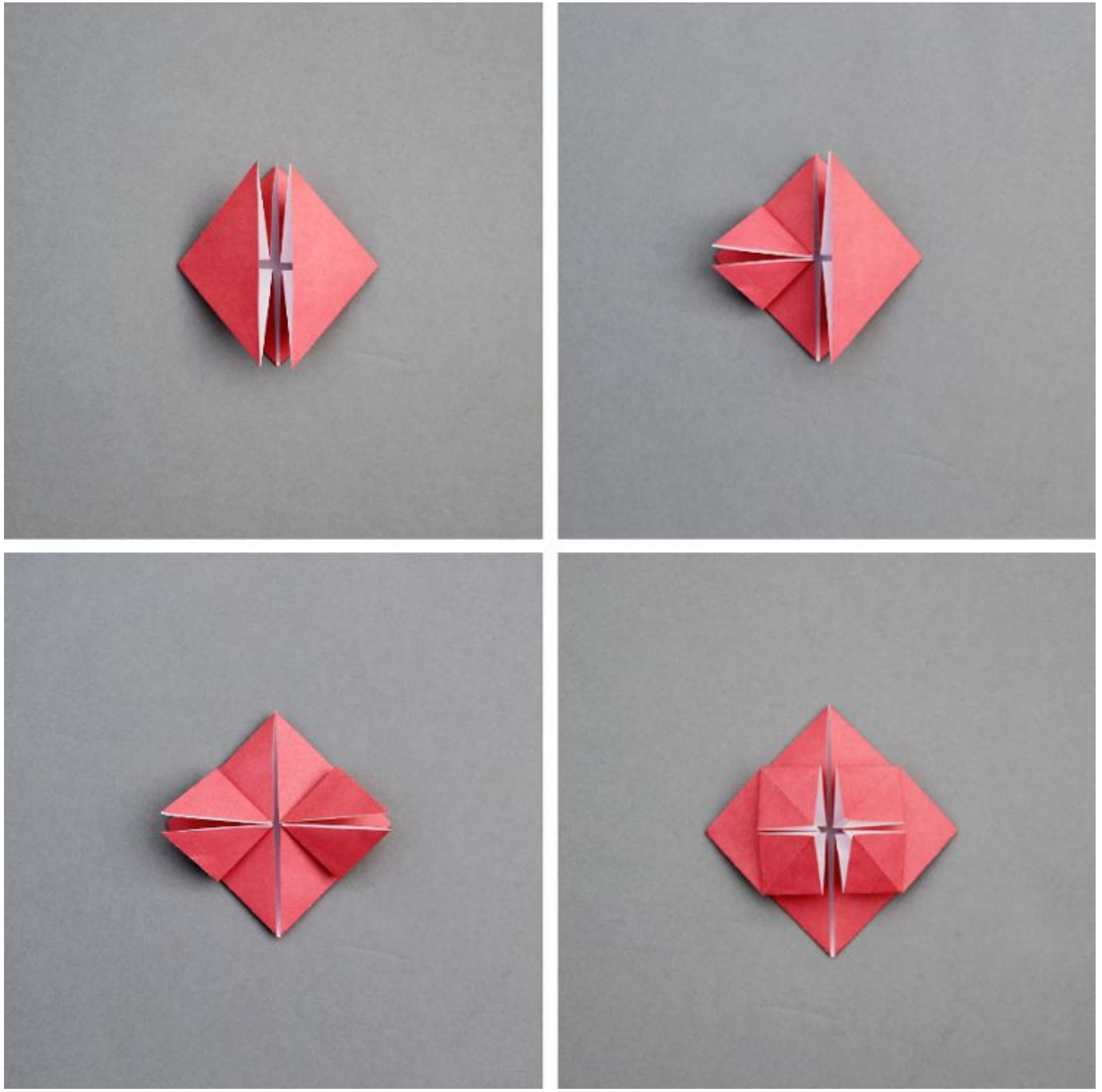


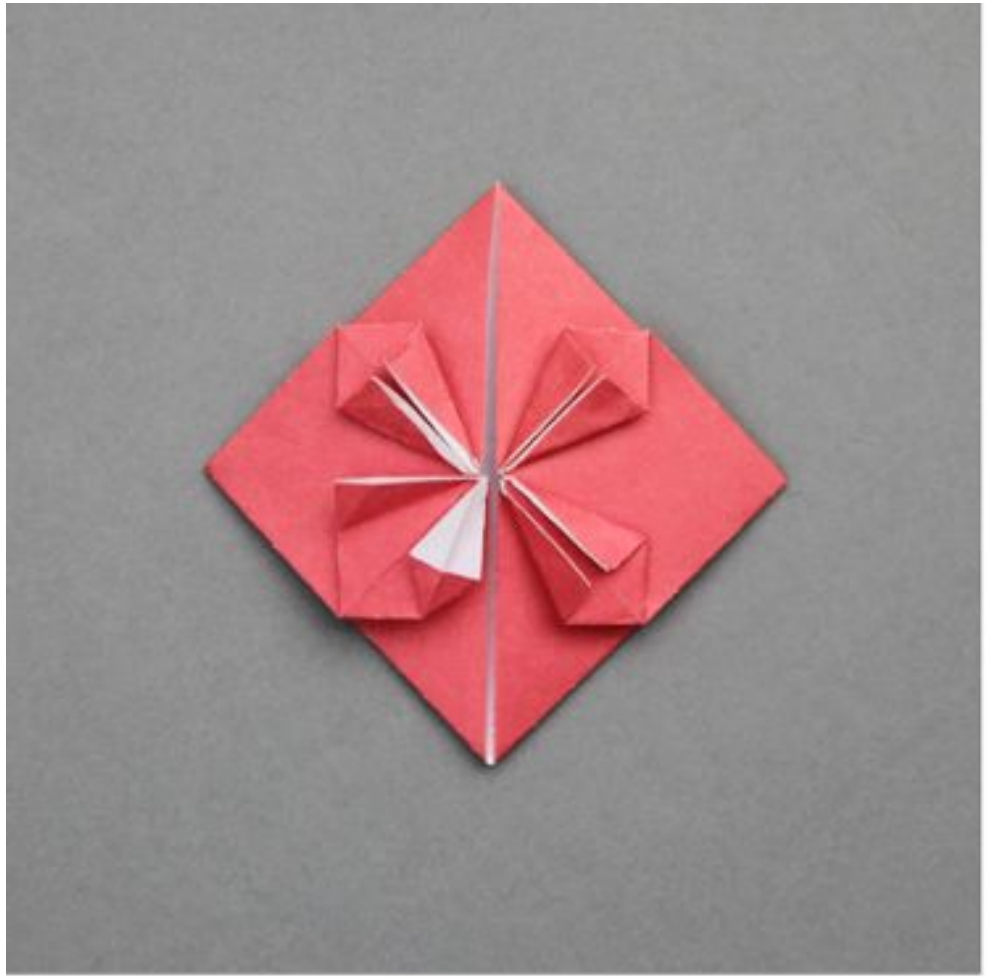


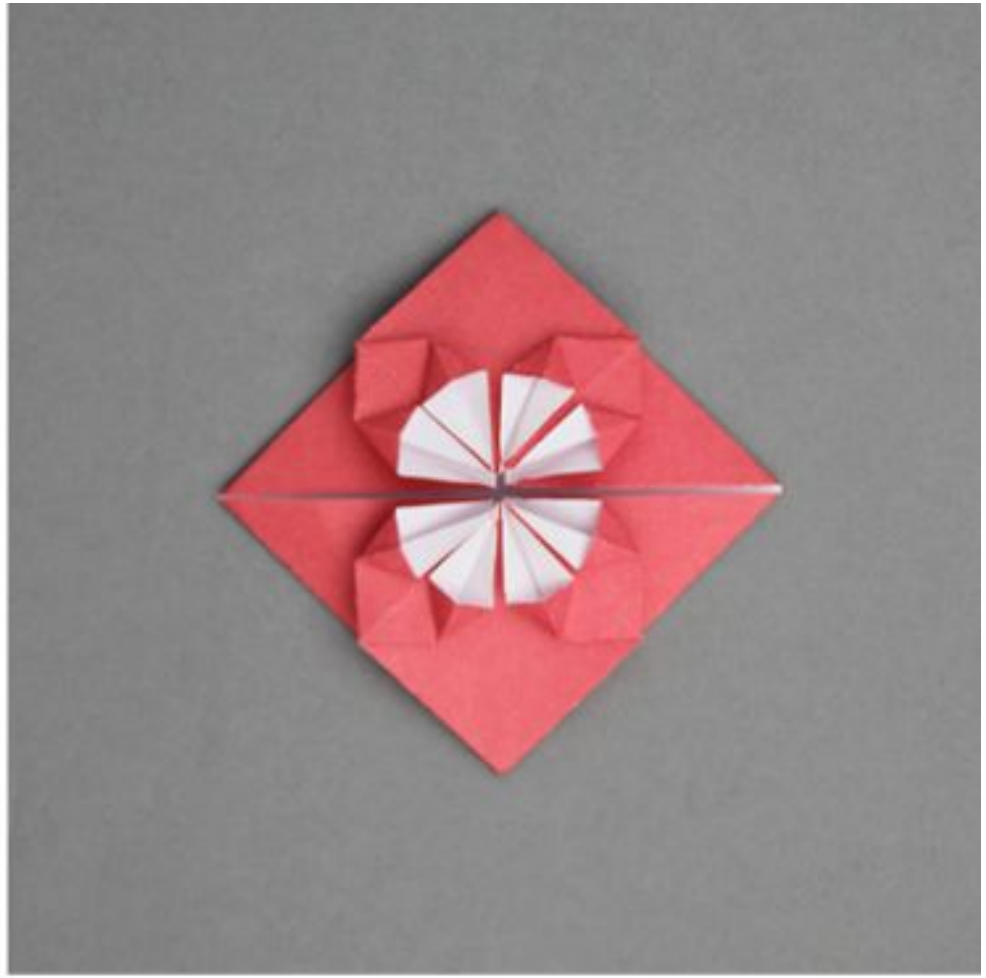


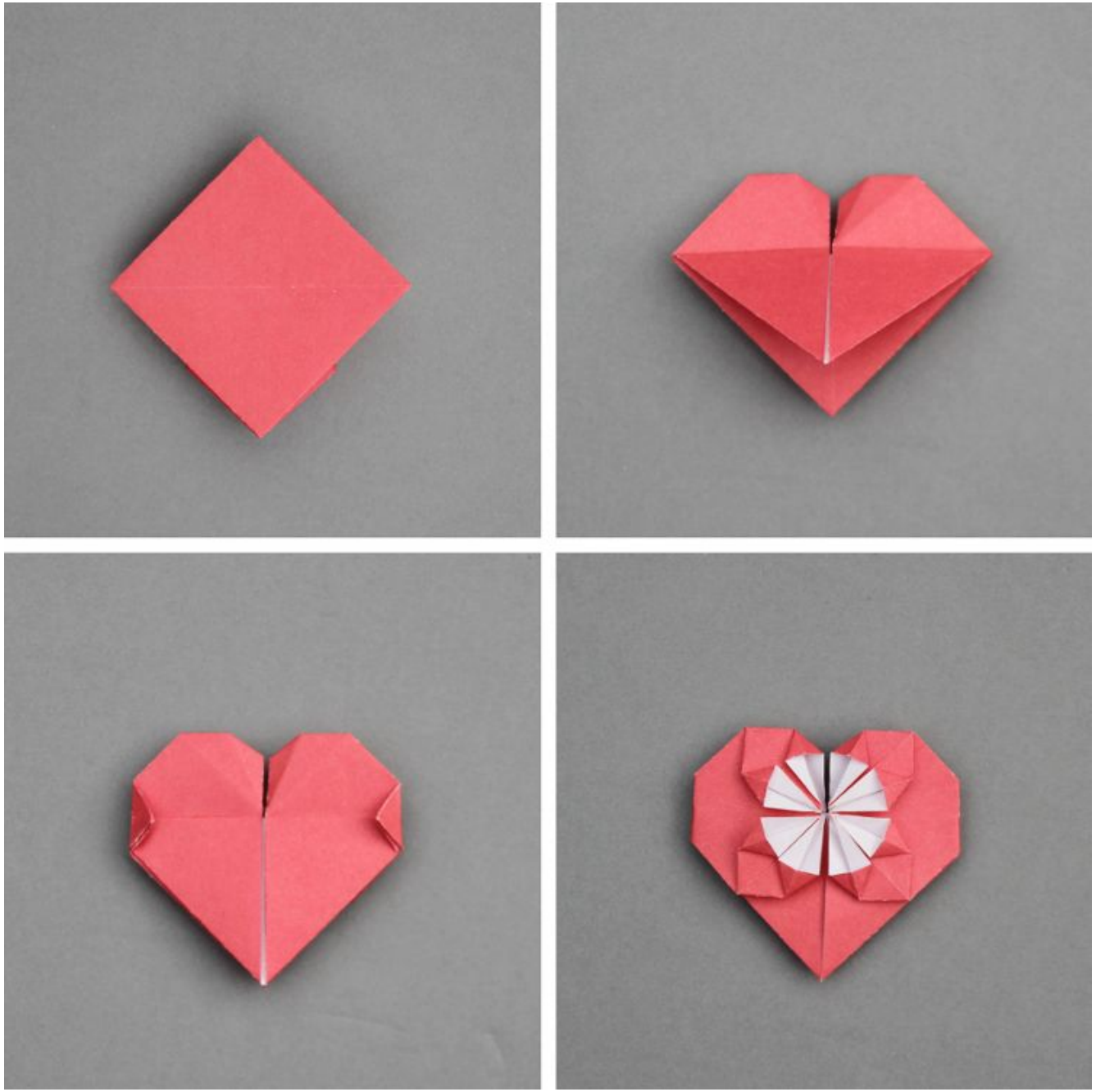
repeat in each direction

Squash folds









Topics we learned

- functions
- for loops
- number data types (floats and ints)
- math, NumPy, and graphics libraries
- object oriented programming (classes, methods, objects)
- sequences (strings, lists)
- files
- decision structures

What type of programming paradigm should we use?

- imperative
- procedural
- object-oriented

What types of functions or methods do we need?

- fold
- rotate
- squash

What to do if there is a mistake?

- go back and refold?
- keep going and hope for the best?
- give up?

Midterm

Remember that the midterm is coming up on **Friday February 28**.

Topics: Chapters 1-7

Understand: Example Programs and Labs

Practice: Additional programming problems are posted

Note: You can bring all of your notes and solved example programs to the Midterm, so you don’t need to memorize anything!

Exception Handling

If an error occurs while a program is running, it is beneficial for the program to have a mechanism to deal with this error.

Examples of common errors:

- take the square root of a negative value
- divide by zero
- value is a string but we want it to be a number or vice/versa
- list index is out of range

Quadratic Equation Code

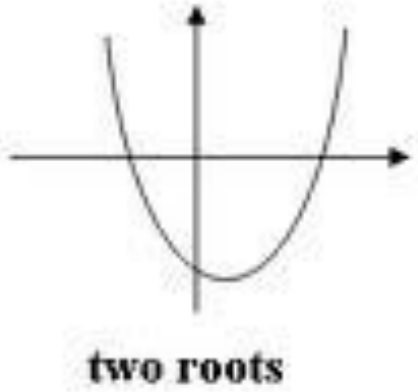
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```
import math
a = eval(input("Enter a value for a: "))
b = eval(input("Enter a value for b: "))
c = eval(input("Enter a value for c: "))

x1 = (-b + math.sqrt(b**2-4*a*c))/(2*a)
x2 = (-b - math.sqrt(b**2-4*a*c))/(2*a)

print("The roots are: {0},{1}".format(x1,x2))
```

Finds the roots for the quadratic function $ax^2+bx+c = 0$

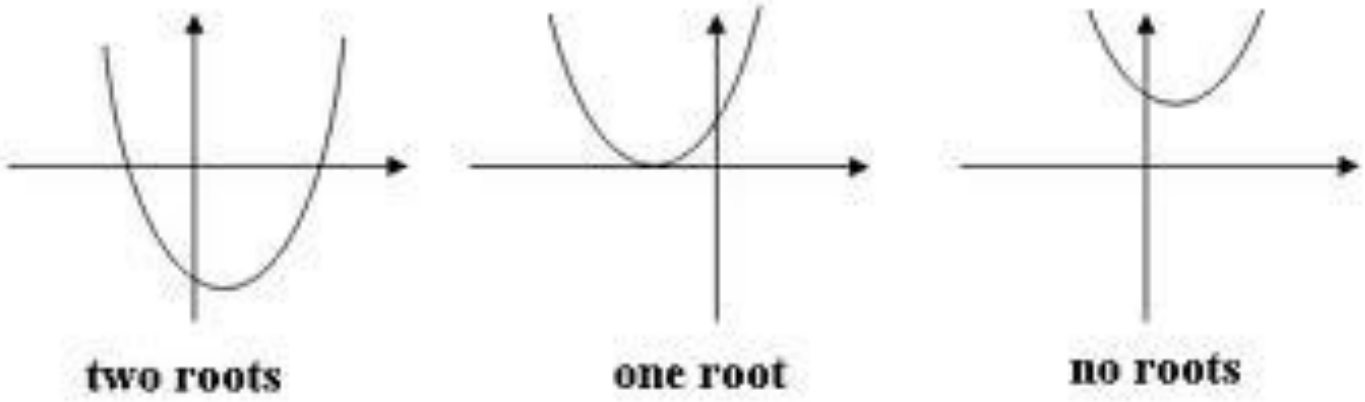


Quadratic Equation

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Write a program to determine if there are:

a) no real roots
b) one root
c) two real roots



for the quadratic function $ax^2+bx+c = 0$

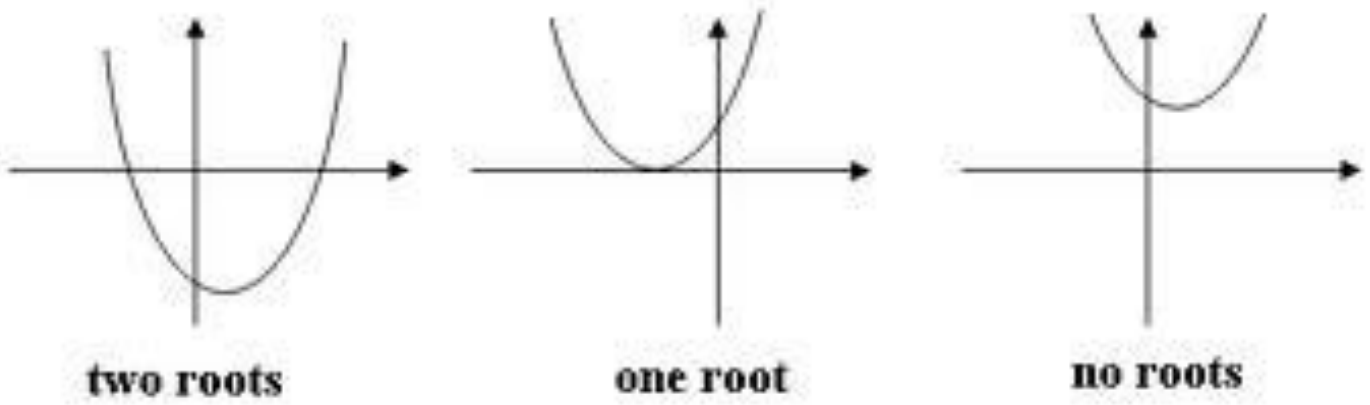
Quadratic Equation

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

1) If b^2 - 4ac < 0, there will be no real roots because there is no real root of a negative number.

2) If b^2 - 4ac = 0, there will only be one root, x=-b/(2a).

3) If b^2 - 4ac > 0, there will be two roots.



Exception Handling

It can become very difficult to consider all of the possible ways that a code might crash.

Python has some built-in behaviours that assist with exception handling.

Exception Handling

```
try:
  <try to run this code>
except <ErrorType>:
  <run this code if there is an error>
```

Exception Handling: Divide by Zero

```
x,y = 3,0
try:
    print(x/y)
except ZeroDivisionError:
    print("you can't divide by 0")
```

Quadratic root problem

ValueError: math domain error

```
try:
    [quadratic function code]
except ValueError:
     [there are no real roots]
```

Types of Errors

https://docs.python.org/3/library/exceptions.html

Exception Handling Benefits

1) The program doesn’t crash if there are no real roots.
2) We could update the program to work for other invalid input values, for example so that if the user inputs a string value instead of a number, the program will not crash.

Quadratic Code with Error Handling

```
import math
try:
  a = float(input("Enter a value for a: "))
  b = float(input("Enter a value for b: "))
  c = float(input("Enter a value for c: "))
  x1 = (-b + math.sqrt(b**2-4*a*c))/(2*a)
  x2 = (-b - math.sqrt(b**2-4*a*c))/(2*a)
  print("The roots are: {0},{1}".format(x1,x2))
except ValueError:
  print("No real roots.")
```

Multiple Exceptions

```
import math
try:
  a = float(input("Enter a value for a: "))
  b = float(input("Enter a value for b: "))
  c = float(input("Enter a value for c: "))
  x1 = (-b + math.sqrt(b**2-4*a*c))/(2*a)
  x2 = (-b - math.sqrt(b**2-4*a*c))/(2*a)
  print("The roots are: {0},{1}".format(x1,x2))
except ValueError as excObj:
  if str(excObj)== "math domain error":
    print("No real roots.")
  else:
    print("Invalid coefficient provided.")
except:
  print("Oops something went wrong! Sorry")
```

Exception Objects

```
except ValueError as excObj:
  if str(excObj)== "math domain error":
    print("No real roots.")
```

*** Exceptions are a type of object. This code assigns the exception object to the variable excObj. The variable is then converted to a string to check if the message is the math domain error.

Update the leap year program so that it won’t crash if the user doesn’t enter a numerical year.

Exception Handling with a while loop

```
while True:
    try:
        x = int(input("Please enter a number: "))
        break
    except ValueError:
        print ("Oops!  That was not a number.  Try again...")
```

Raising an Exception

Even if there is no Python error, we may still wish to identify a problem with the code. For example, we may not want to have a student's grade in a course to be greater than 100.

```
grade = 500
if grade > 100:
  raise Exception('The student grade should not exceed 100. The value
of the grade entered was: {}'.format(grade))
```