

# Topic 4: Decisions

# Recommended Readings

- Strongly Recommended Exercises
  - The Python Workbook, 2<sup>nd</sup> Edition: 39, 40, 46, and 57
- Recommended Exercises
  - The Python Workbook, 2<sup>nd</sup> Edition: 37, 44, 48, 50, 58, and 59
- Recommended Readings
  - The Python Workbook, 2<sup>nd</sup> Edition: Chapter 2

# Review

- What kinds of statements have we seen so far?
  - Assignment statements
  - Input statements
  - Output statements
- These are generally necessary, but not sufficient, to solve “interesting” problems

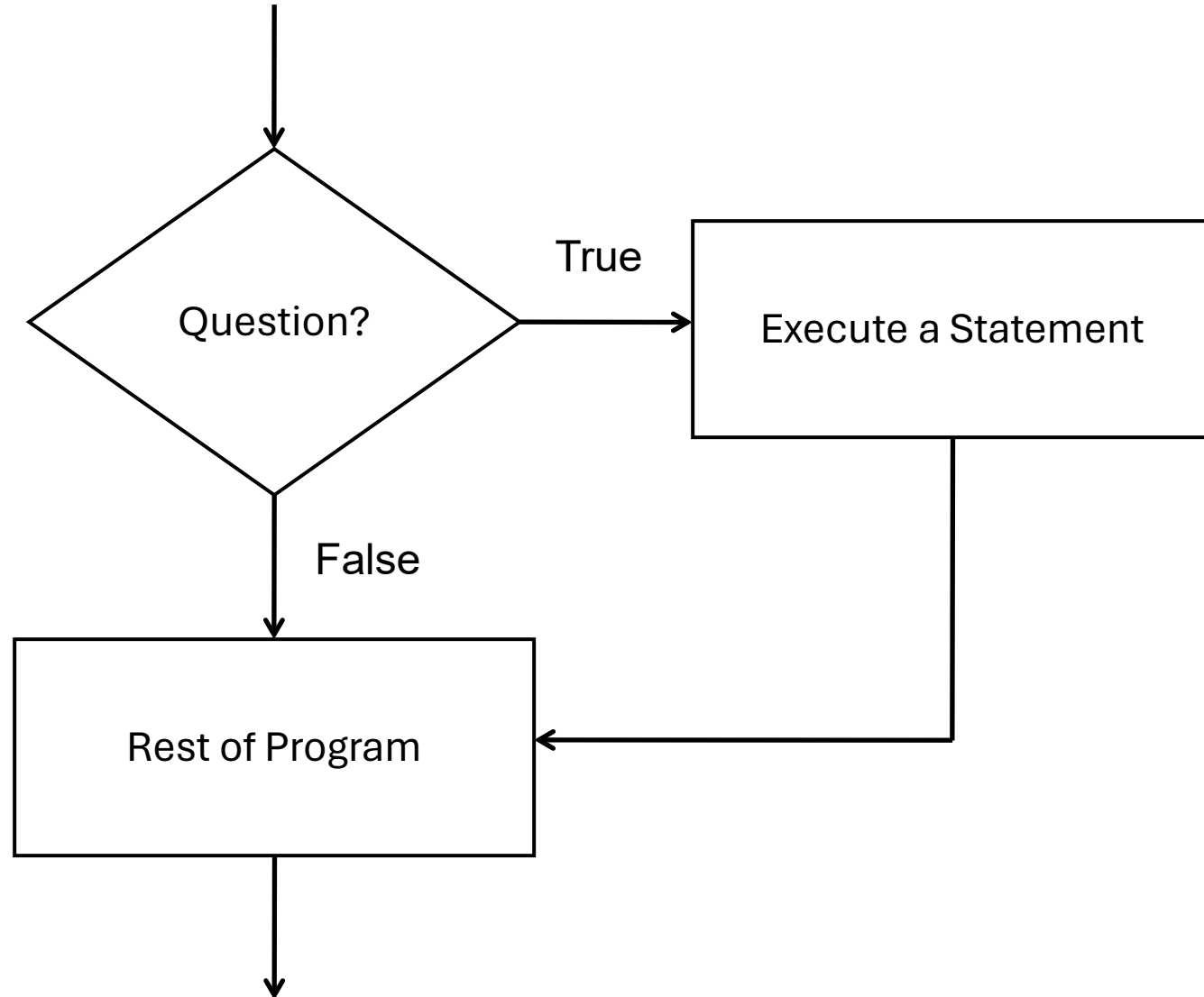
# Example

- Determine the state of gold when it is at a given temperature
  - Gold is solid when the temperature less than 1064 degrees Celsius
  - Gold is liquid when the temperature is between 1064 and 2856 degrees Celsius
  - Otherwise gold is gaseous

# If Statements

- If statements
  - Permit or prevent another statement from executing
  - Start with the word `if`
  - Allow us to test anything that can be determined to be true or false
- General Form:
  - `if` condition:  
    body

# Decisions



# Condition

- The condition portion of an if statement must be a Boolean result
  - True or False
  - Can be
    - Value of a variable
    - Result of a function
    - Result of a relational operator
    - ...

# Relational Operators

- Relational operators compare two values
  - Result will be true or false
  - Operators:
    - < less than
    - > greater than
    - <= less than or equal
    - >= greater than or equal
    - == equal
    - != not equal



# Relational Operations

- Values tested can be
  - Variables
  - Literals
  - Results from functions
  - Expressions
  - ...
- Types tested can be
  - Integers, Floats, Booleans, Strings
  - ...

# Gold Example

# Liquid Gold?

- How do we test whether the gold is liquid?
  - temperature must be greater than 1064 degrees Celsius
  - temperature must be less than 2856 degrees Celsius

# Boolean Logic

- A system of logical values and operators
  - Values
    - True, False
  - Operators
    - And
    - Or
    - Not
    - Xor
    - ...
  - Used to form complex conditions

# Boolean Logic

- Truth tables describe the behavior of logical operators

| Operand(s)        | Result           | A      | not A |
|-------------------|------------------|--------|-------|
| Operand<br>Values | Result<br>Values | 0<br>1 |       |

- The not operator flips the value of its operand

# Boolean Logic

- And operator
  - Takes two operands
  - Produces one result
  - Result is true if and only if both operands are true

| A | B | A and B |
|---|---|---------|
| 0 | 0 |         |
| 0 | 1 |         |
| 1 | 0 |         |
| 1 | 1 |         |

# Boolean Logic

- Or operator
  - Takes two operands
  - Produces one result
  - Result is true if one operand is true (or both operands are true)

| A | B | A or B |
|---|---|--------|
| 0 | 0 |        |
| 0 | 1 |        |
| 1 | 0 |        |
| 1 | 1 |        |

# Boolean Logic

- Exclusive or operator
  - Takes two operands
  - Produces one result
  - Result is true if exactly one operand is true

| A | B | A xor B |
|---|---|---------|
| 0 | 0 |         |
| 0 | 1 |         |
| 1 | 0 |         |
| 1 | 1 |         |



# Boolean Logic

- Python doesn't include an xor operator
- What logical expressions can we use to achieve the same result?

# Boolean Logic

- When is not(A and B) true?

| A | B | A and B | not (A and B) |
|---|---|---------|---------------|
| 0 | 0 |         |               |
| 0 | 1 |         |               |
| 1 | 0 |         |               |
| 1 | 1 |         |               |

- We call this operation nand

# Boolean Logic

- When is not(A or B) true?

| A | B | A or B | not (A or B) |
|---|---|--------|--------------|
| 0 | 0 |        |              |
| 0 | 1 |        |              |
| 1 | 0 |        |              |
| 1 | 1 |        |              |

- We call this operation nor

# Boolean Logic

- Example:
  - Construct a truth table for  $A$  and  $(B \text{ or not } C)$ :

# Boolean Logic

- Boolean logic is the basis for computation in modern computers
  - Circuits can implement logical operations
  - Arithmetic operations can be built up from logical operations
  - Memory can be constructed by including feedback loops in the circuits

# Gold Example

# Precedence

- Relational and logical operators have lower precedence than mathematical operators
  - Mathematical Operators
  - Relational Operators
  - not
  - and
  - or
  - assignment

# Precedence

- Consider the following expressions:

```
w = 3 + 4 * 5 < 3 * 4 + 5 or 1 / 2 != 0
```

```
a = False
```

```
b = False
```

```
c = True
```

```
x = 5
```

```
if a or b and c or 1 < x and x < 10:
```

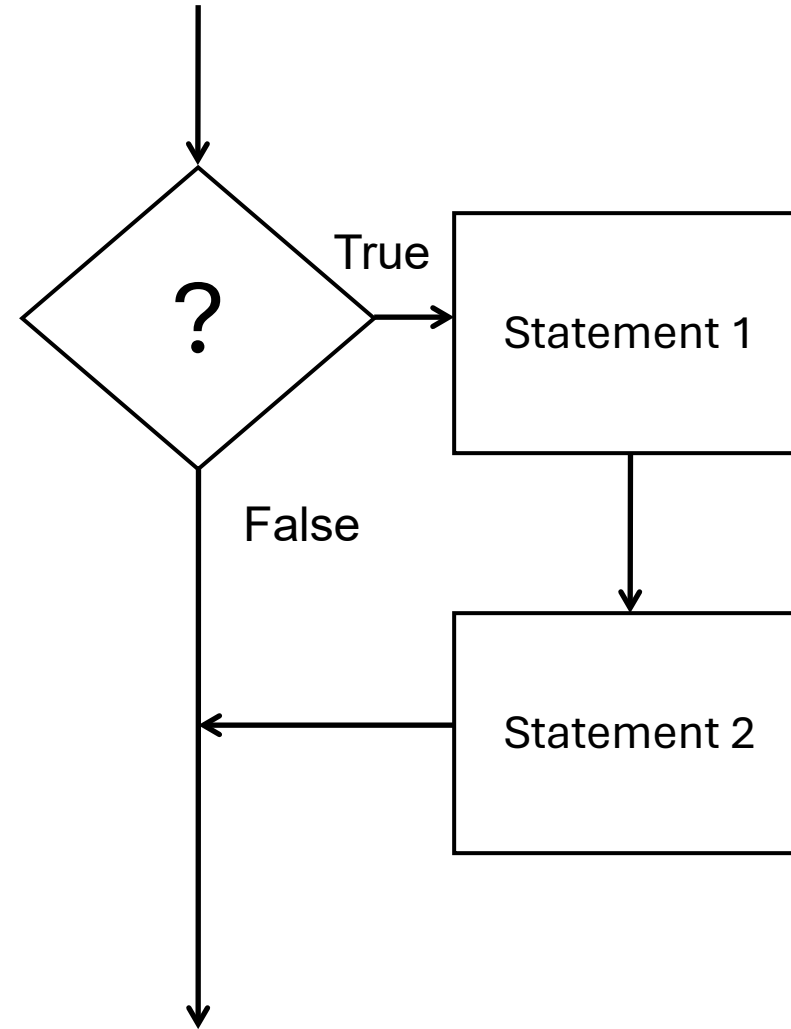
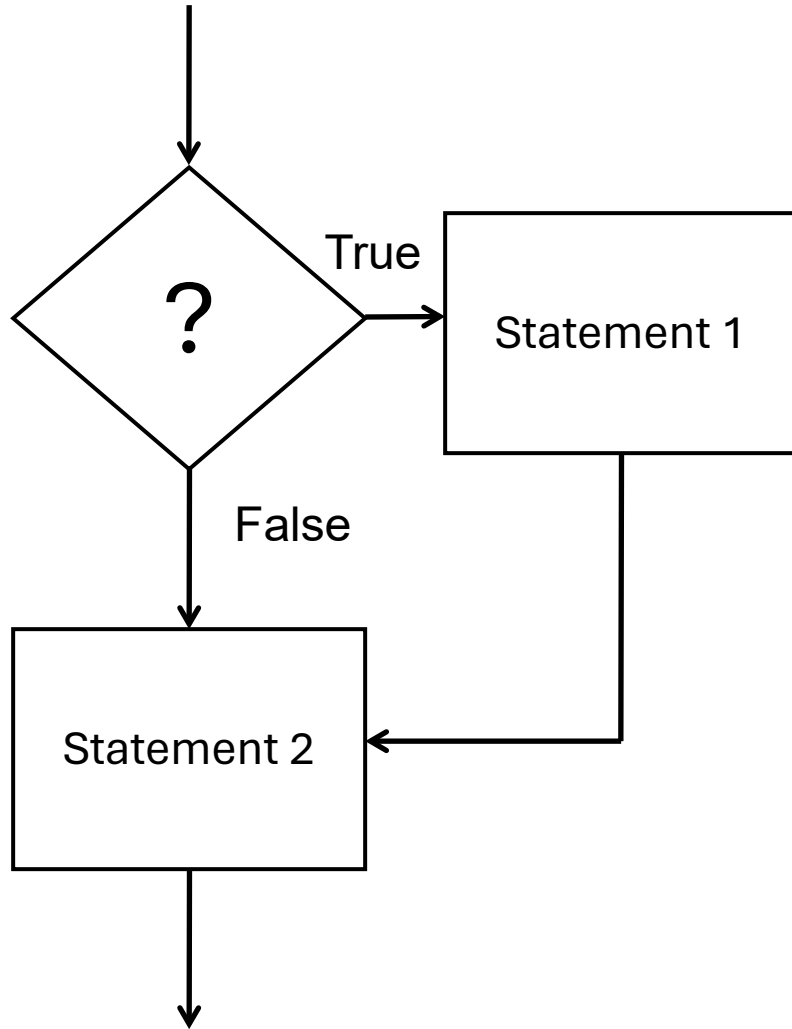
```
    print(x)
```



# If Statement Conditions

- Don't make the condition unnecessarily complex
  - `if x:` is equivalent to `if x == True:`
  - `if not x:` is equivalent to `if x == False:`

# Compound Statements



# Compound Statements

- The body of an if statement
  - May contain one statement
  - May contain many statements
- How do we know which statements are included in the body?
  - Body is determined by indenting
  - Body ends with the next line that is indented the same amount as the `if`

# Compound Statements

```
x = int(input())  
print("A")  
if x < 0:  
    print("B")  
    print("C")  
print("D")  
print("E")  
print("F")
```

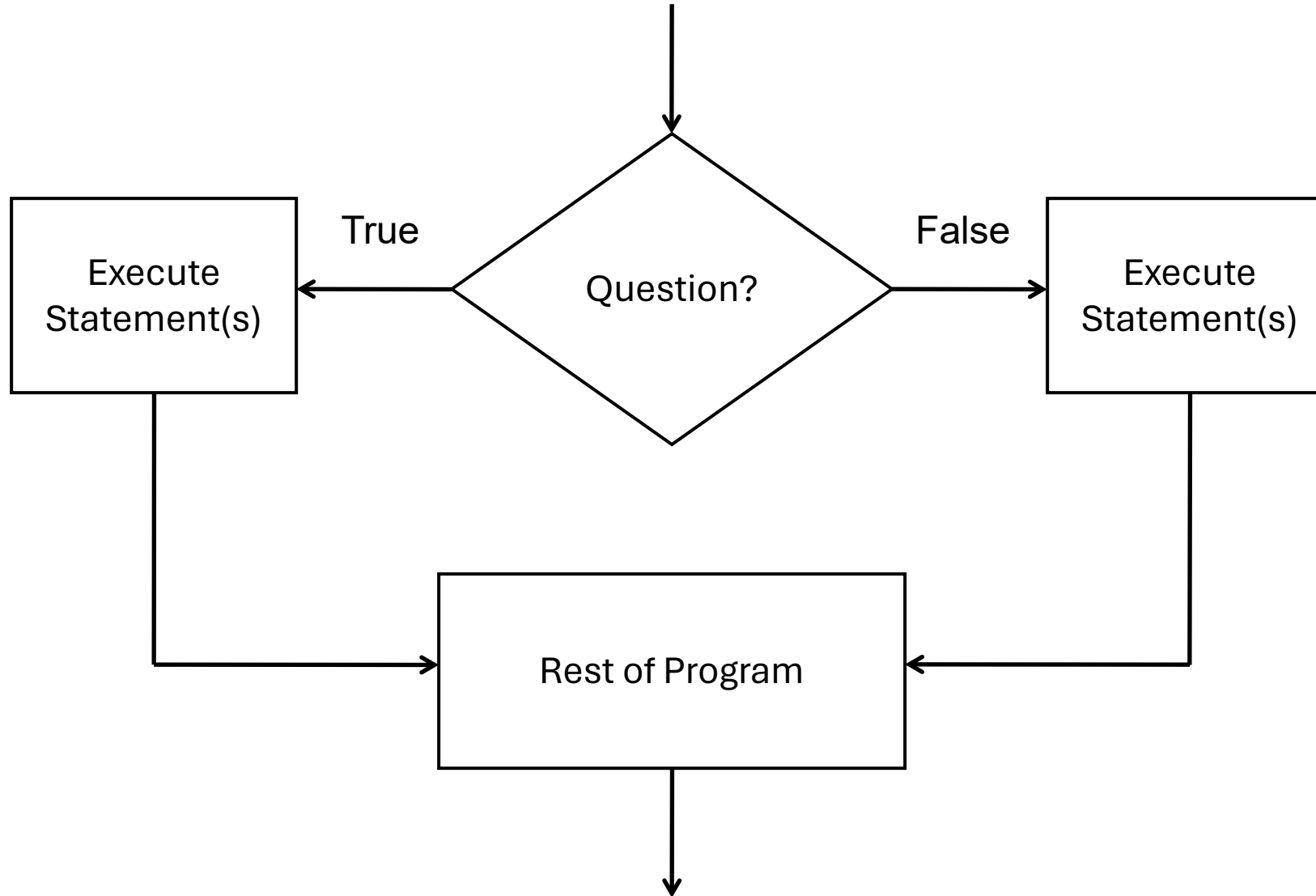
# If-Then-Else

- What if we have a condition
  - want to do something when the condition is true
  - want to do something else when the condition is false

```
if condition:  
    print("Doing something...")
```

```
if not condition:  
    print("Doing something else...")
```

# If-Then-Else



# Gold Example

# Nested If Statements

- An if statement can reside in the body of another if statement
  - How do we expand our program so that it handles all three states?
    - Gold is solid when the temperature less than 1064 degrees Celsius
    - Gold is liquid when the temperature is between 1064 and 2856 degrees Celsius
    - Otherwise gold is gaseous



# Gold Example

# If-Then-Elif-Else

- Allows exactly one of several options to execute
  - Conditions are tested sequentially until one evaluates to True
  - Body of the condition is executed
  - No further conditions are considered once a condition that evaluates to True is found

# Gold Example

# Multiple Elif Example

# Tax Example

- What if we want to write a program that calculates federal income tax
  - Tax payable is
    - 15% of income up to \$45,916
    - 20.5% of income from \$45,916 to \$91,831
    - 26% of income from \$91,831 to \$142,353
    - 29% of income from \$142,353 to \$202,800
    - 33% of income above \$202,800

# Tax Example

# Testing

- The process of executing a program in an attempt to locate bugs
  - How many times do we need to run the program?
  - What can't testing do?

# Testing

- Black-box testing
  - Test the program without looking at the source code
  - Tests are generally functional / behavioural
- White-box testing
  - Design test cases for the program by looking at its source code
  - Tests are generally structural



# White Box Test Coverage

- How thoroughly do the cases test the code?
  - Condition Coverage: Every decision point in the program is executed
  - Statement Coverage: Every statement in the program is executed
  - Path Coverage: Every possible path through the program is executed

# Testing Example

# The Dangers of Floating Point Numbers

- Floating point numbers approximate real numbers
  - Can cause problems when testing for equality

# Wrapping Up

- Three kinds of decision statements
  - If statement
  - If-Else statement
  - If-Elif-...-Elif-Else statement
- Each makes it possible to change the flow of control through the program

# Wrapping Up

- More complex control flow requires
  - Additional design
  - Additional testing
    - Black box
    - White box

# Where Are We Going?

- What if we want to do something several times?
  - A fixed number of times?
  - A number of times entered by the user?
  - Keep doing something until a specific event occurs?
- Next Up: Repetition