[301] Conditions

Tyler Caraza-Harter

Learning Objectives Today

Reason about conditions
- Conditional execution
- Alternate execution
- Chained execution
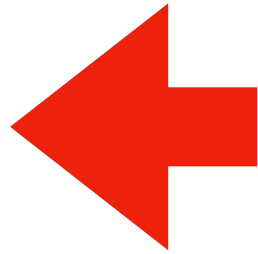- Nested conditions

Understand code blocks
- Be able to identify the lines of code in the same block

Sanity checking
- Recognize errors
- Sanitize bad data automatically

Today's Outline

Review ⬅

Control Flow Diagrams

Basic syntax for "if"

Identifying code blocks

Demos

Indentation Example

```
print("A")
print("B")

def print_letters():
    print("C")
    print("D")

print("E")
print("F")

print_letters()
```

Indentation Example

```
print("A")
print("B")

def print_letters():
    print("C")
    print("D")

print("E")
print("F")

print_letters()
```

A
B
E
F
C
D

Indentation Example

```
print("A")
print("B")

def print_letters():
    print("C")
    print("D")

print("E")
print("F")

print_letters()
```

**indented, so "inside"
print_letters function**

**A
B
E
F
C
D**

Indentation Example

```
print("A")
print("B")

def print_letters():
    print("C")
    print("D")

print("E")
print("F")

print_letters()
```

indented, so "inside" print_letters function

printed last because print_letters is called last

A
B
E
F
C
D

Indentation Example

```
print("A")
print("B")

def print_letters():
    print("C")
    print("D")

print("E")
print("F")

print_letters()
```

indented, so "inside" print_letters function

A
B
E
F
C
D

Indentation Example

```
print("A")
print("B")
```

not indented, so "outside" any function

```
def print_letters():
    print("C")
    print("D")
```

indented, so "inside" print_letters function

```
print("E")
print("F")

print_letters()
```

what does it print?

A
B
E
F
C
D

Indentation Example

```
print("A")
print("B")
```
not indented, so "outside" any function

```
def print_letters():
    print("C")
    print("D")
```
indented, so "inside" print_letters function

```
print("E")
print("F")
```
also not indented, so "outside" any function. Runs BEFORE print_letters is called

```
print_letters()
```

what does it print?

A
B
E
F
C
D

Indentation Example

```
print("A")
print("B")
```
not indented, so **"outside"** any function

```
def print_letters():
    print("C")
    print("D")
```
indented, so **"inside"** print_letters function

```
print("E")
print("F")
```
also not indented, so **"outside"** any function. Runs **BEFORE** print_letters is called

```
print_letters()
```

A
B
E
F
C
D

We use **indenting** to tell Python which code is **inside** or **outside** of a function (or other things we'll learn about soon).

Indentation Example

```python
print("A")
print("B")
```
not indented, so "outside" any function

```python
def print_letters():
    print("C")
    print("D")
```
indented, so "inside" print_letters function

blank lines are **irrelevant**

```python
print("E")
print("F")
```
also not indented, so "outside" any function. Runs BEFORE print_letters is called

```python
print_letters()
```

what does it print?

**A
B
E
F
C
D**

We use **indenting** to tell Python which code is **inside** or **outside** of a function (or other things we'll learn about soon).

Indentation Example

```
print("A")
print("B")

def print_letters():
    print("C")
    print("D")

print("E")
print("F")

print_letters()
```

we'll often call the lines of code **inside** something a **"block"** of code

A

B

E

F

C

D

Today's Outline
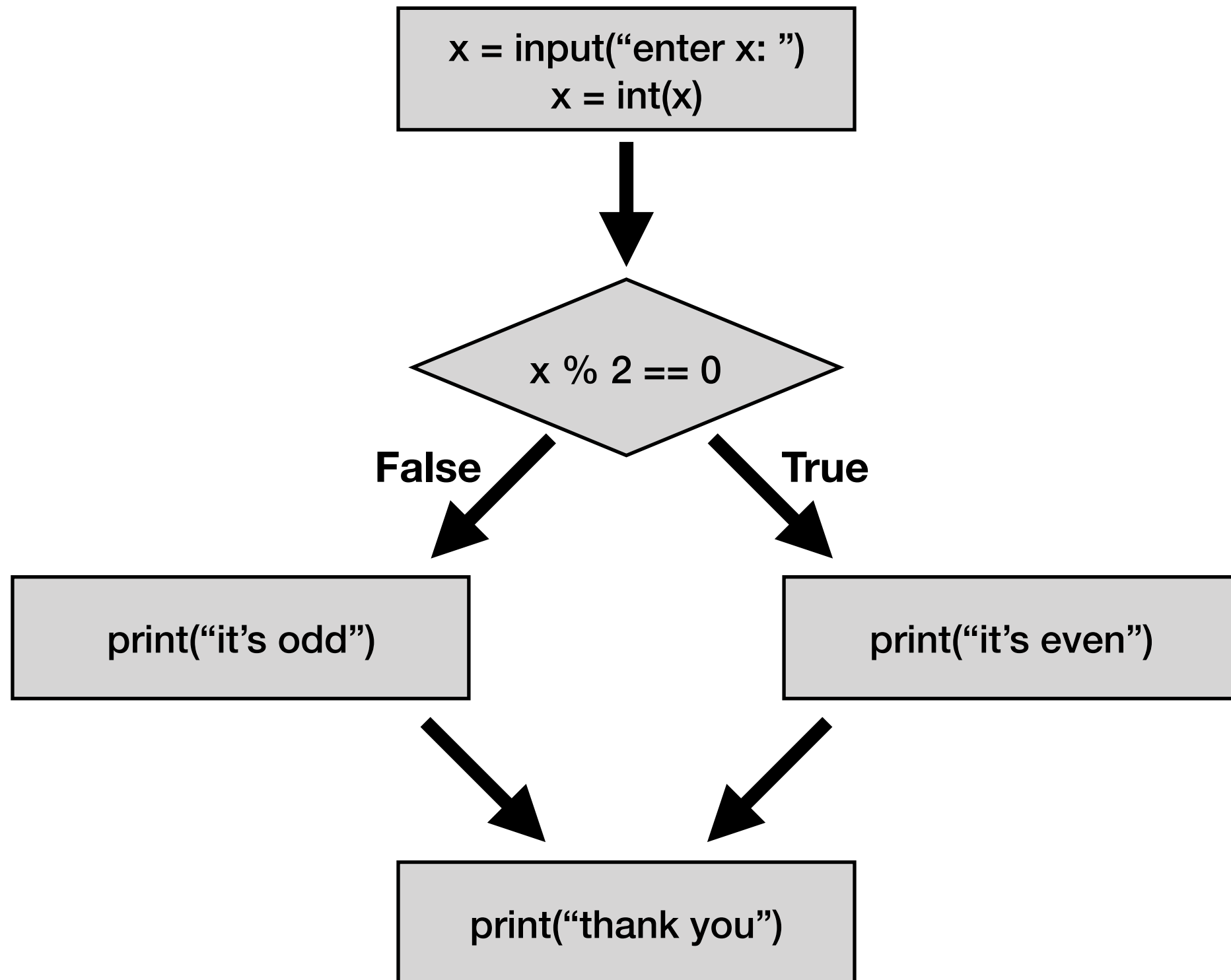
Review
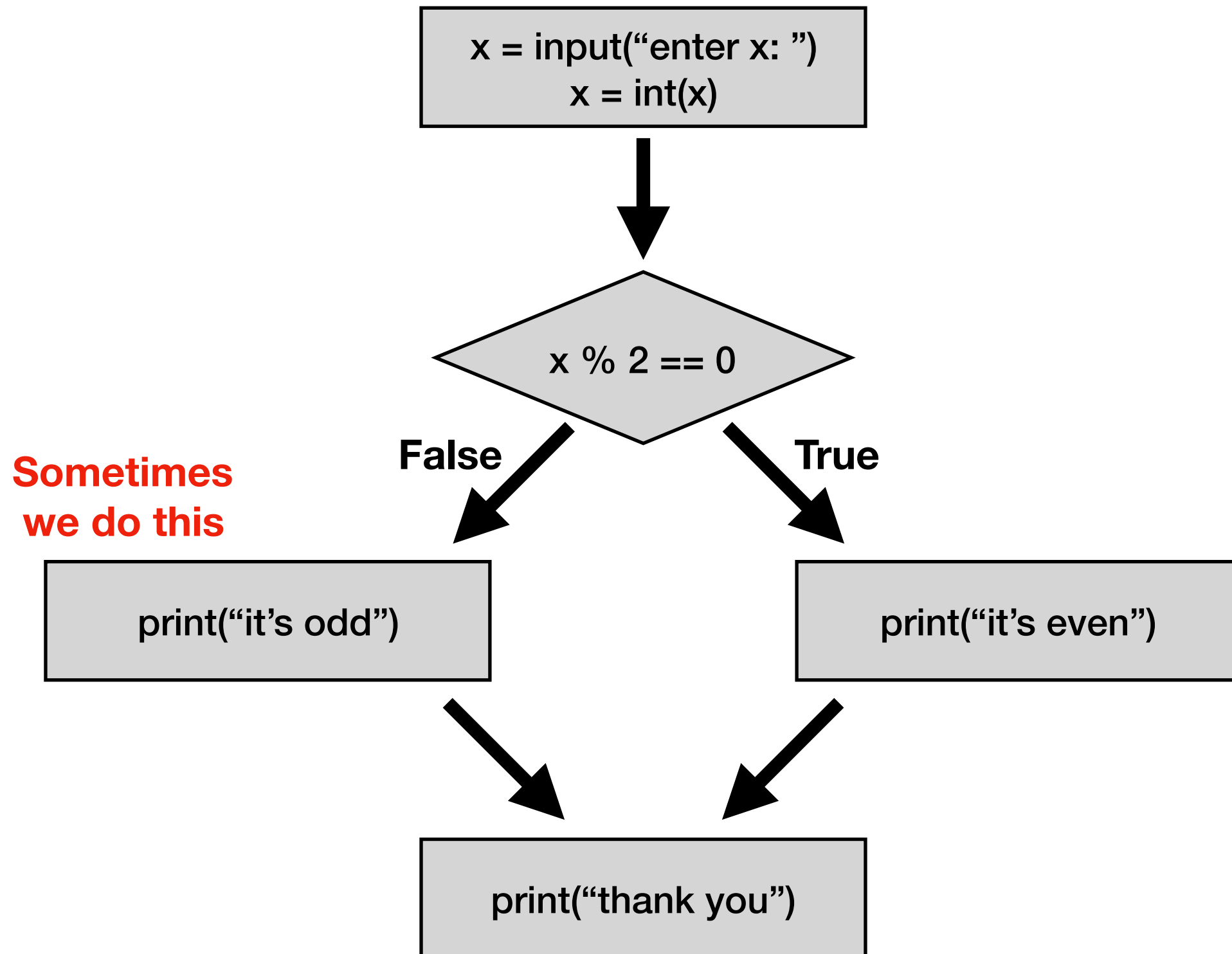
Control Flow Diagrams ⬅
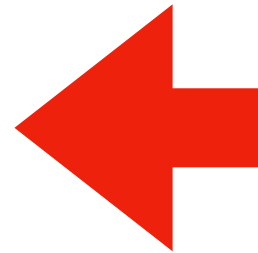
Basic syntax for "if"

Identifying code blocks

Demos

Control Flow Diagrams

Control Flow Diagrams

```
x = input("enter x: ")
x = int(x)
```

```
x % 2 == 0
```

False

True

Sometimes we do this

```
print("it's odd")
```
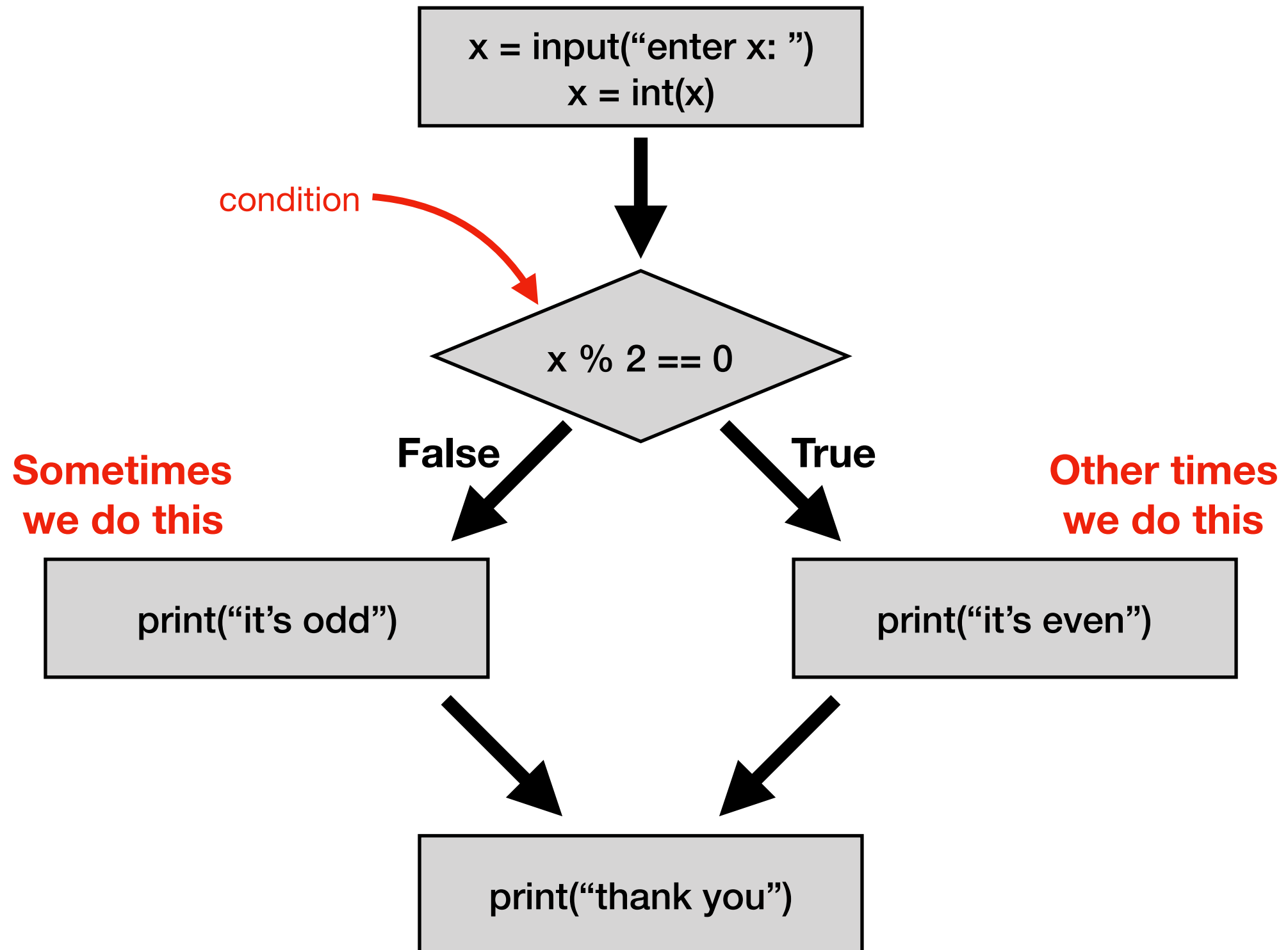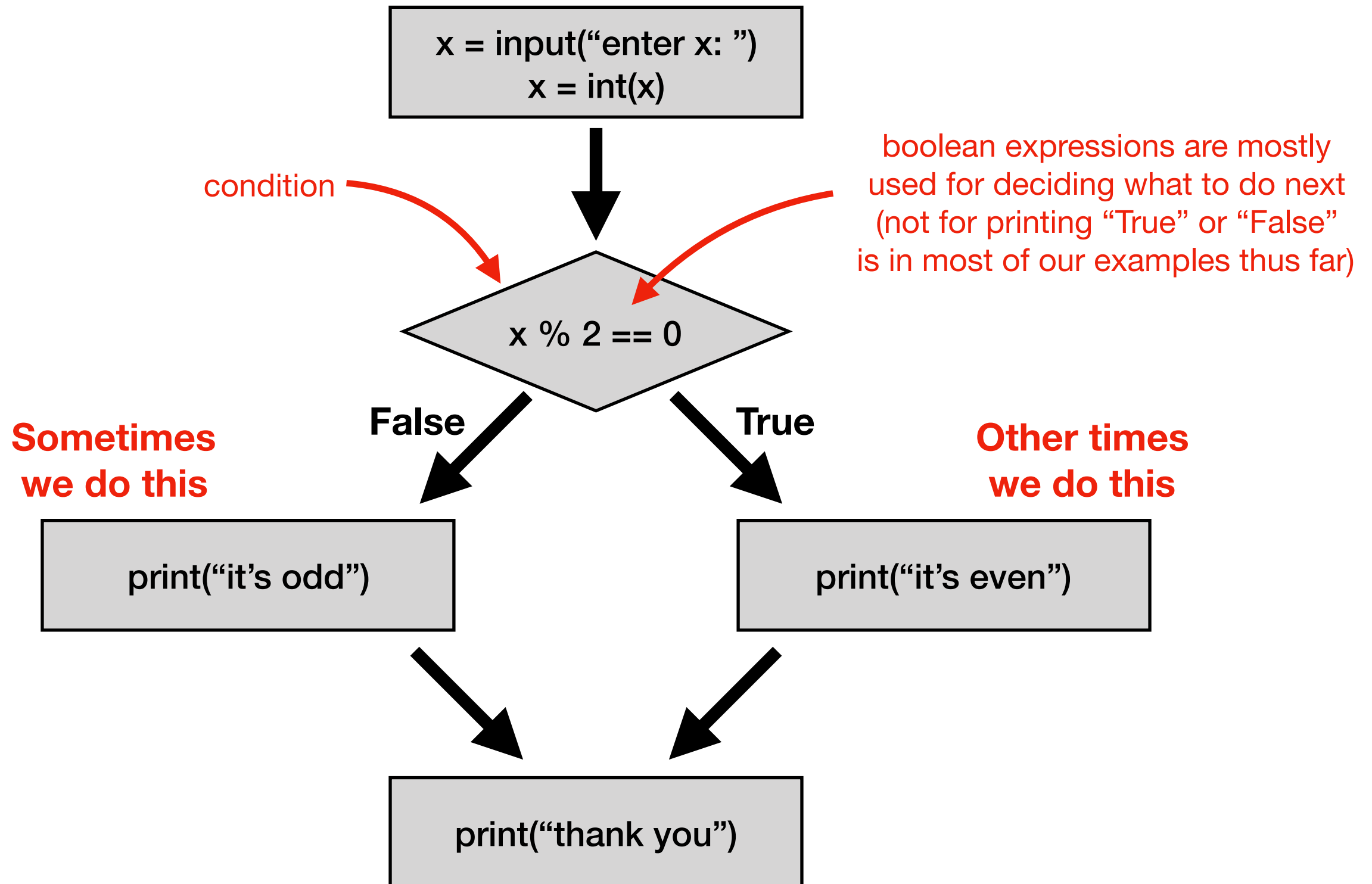
```
print("it's even")
```

```
print("thank you")
```

Control Flow Diagrams
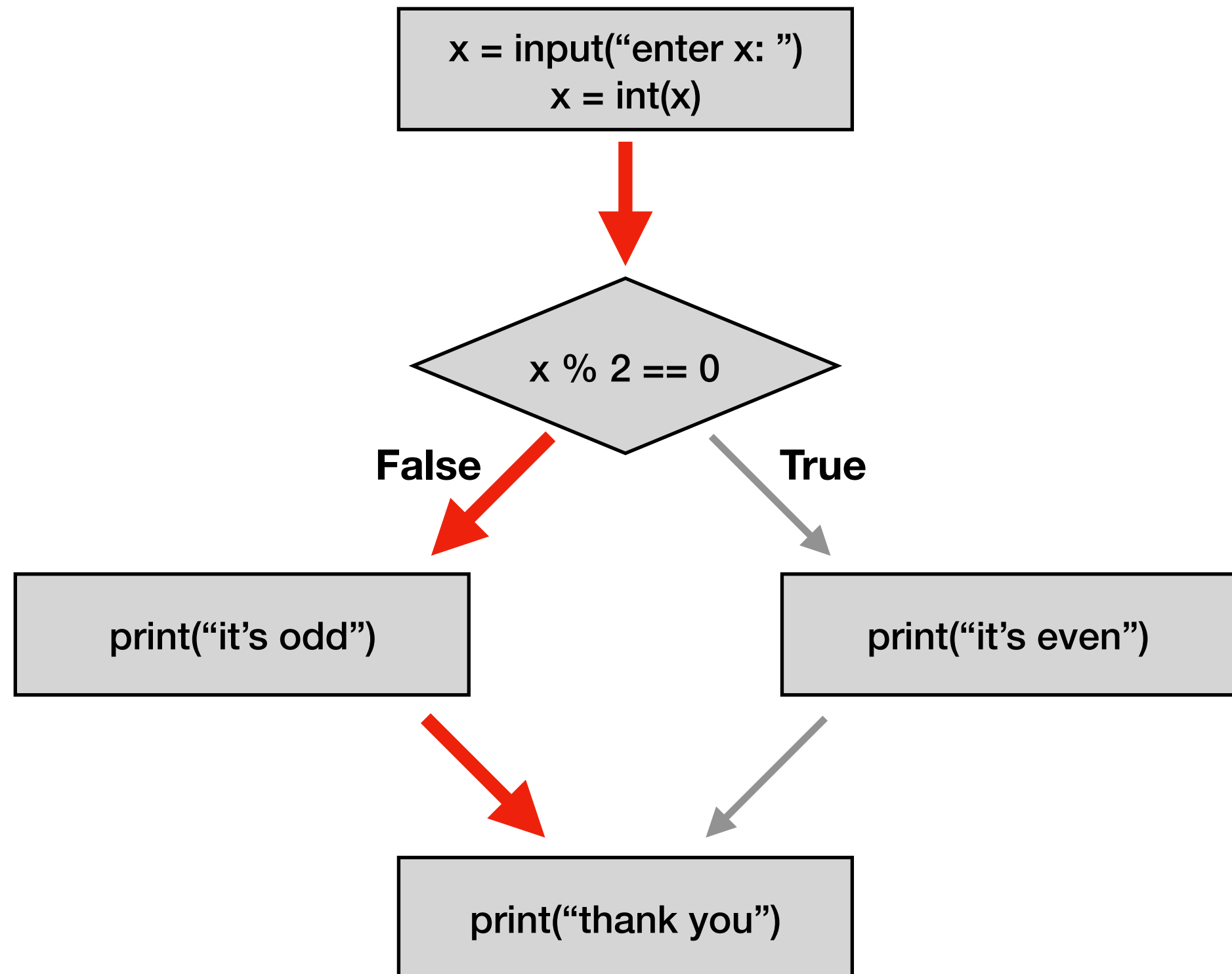
Control Flow Diagrams

Control Flow Diagrams

x = input("enter x: ")
x = int(x)

condition

boolean expressions are mostly used for deciding what to do next (not for printing "True" or "False" is in most of our examples thus far)

x % 2 == 0

False

True

Sometimes we do this

Other times we do this

print("it's odd")

print("it's even")

print("thank you")
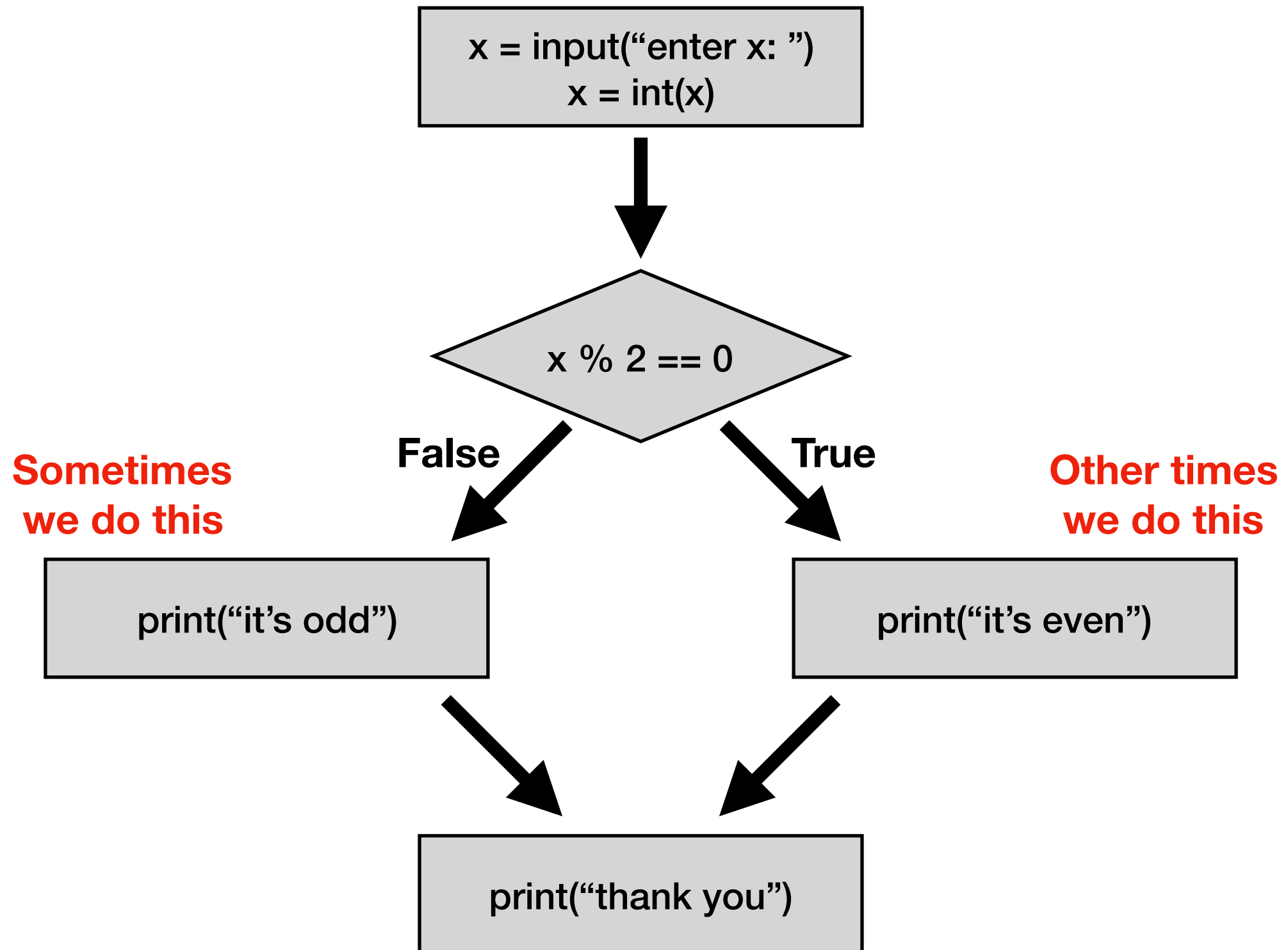
"Paths of Execution"

Input/Output:

```
enter x: 7
it's odd
thank you
```

x = input("enter x: ")
x = int(x)

x % 2 == 0

False   **True**

print("it's odd")   print("it's even")

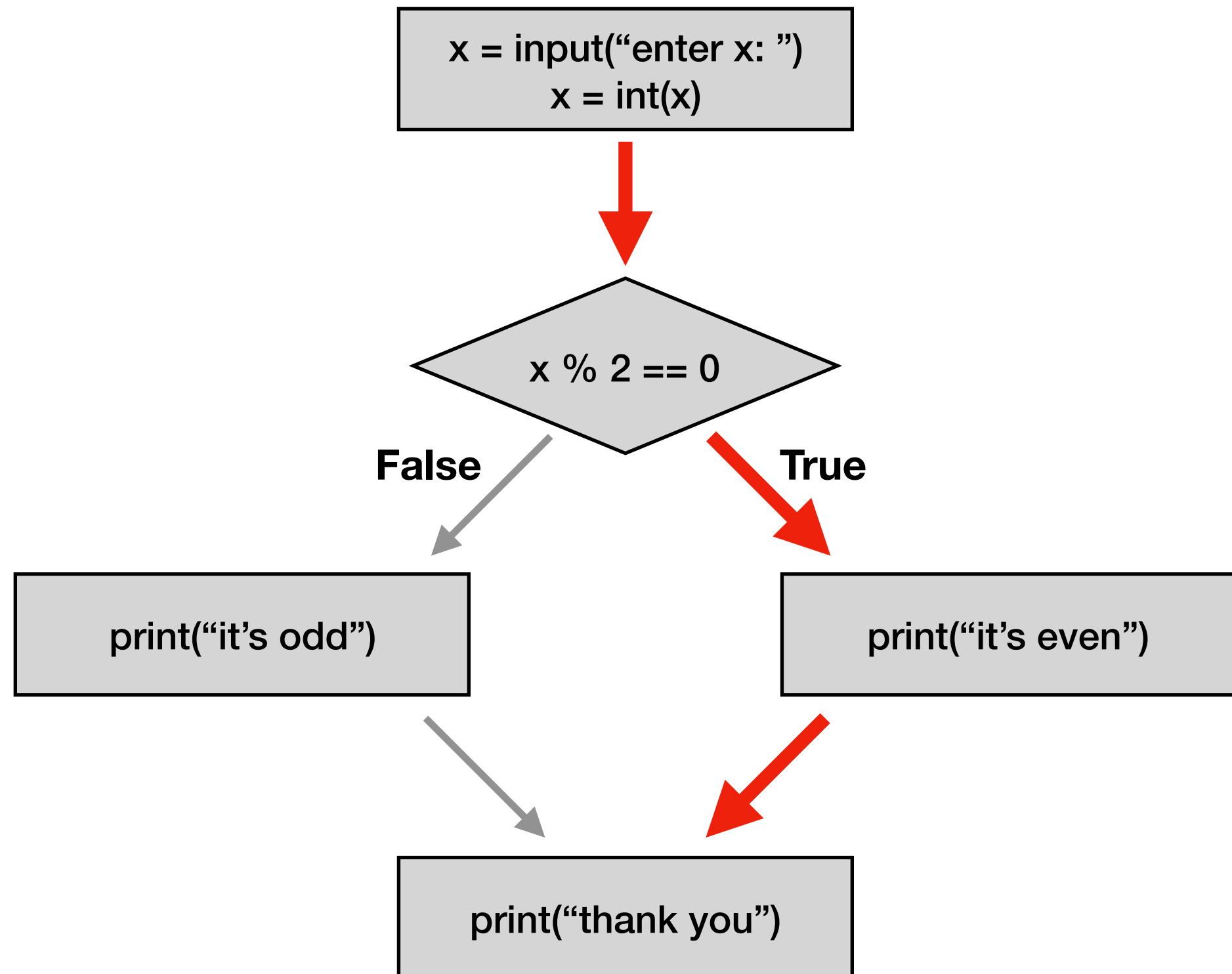print("thank you")

"Paths of Execution"
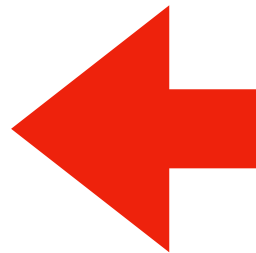
Input/Output:

```
enter x: 8
it's even
thank you
```

Today's Outline

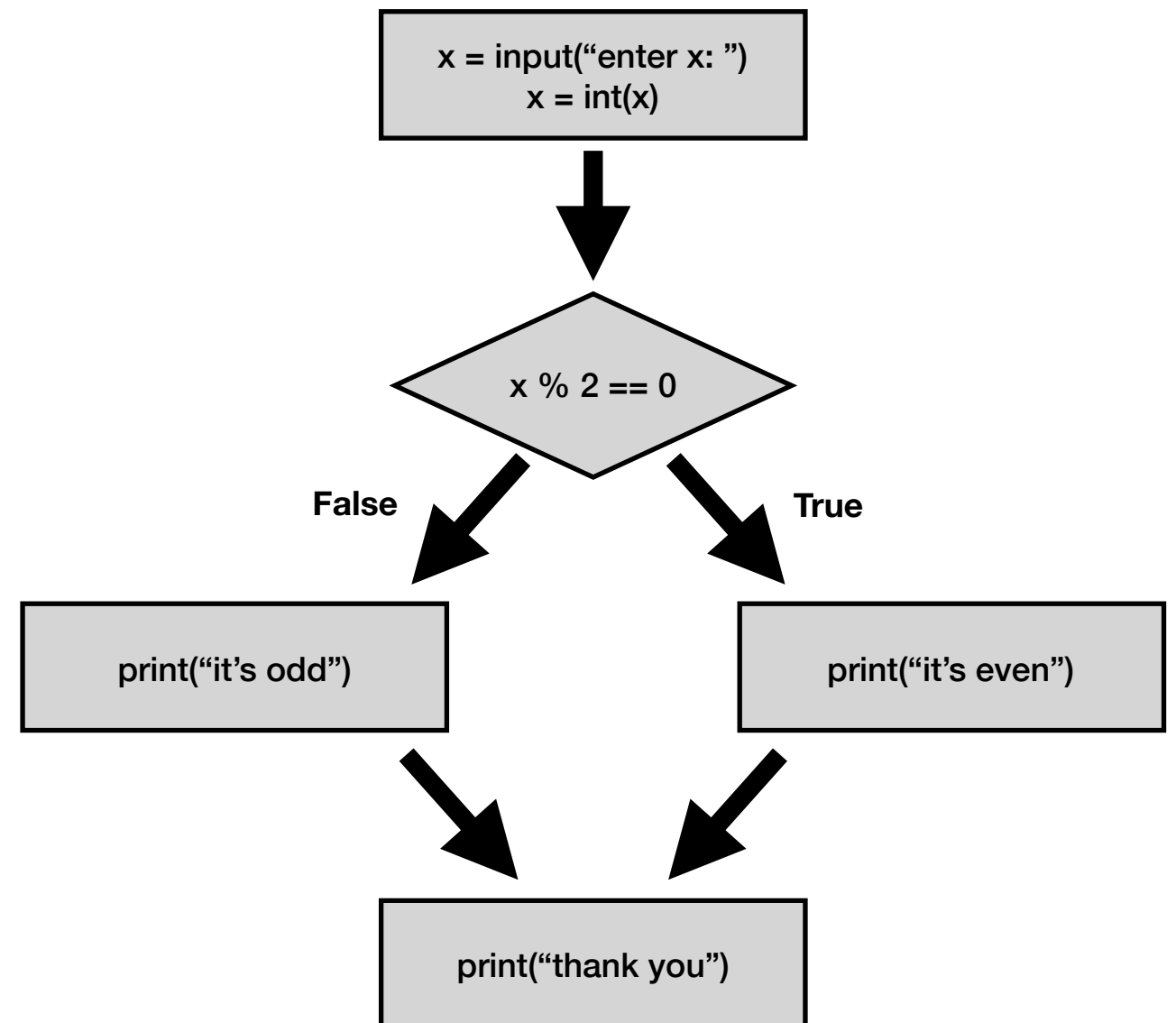Review

Control Flow Diagrams

Basic syntax for "if" ⬅

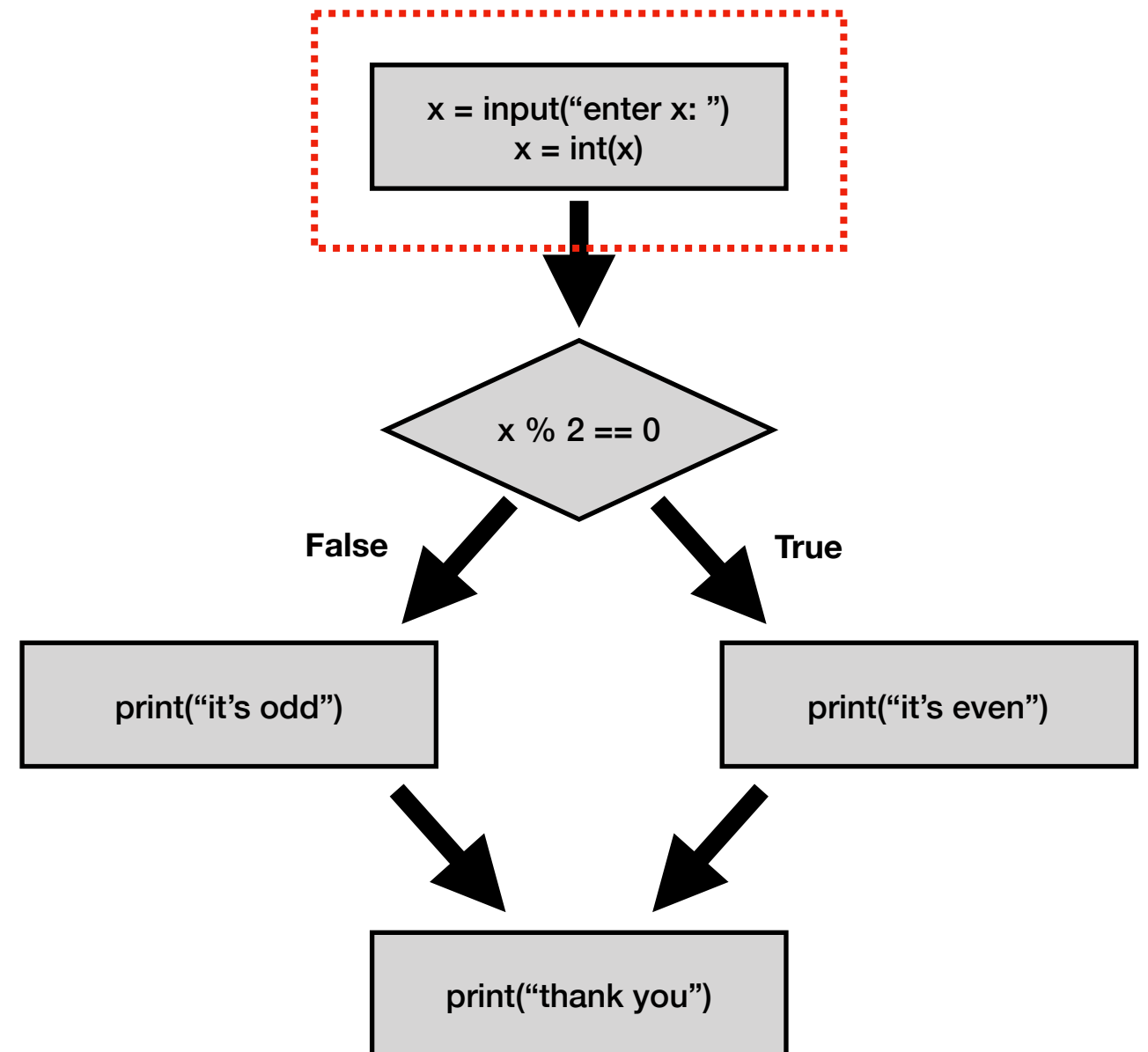Identifying code blocks

Demos

Writing conditions in Python

Code:

x = input("enter x: ")
x = int(x)

x % 2 == 0

False

True

print("it's odd")

print("it's even")

print("thank you")

Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)
```
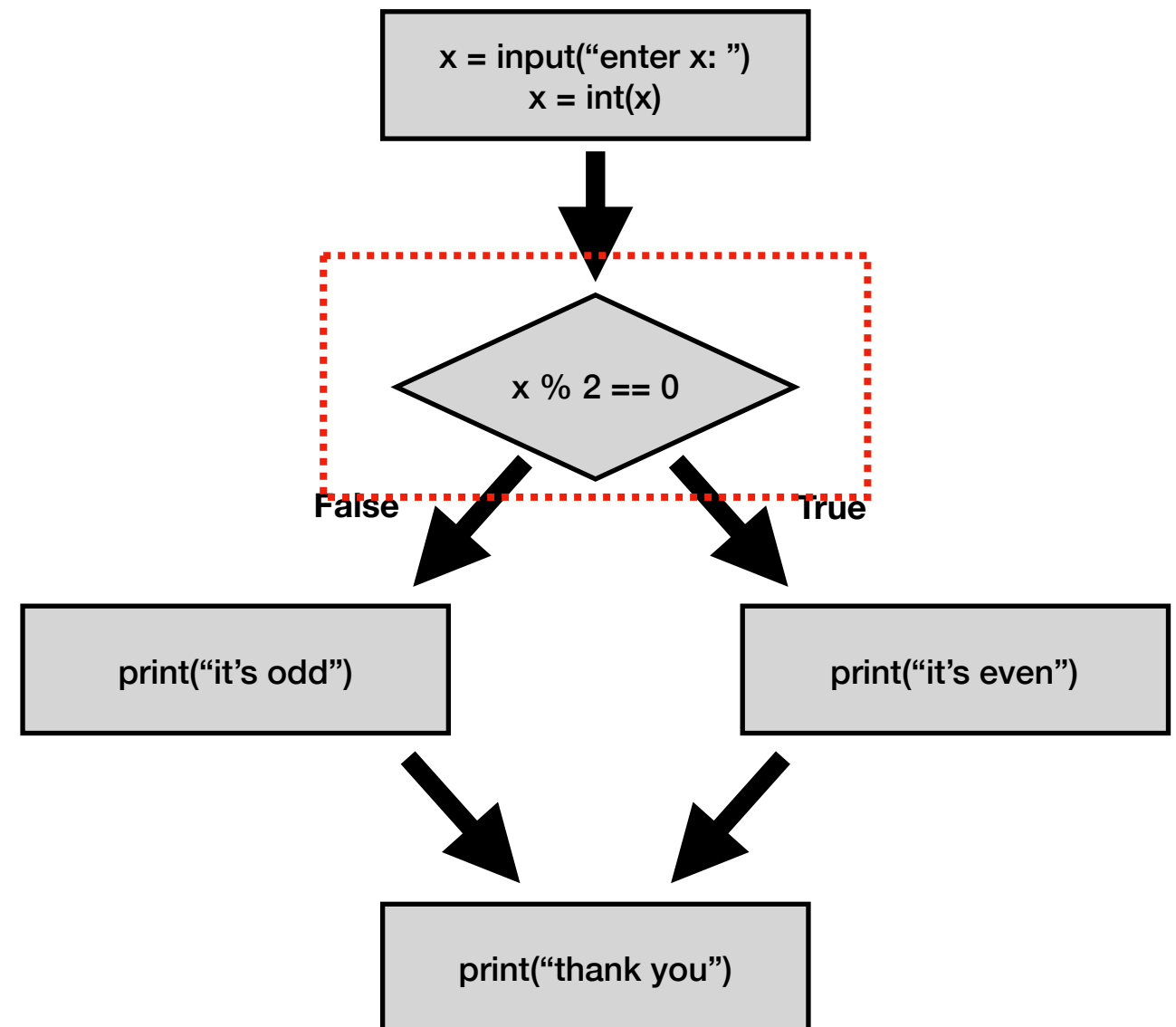
Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
```
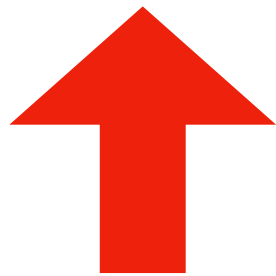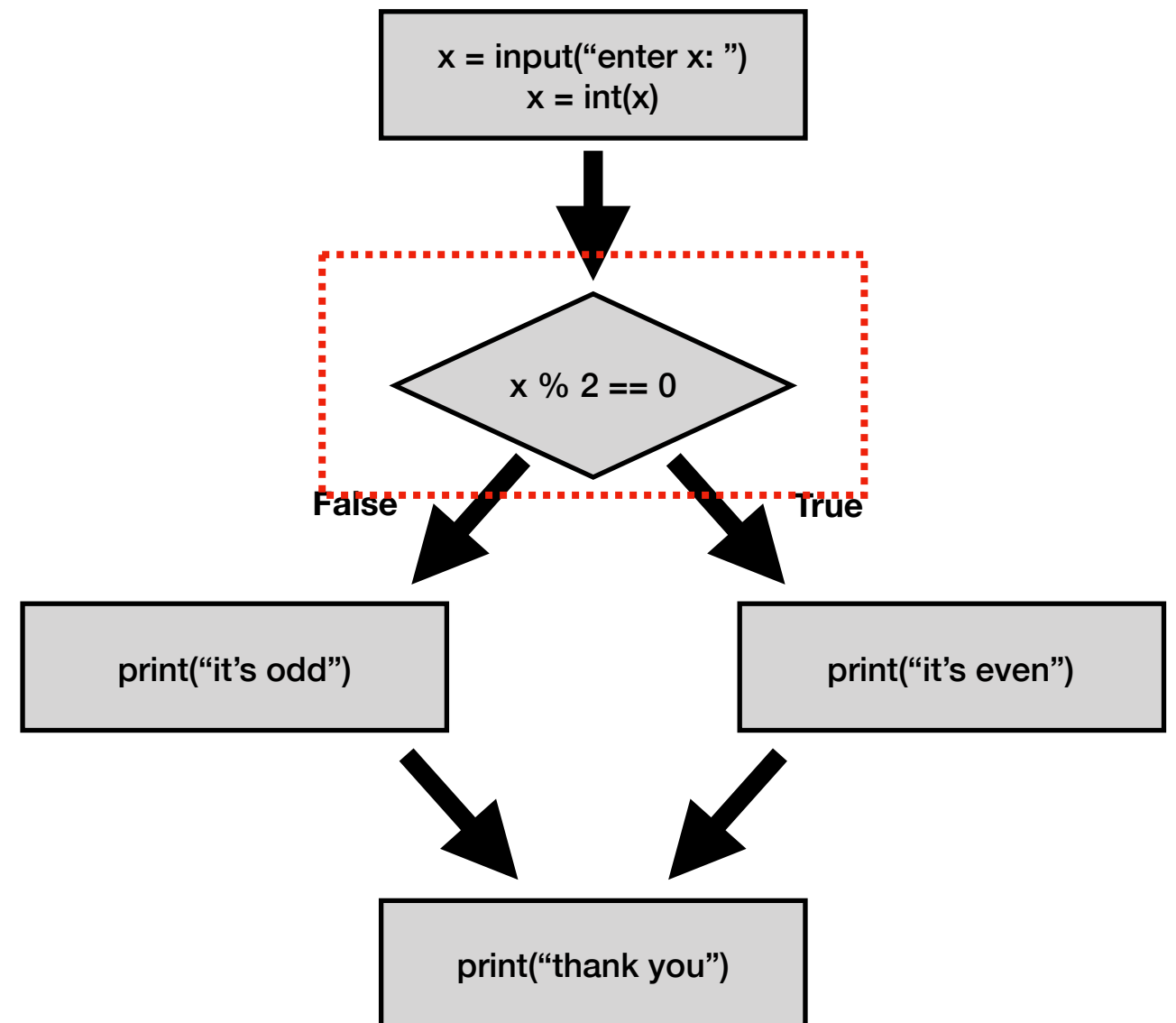
Writing conditions in Python

Code:

```
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
```

boolean expression

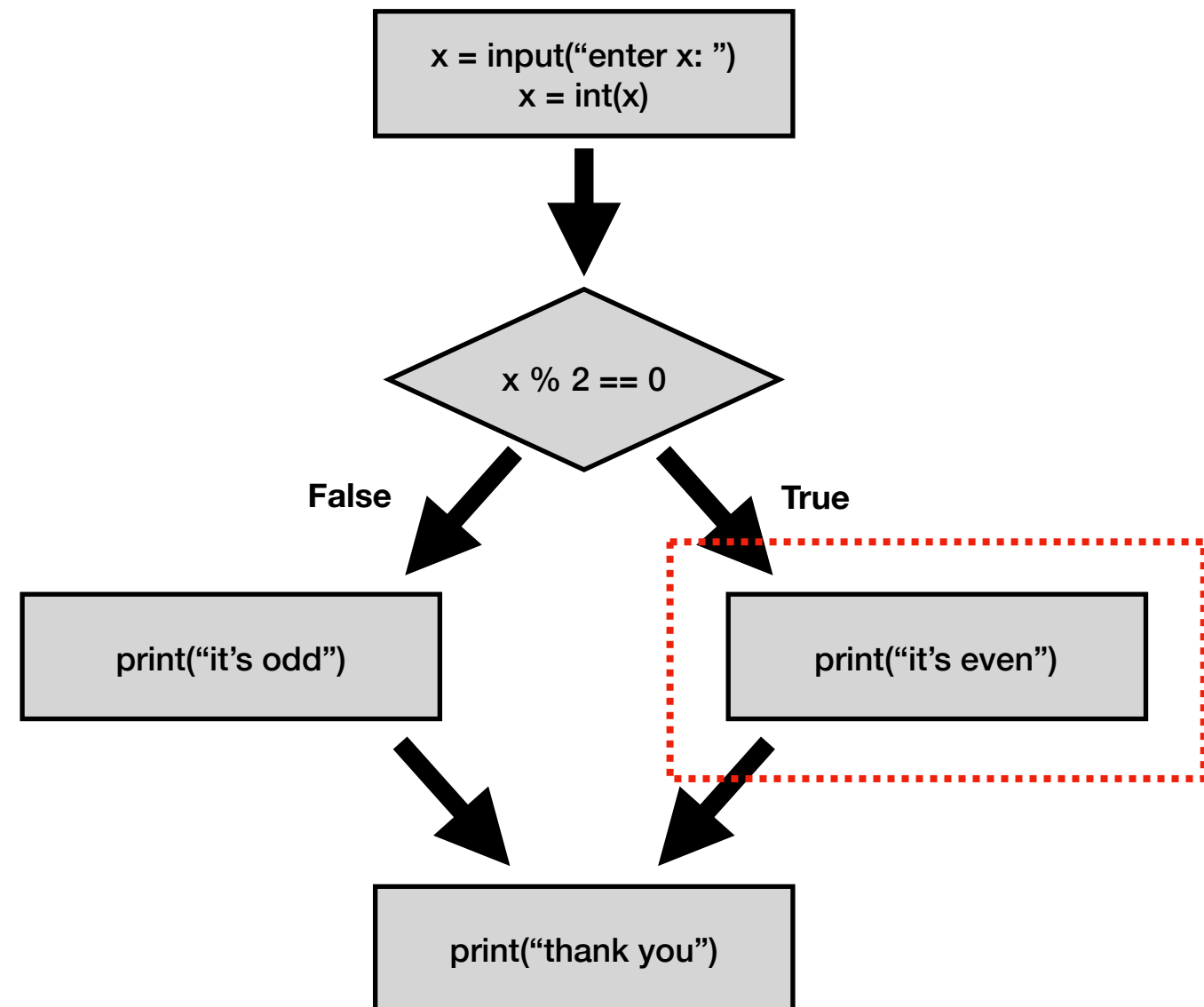# Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
```
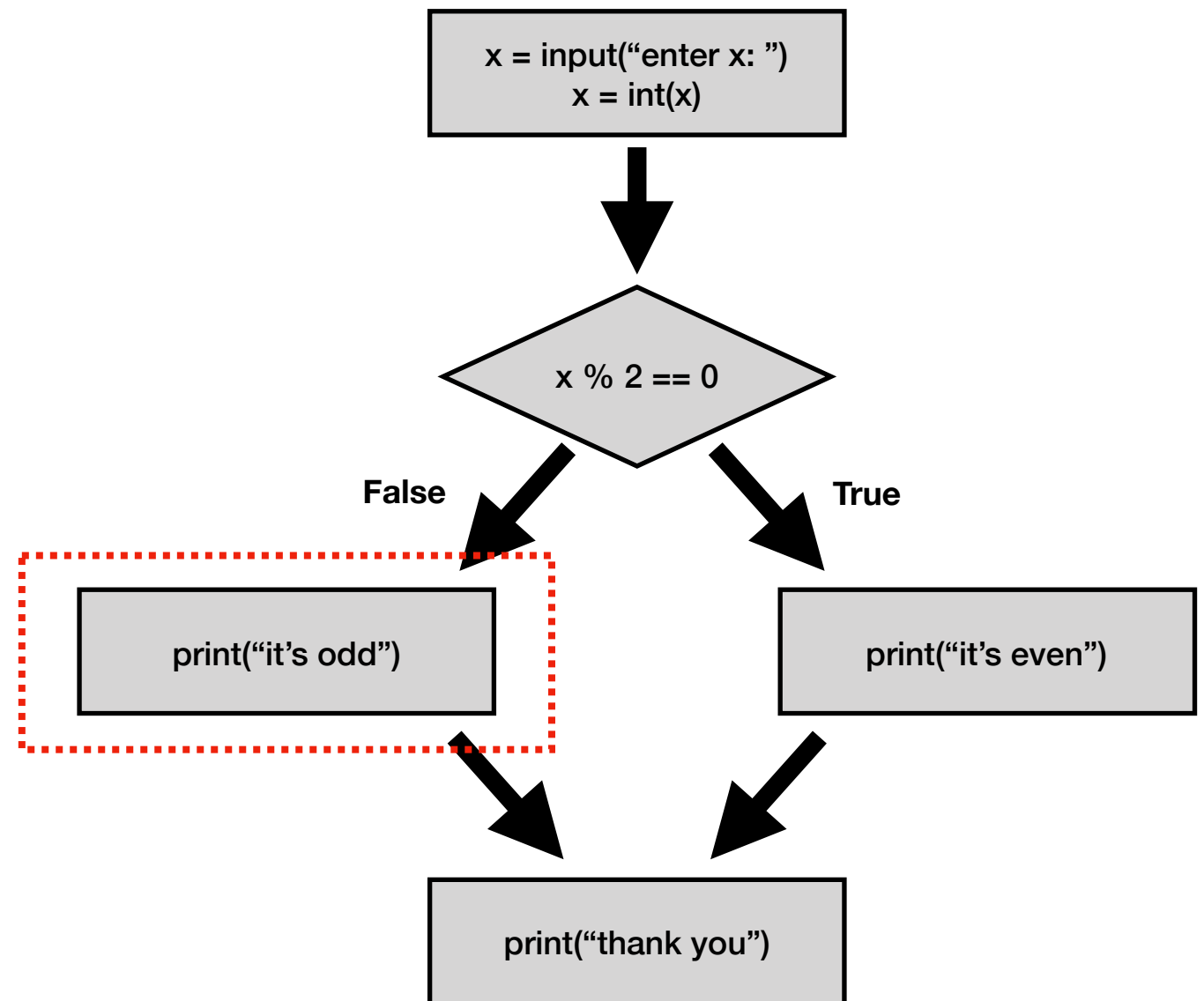
Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
else:
    print("it's odd")
```

Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
else:
    print("it's odd")

print("thank you")
```

Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
else:
    print("it's odd")

print("thank you")
```
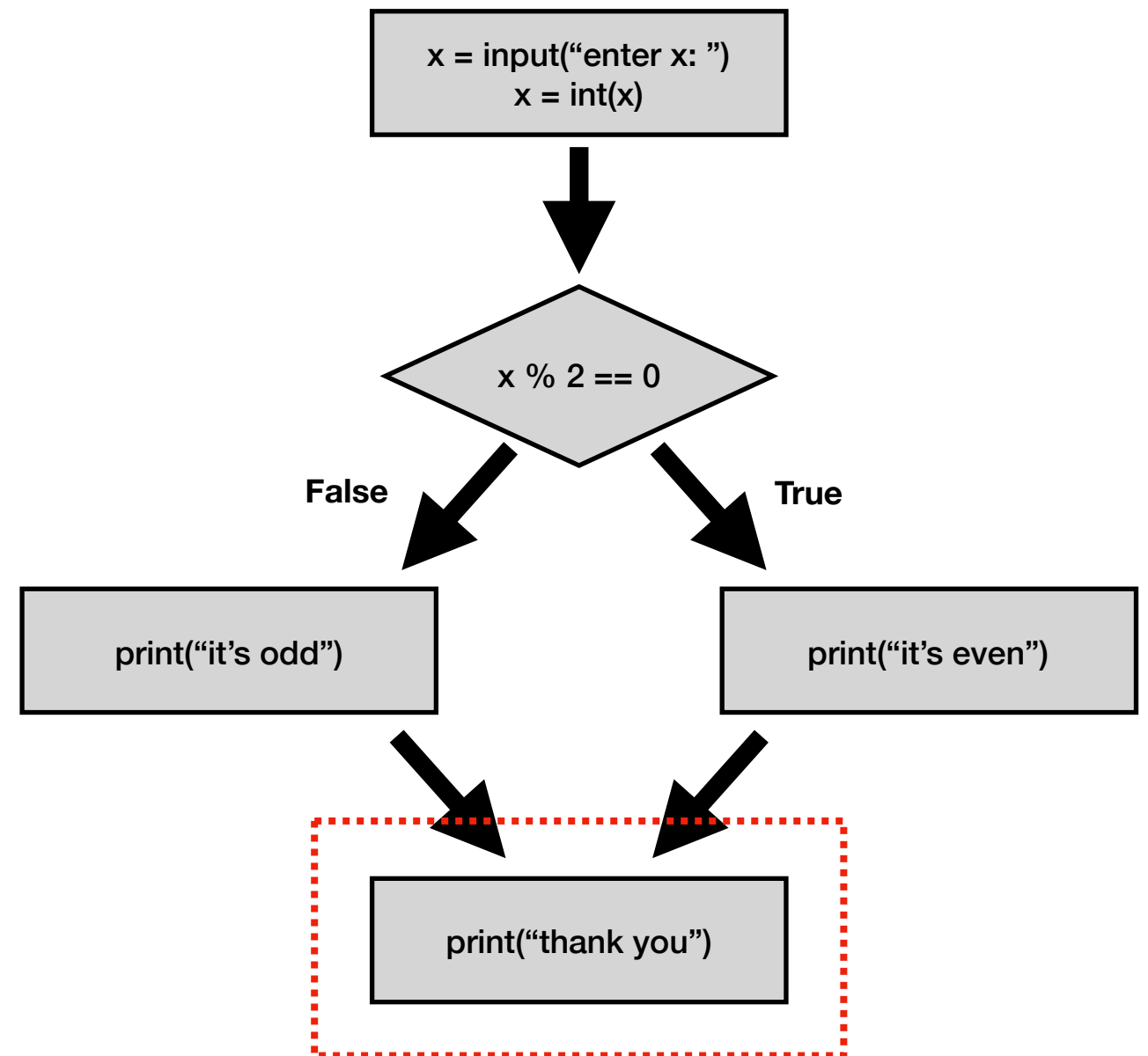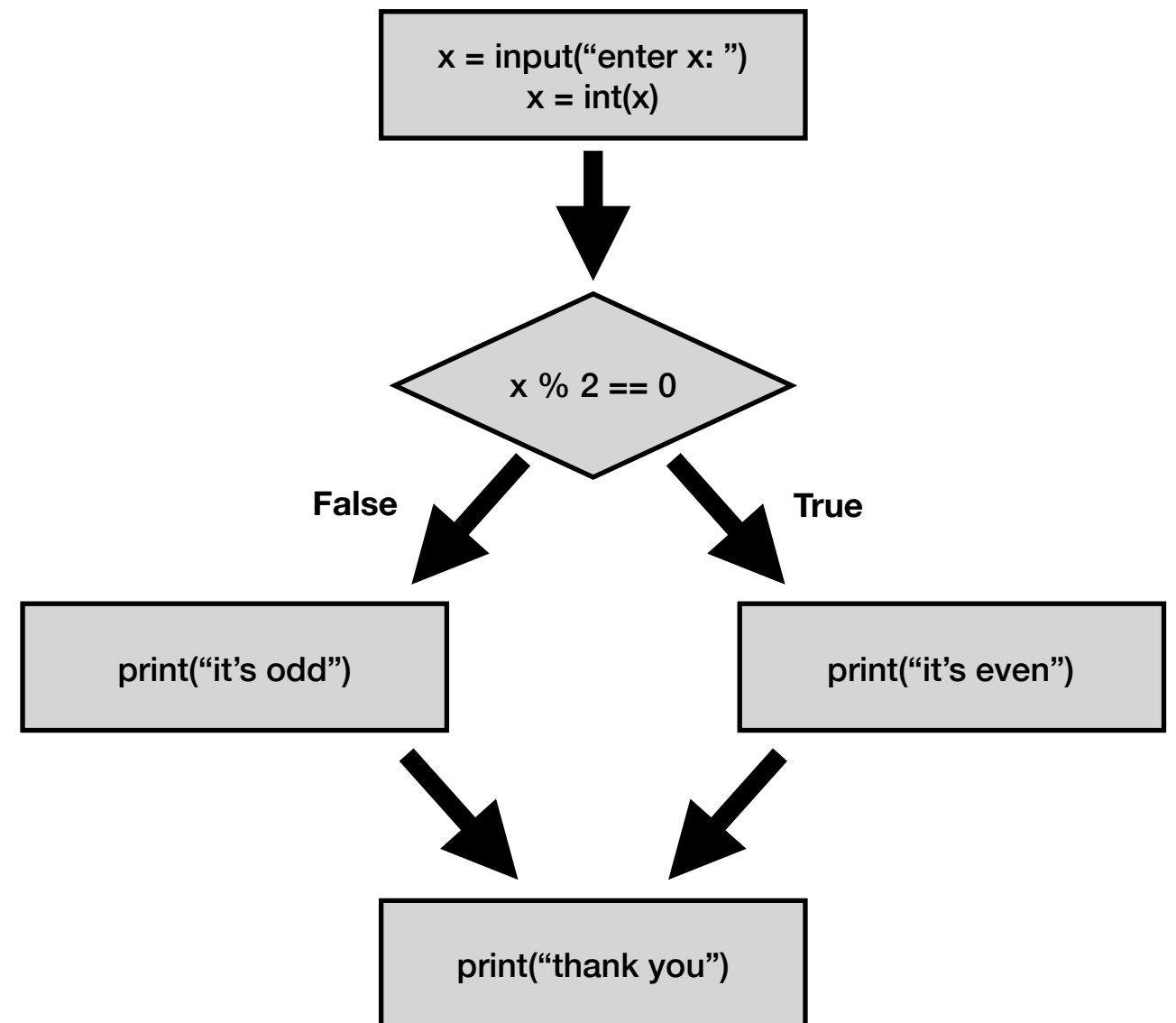
Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
else:
    print("it's odd")
    print("good!")

print("thank you")
```
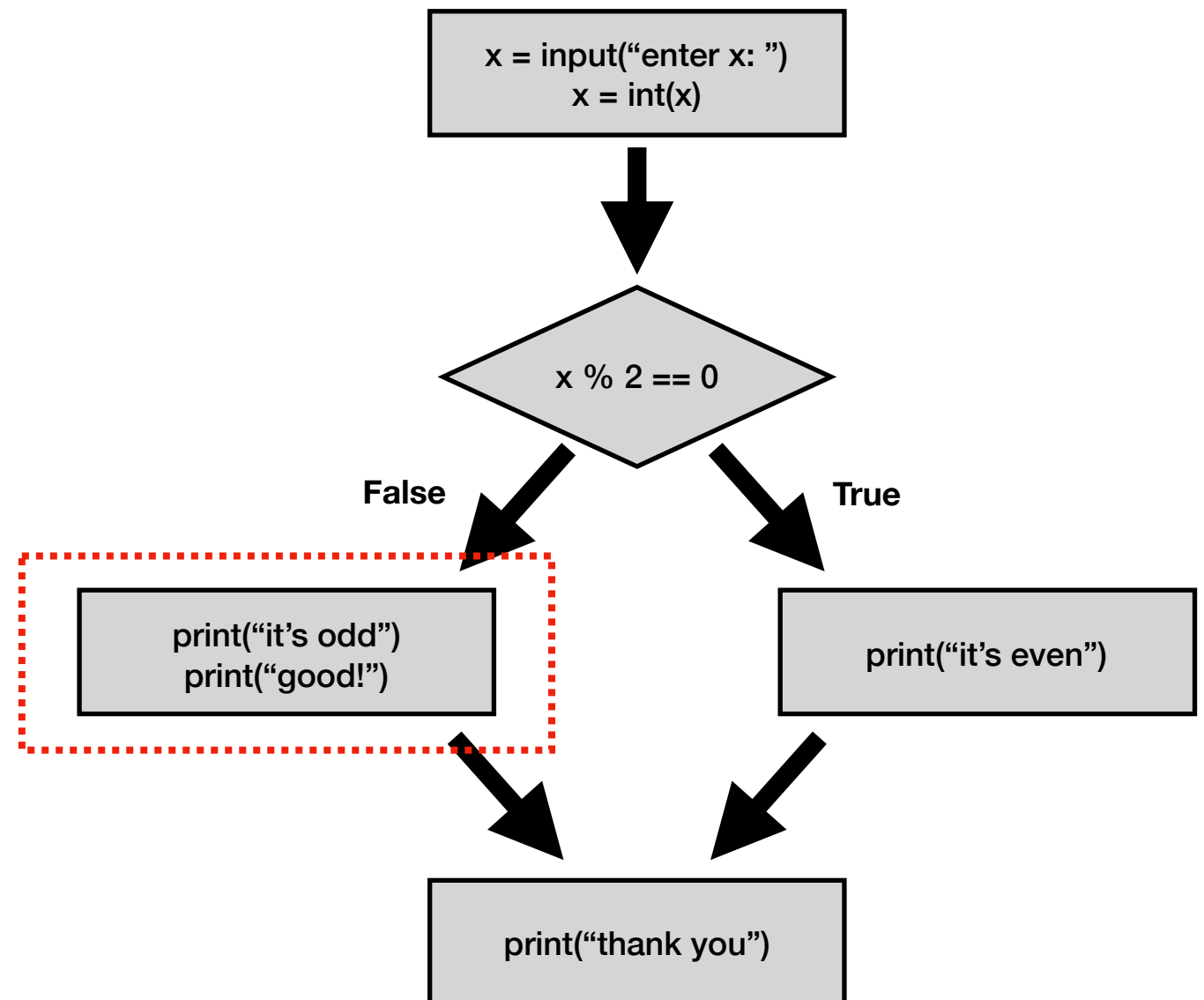
Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
    print("we wanted odd")
else:
    print("it's odd")
    print("good!")

print("thank you")
```
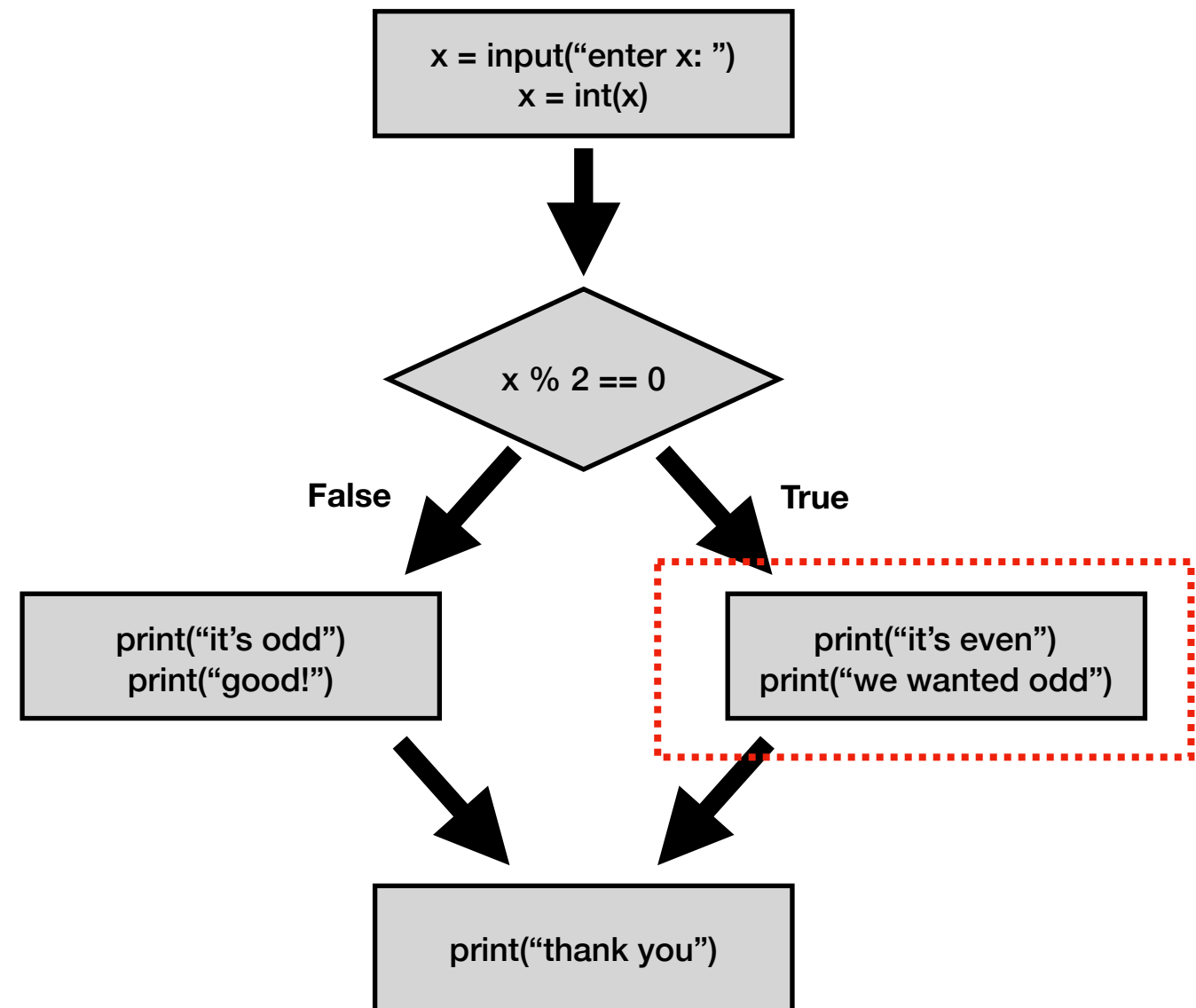
Writing conditions in Python

Code:

```python
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
    print("we wanted odd")
else:
    print("it's odd")
    print("good!")

print("thank you")
print("all done")
```

Today's Outline

Review

Control Flow Diagrams

Basic syntax for "if"

Identifying code blocks ⬅

Demos

Code Blocks

Code:

```
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
    print("we wanted odd")
else:
    print("it's odd")
    print("good!")

print("thank you")
print("all done")
```

Code Blocks

Code:

```
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
    print("we wanted odd")
else:
    print("it's odd")
    print("good!")

print("thank you")
print("all done")
```

**block of code
inside "if"**

Code Blocks

Code:

```
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
    print("we wanted odd")
else:
    print("it's odd")
    print("good!")

print("thank you")
print("all done")
```
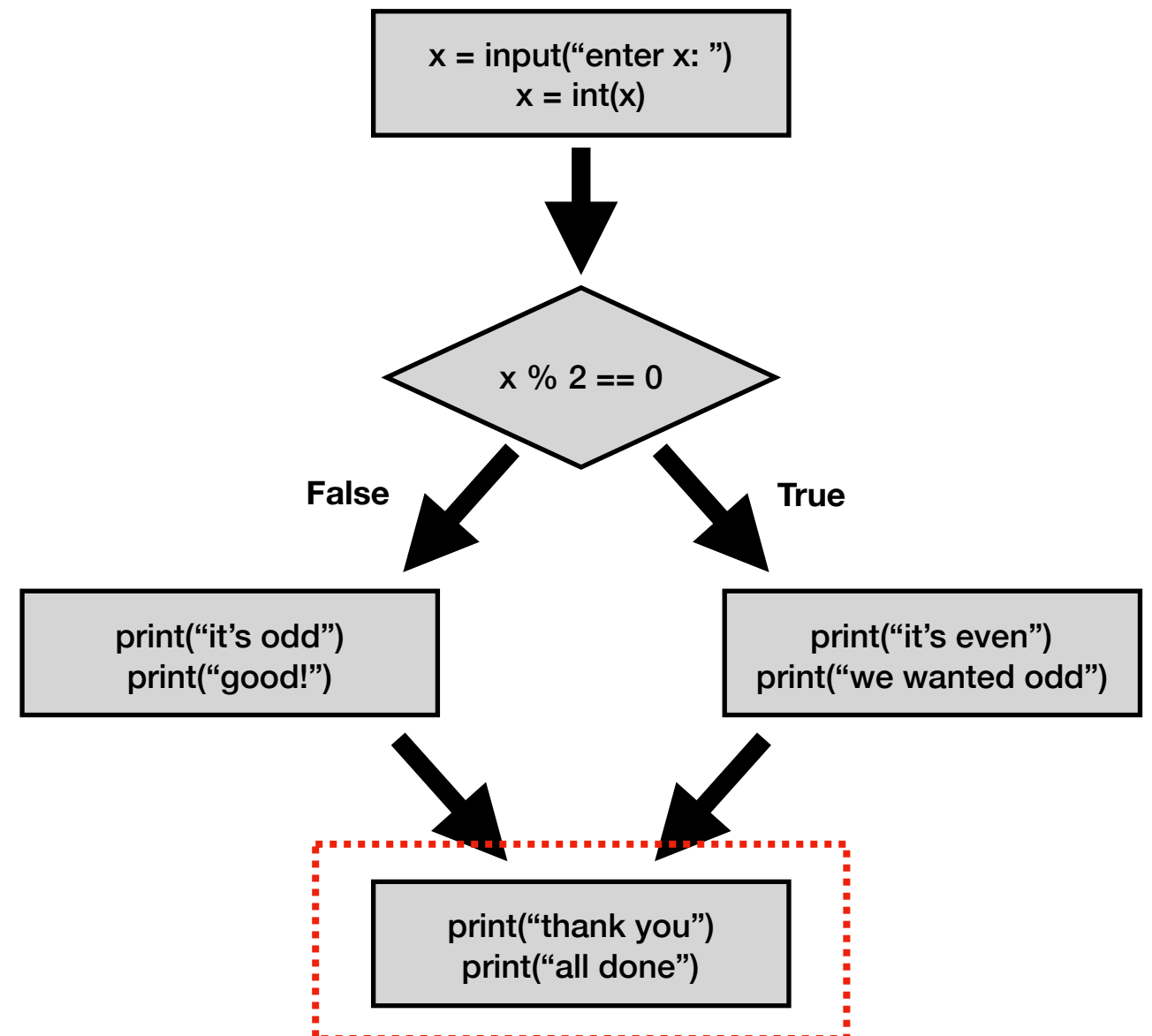
block of code inside "if"

block of code inside "else"

Code Blocks

Code:

```
x = input("enter x: ")
x = int(x)

if x % 2 == 0:
    print("it's even")
    print("we wanted odd")
else:
    print("it's odd")
    print("good!")

print("thank you")
print("all done")
```
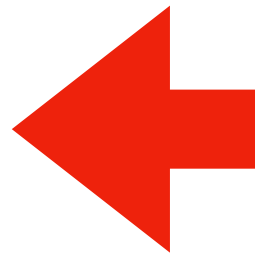
block of code inside "if"

block of code inside "else"

What if all this were inside a function?

Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

block of code inside "if"

block of code inside "else"

Code Blocks

Code:

```python
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

block of code inside "if"

block of code inside "else"

block of code in check_oddness

Code Blocks

You need to get good at "seeing" code blocks in Python code.

Code:

```python
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

block of code inside "if"

block of code inside "else"

block of code in check_oddness

Code Blocks

You need to get good at "seeing" code blocks in Python code.
Even blocks inside blocks inside blocks…

Code:

```python
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

block of code inside "if"

block of code inside "else"

block of code in check_oddness

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

Step 1: look for a colon at end of a line

Identifying Code Blocks

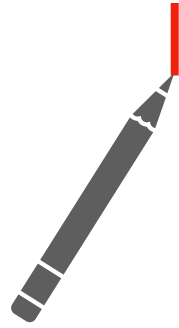**Code:**

```python
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

Step 2: start drawing a line on next code line, indented in

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")


    print("thank you")
    print("all done")

check_oddness()
```

Step 3: continue down until you hit code that is less indented

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

Step 4: box off the code

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")


    print("thank you")
    print("all done")

check_oddness()
```
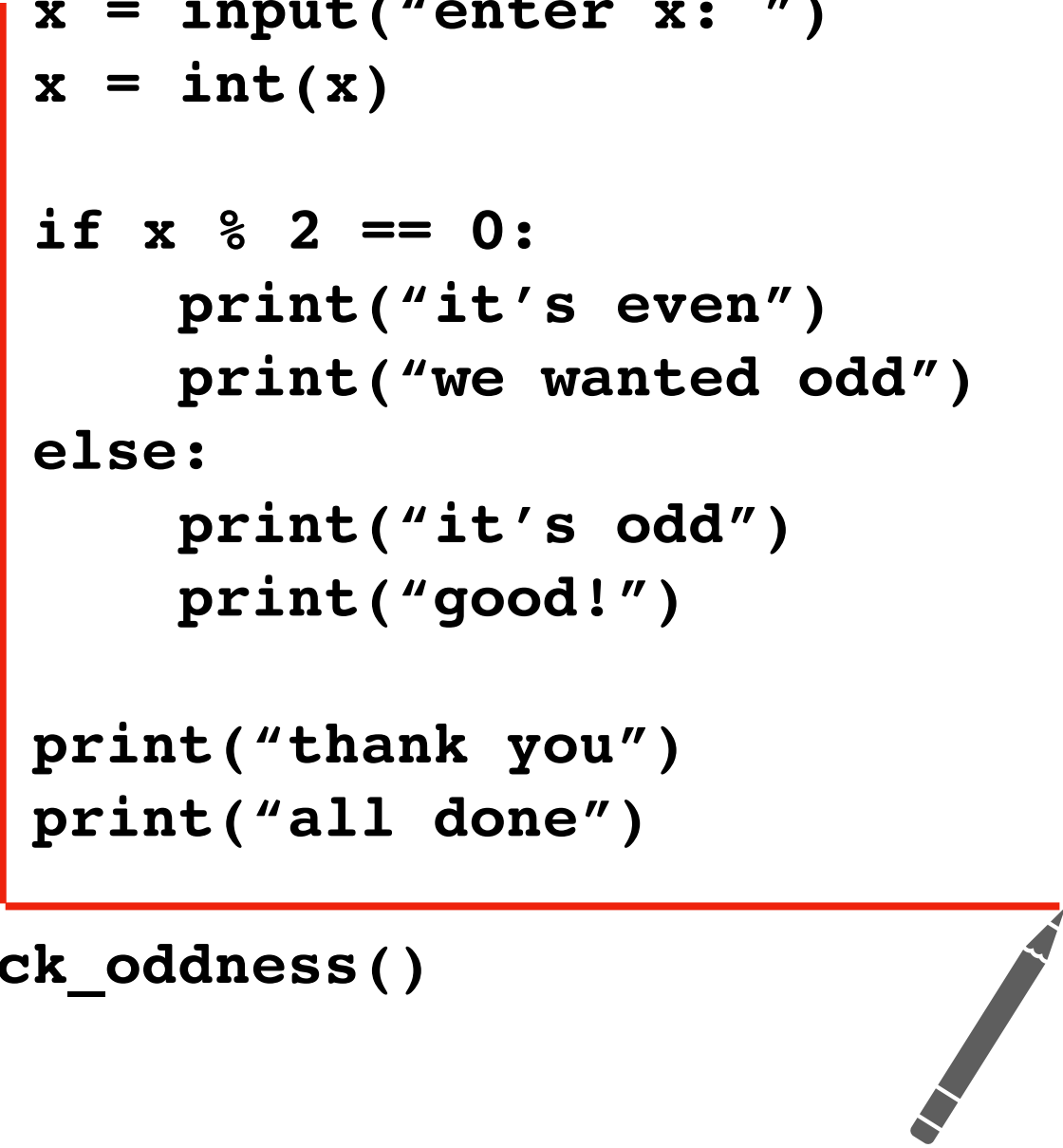
Step 4: box off the code

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

to find more boxes, look for the next colon and repeat

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")


    print("thank you")
    print("all done")

check_oddness()
```

to find more boxes, look for the next colon and repeat

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")


    print("thank you")
    print("all done")

check_oddness()
```

to find more boxes,
look for the next colon
and repeat

Identifying Code Blocks

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")

    print("thank you")
    print("all done")

check_oddness()
```

to find more boxes, look for the next colon and repeat

Identifying Code Blocks

Do practice problems on worksheet

Code:

```
def check_oddness():
    x = input("enter x: ")
    x = int(x)

    if x % 2 == 0:
        print("it's even")
        print("we wanted odd")
    else:
        print("it's odd")
        print("good!")


    print("thank you")
    print("all done")

check_oddness()
```

to find more boxes,
look for the next colon
and repeat

Today's Outline

Review

Control Flow Diagrams

Basic syntax for "if"

Identifying code blocks

Demos