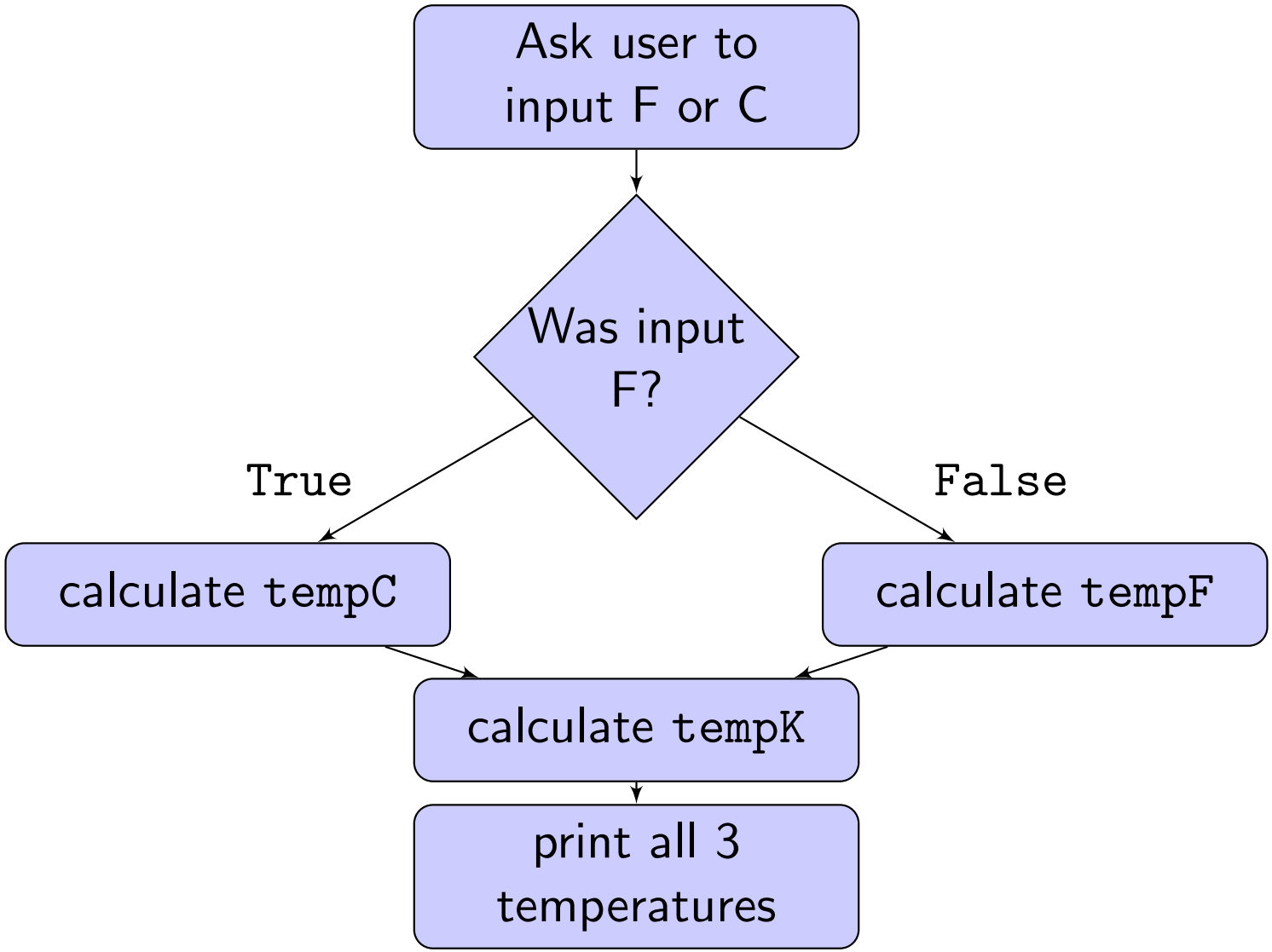
# Advanced Control Flow and Boolean Logic

# Basic Decision Control Flow



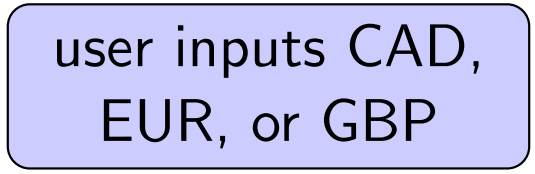
# Advanced Decision Control Flow

# Advanced Decision Control Flow

## Convert CAD, EUR, GBP to USD

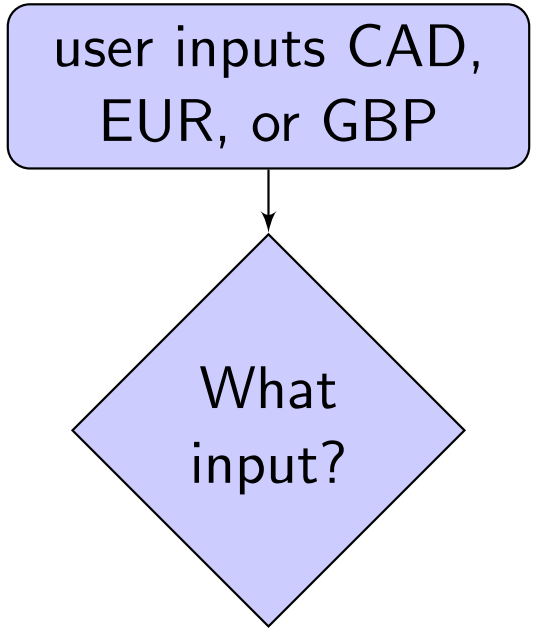
# Advanced Decision Control Flow

## Convert CAD, EUR, GBP to USD



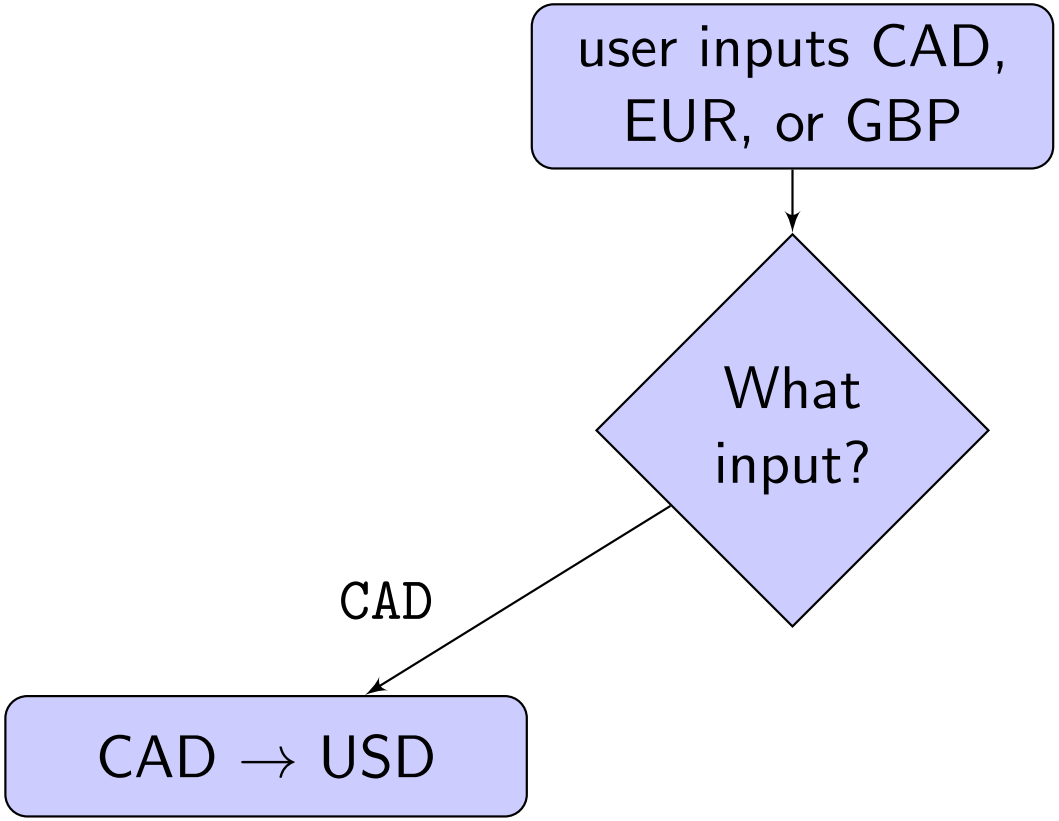
# Advanced Decision Control Flow

## Convert CAD, EUR, GBP to USD



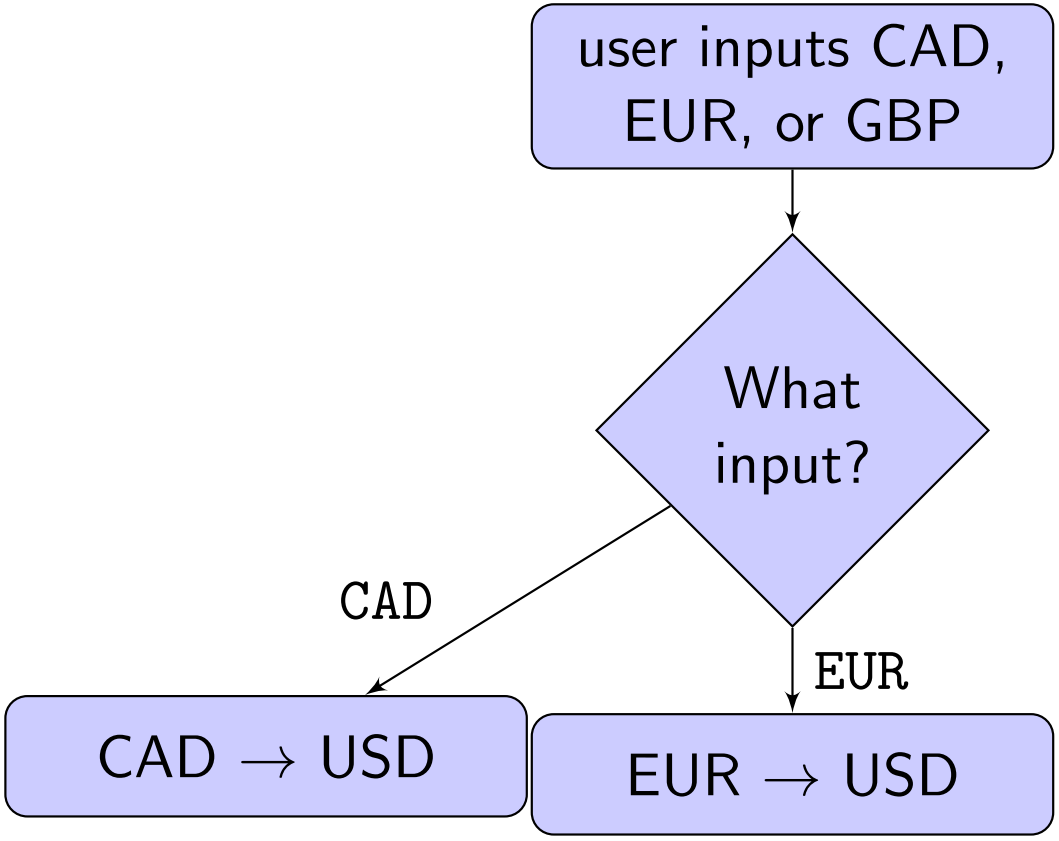
# Advanced Decision Control Flow

Convert CAD, EUR, GBP to USD



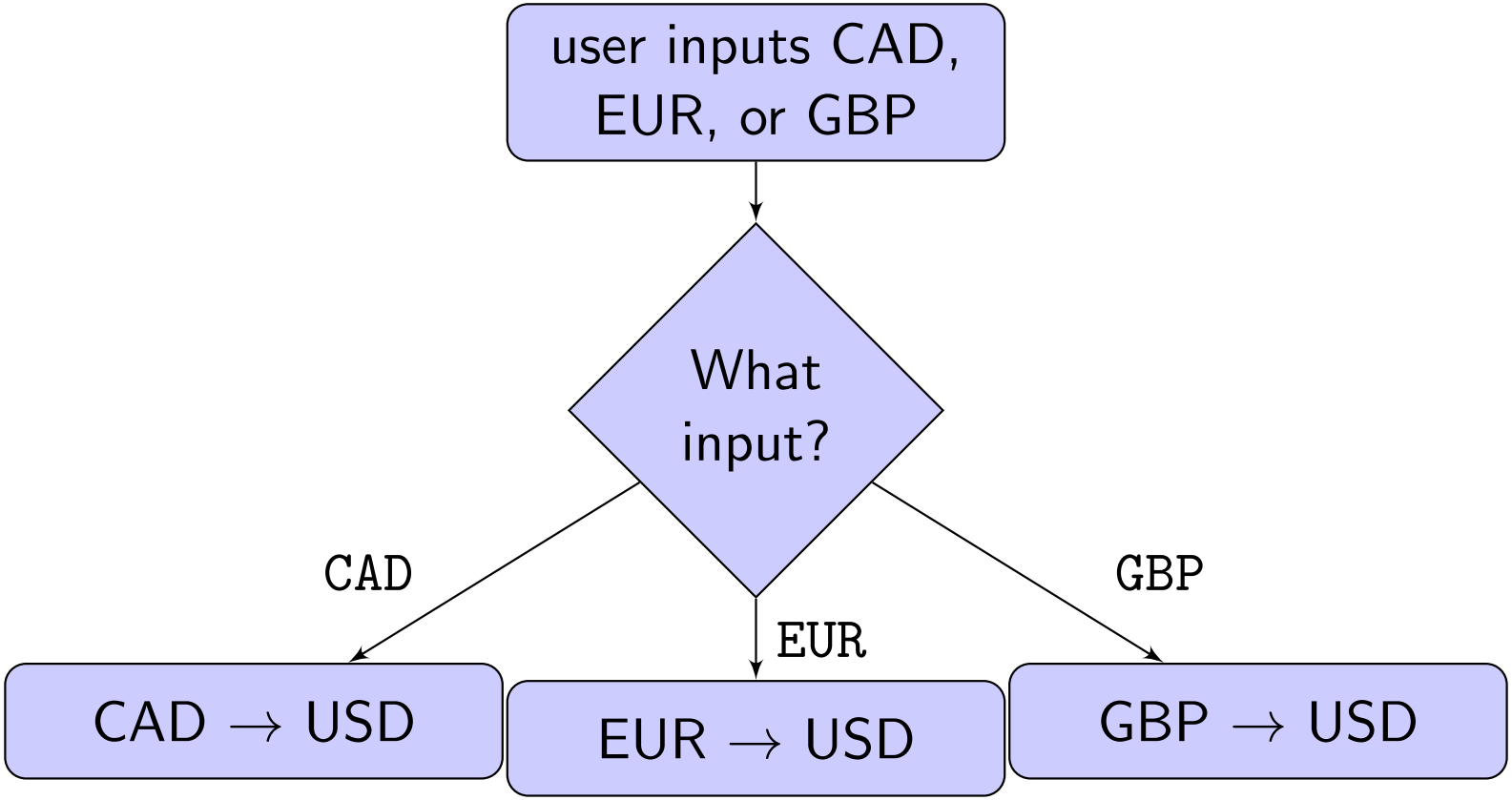
# Advanced Decision Control Flow

Convert CAD, EUR, GBP to USD



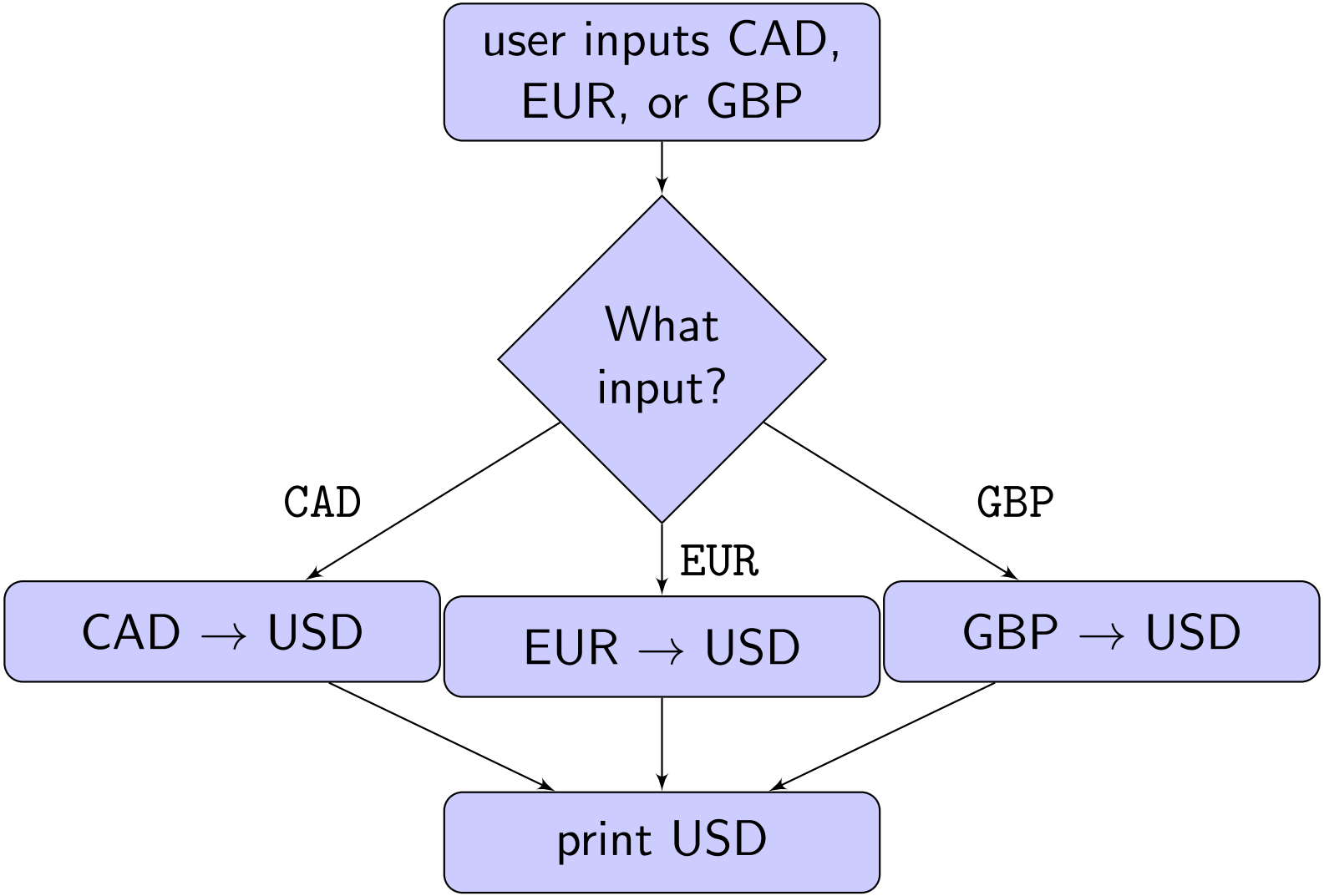
# Advanced Decision Control Flow

Convert CAD, EUR, GBP to USD



# Advanced Decision Control Flow

## Convert CAD, EUR, GBP to USD



# Nested if-else

# Nested if-else

`if` *condition1* :

# Nested if-else

`if` *condition1* :

    Python command 1
    Python command 2

# Nested if-else

```
if condition1 :
    Python command 1
    Python command 2
else:
```

# Nested if-else

```
if condition1 :
    Python command 1
    Python command 2
else:
    if condition2 :
```

# Nested if-else

```
if condition1 :
    Python command 1
    Python command 2
else:
    if condition2 :
        Python command 3
        Python command 4
```

# Nested if-else

```
if condition1 :
    Python command 1
    Python command 2
else:
    if condition2 :
        Python command 3
        Python command 4
    else:
        Python command 5
        Python command 6
```

# Nested if-else

```
if condition1 :
    Python command 1
    Python command 2
else:
    if condition2 :
        Python command 3
        Python command 4
    else:
        Python command 5
        Python command 6
    Python command 7 # executed by outer else block
```

# Nested if-else

```
if condition1 :
    Python command 1
    Python command 2
else:
    if condition2 :
        Python command 3
        Python command 4
    else:
        Python command 5
        Python command 6
    Python command 7 # executed by outer else block
Python command 8 # executed in all cases
```

# if-elif-else statement

# if-elif-else statement

`if` *condition1* :

# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
```

# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
elif condition2 :
```

# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
elif condition2 :
    Python command 3
    Python command 4
```

# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
elif condition2 :
    Python command 3
    Python command 4
else:
    Python command 5
    Python command 6
```

# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
elif condition2 :
    Python command 3
    Python command 4
else:
    Python command 5
    Python command 6
Python command 8 # executed in all cases
```

# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
elif condition2 :
    Python command 3
    Python command 4
else:
    Python command 5
    Python command 6
Python command 8 # executed in all cases
```

No place for command 7?

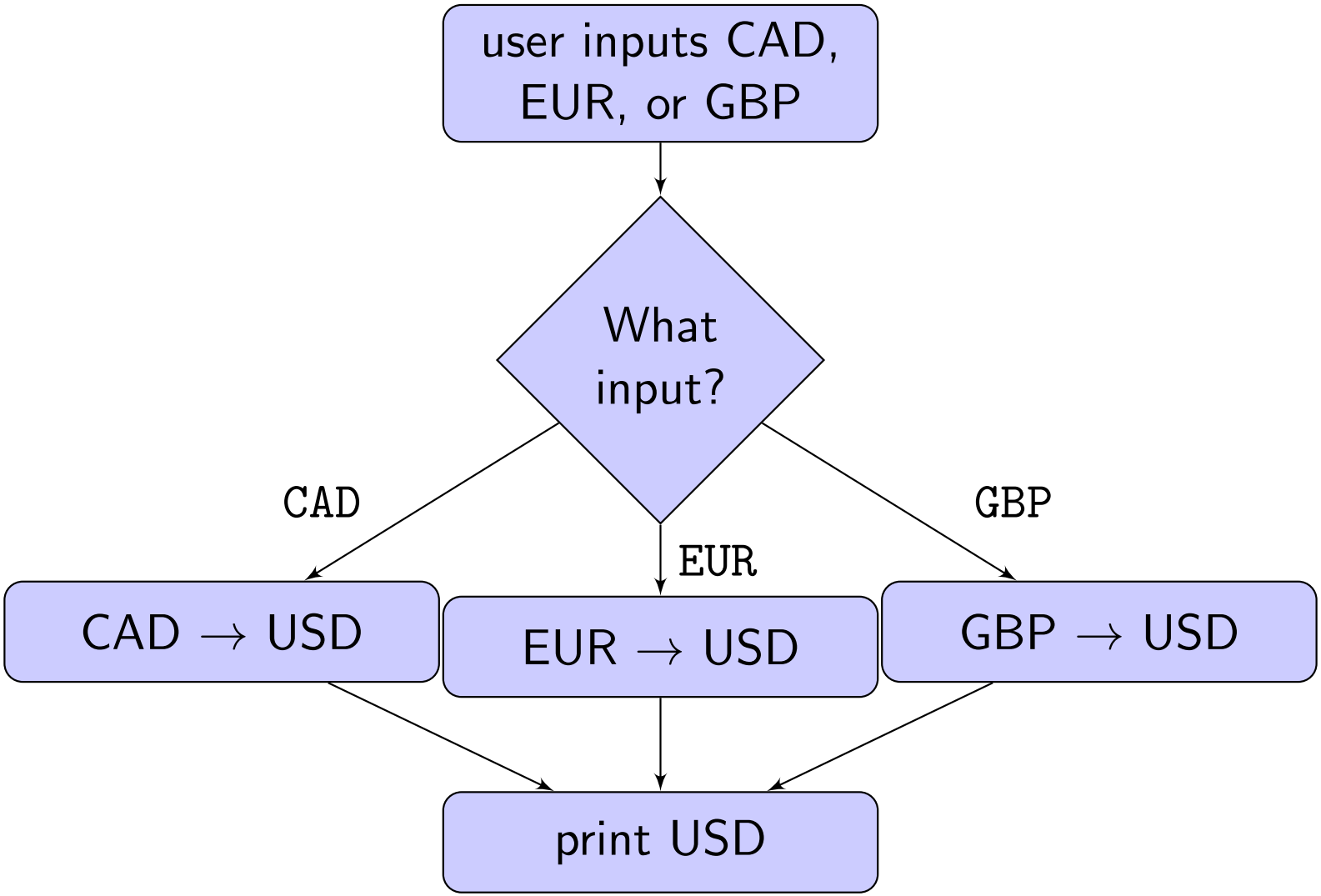
# if-elif-else statement

```
if condition1 :
    Python command 1
    Python command 2
elif condition2 :
    Python command 3
    Python command 4
else:
    Python command 5
    Python command 6
Python command 8 # executed in all cases
```

No place for command 7?

Notice that each block is mutually exclusive!

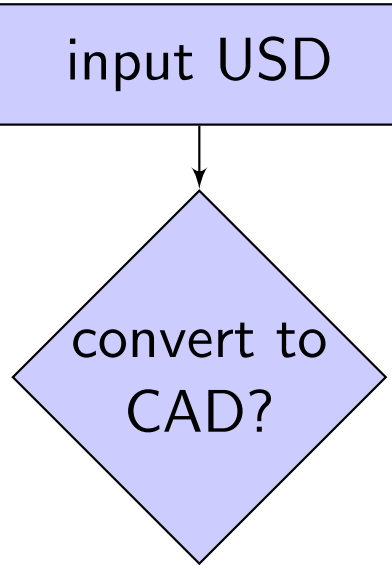
# Advanced Decision Control Flow



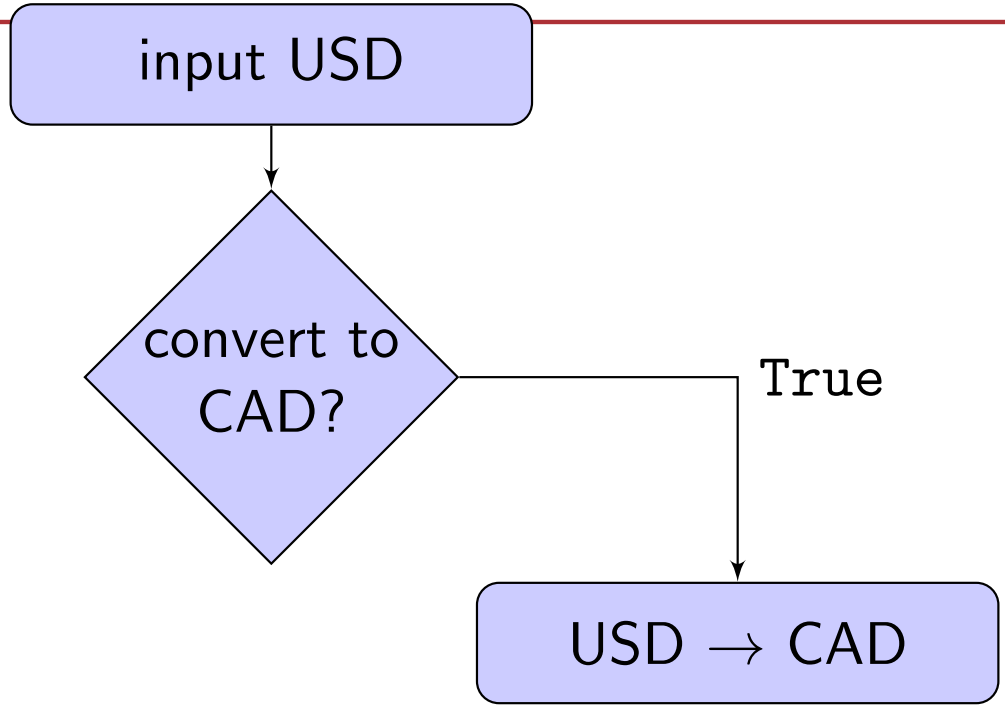
# if-sequence

# if-sequence

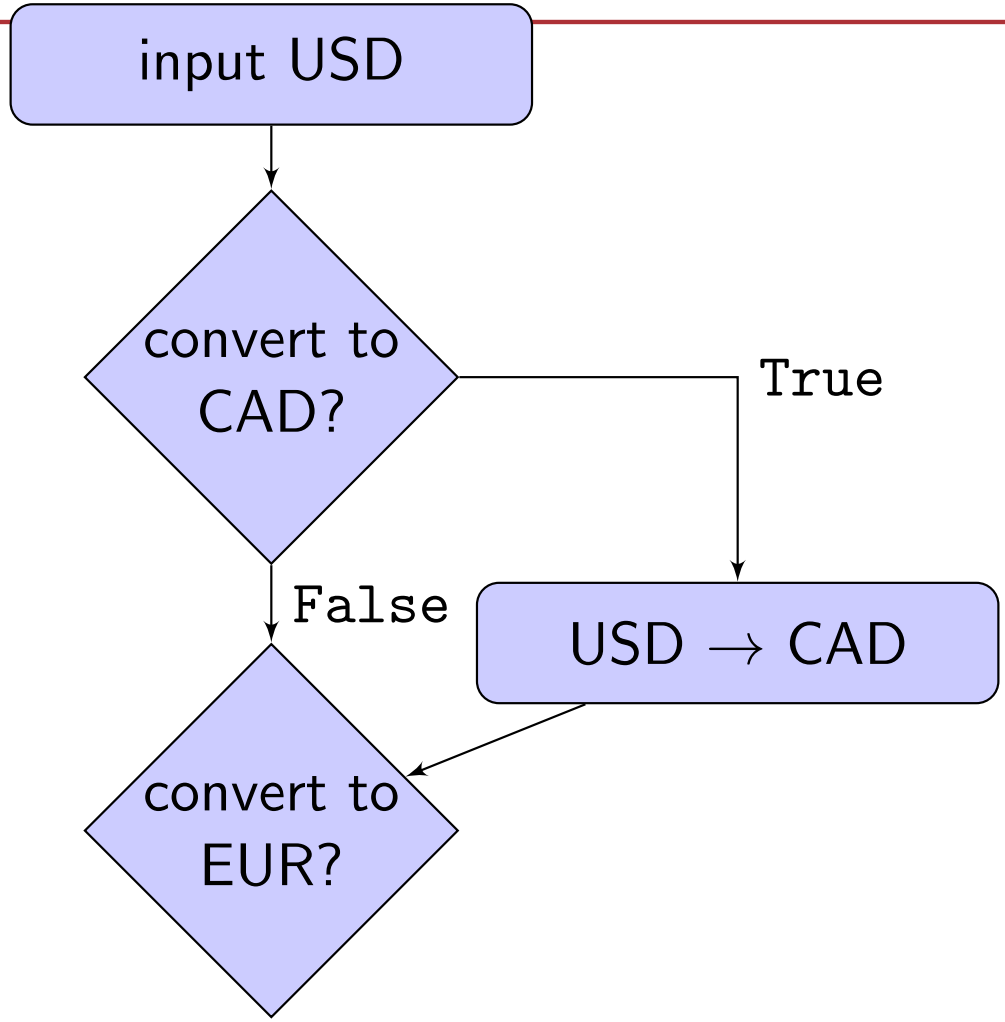
# if-sequence



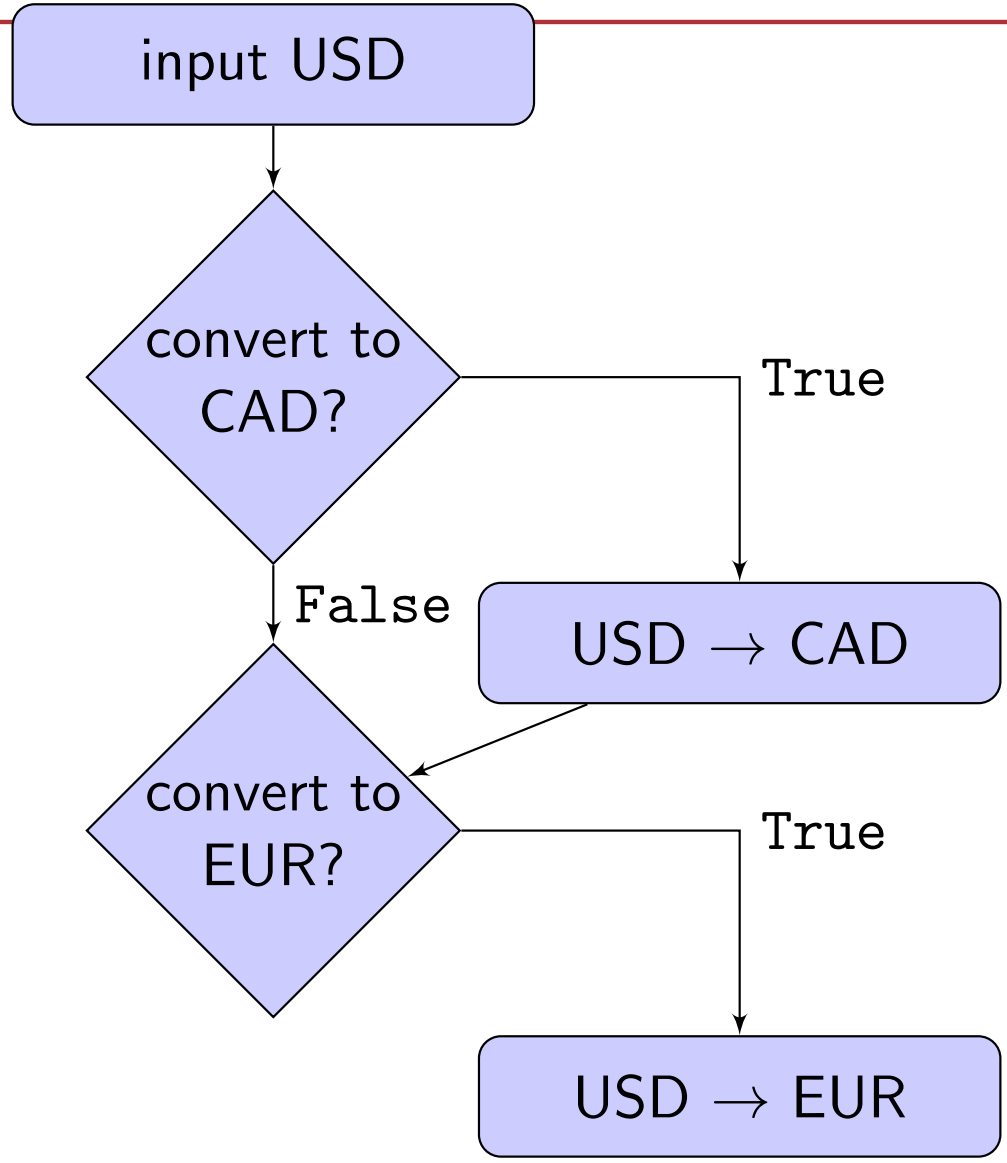
# if-sequence



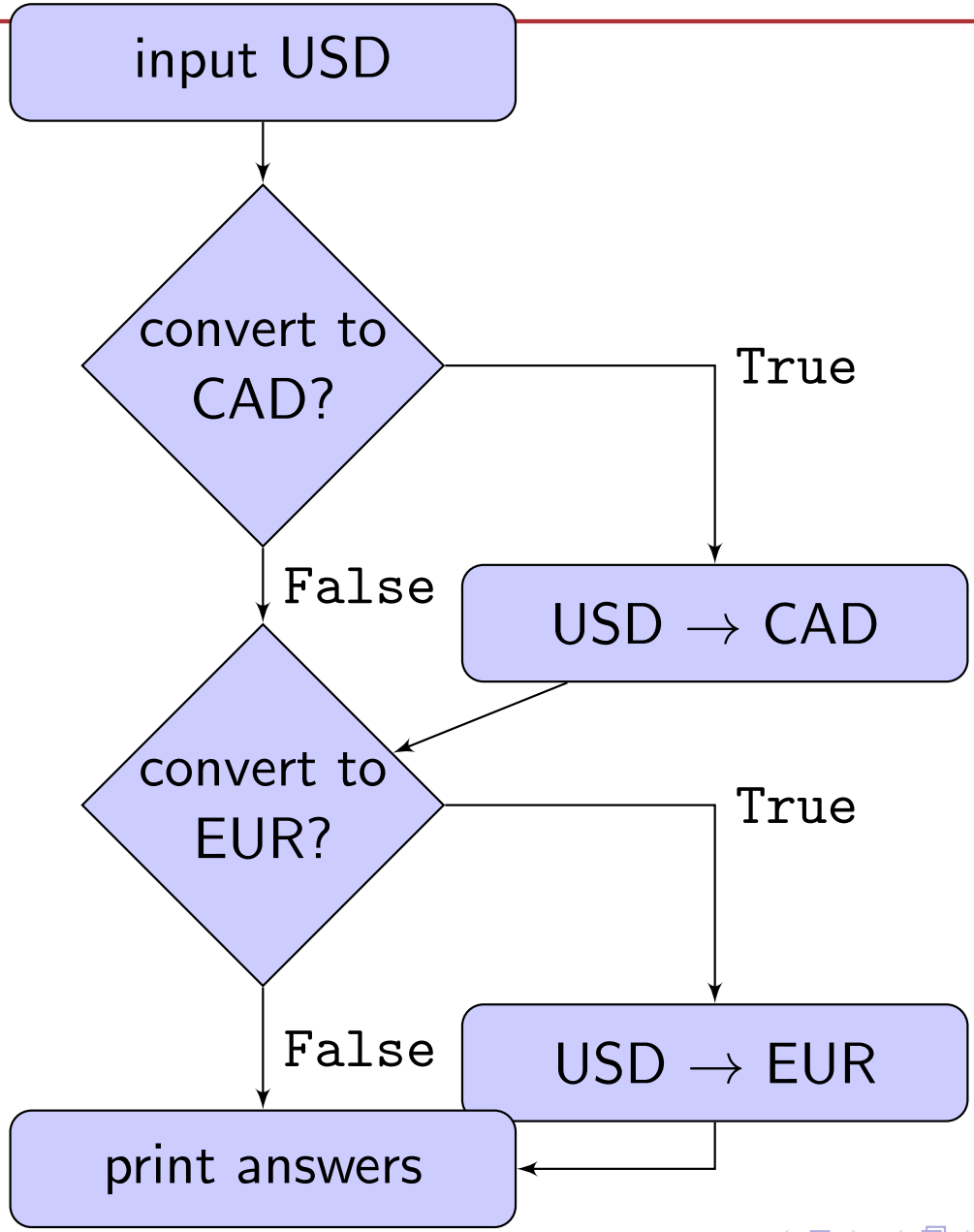
# if-sequence



# if-sequence


input USD
convert to CAD?
True
False
USD → CAD
convert to EUR?
True
USD → EUR

# if-sequence


input USD
convert to CAD?
True
False
USD → CAD
convert to EUR?
True
False
USD → EUR
print answers

# if-sequence

# if-sequence

`if` *condition1* :

# if-sequence

`if` *condition1* :

    Command 1

# if-sequence

```
if condition1 :
        Command 1
if condition2 :
```

# if-sequence

```
if condition1 :
    Command 1
if condition2 :
    Command 2
```

# if-sequence

```
if condition1 :
        Command 1
if condition2 :
        Command 2
Command 3
```

# Concept Check!

What do each of the following code blocks print?

1.
```
userInput = 8
if userInput < 4:
    print("Small number")
elif userInput < 10:
    print("Medium number")
elif userInput < 100:
    print("Big number")
```

2.
```
userInput = 8
if userInput < 4:
    print("Small number")
if userInput < 10:
    print("Medium number")
if userInput < 100:
    print("Big number")
```

# Advanced Logic with Flag Variables

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"
```

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")
```

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")

loginSuccess = True
```

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")

loginSuccess = True # Flag variable
```

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")

loginSuccess = True # Flag variable

if userName != SECRET_USER_NAME:
    loginSuccess = False
```

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")

loginSuccess = True # Flag variable

if userName != SECRET_USER_NAME:
    loginSuccess = False

if password != SECRET_PASSWORD:
    loginSuccess = False
```

# Advanced Logic with Flag Variables

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")

loginSuccess = True # Flag variable

if userName != SECRET_USER_NAME:
    loginSuccess = False

if password != SECRET_PASSWORD:
    loginSuccess = False

# Only True if both userName and password correct
if loginSuccess:
    print("Welcome to BookFace")
```

# An easier way...

# An easier way...

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")
```

# An easier way...

```
SECRET_USER_NAME = "zarkerberg11"
SECRET_PASSWORD = "myspace99"

userName = input("Please enter your user name:  ")
password = input("Please enter your password:  ")

# Only True if both userName and password correct
if (userName == SECRET_USER_NAME) and (password
                == SECRET_PASSWORD):
    print("Welcome to BookFace")
```

# Boolean logic

# Boolean logic

- **and**: `A and B` if **both** A is True and B is True

# Boolean logic

- **and**: `A and B` if **both** A is True and B is True
- **or**: `A or B` if **either** A is True or B is True

# Boolean logic

- **and**: `A and B` if **both** A is True and B is True
- **or**: `A or B` if **either** A is True or B is True
- **not**: `not A` **opposite:** True if `A` is False

# Boolean logic

- **and**: `A and B` if **both** A is True and B is True
- **or**: `A or B` if **either** A is True or B is True
- **not**: `not A` **opposite:** True if `A` is False

Just like with numerical operators, use parentheses to create complex expressions:

# Boolean logic

- **and**: `A and B` if **both** A is True and B is True
- **or**: `A or B` if **either** A is True or B is True
- **not**: `not A` **opposite:** True if `A` is False

Just like with numerical operators, use parentheses to create complex expressions:

```
userInput = int(input("Please enter a number:  "))
if (((not (userInput < 7)) and (not (userInput > 8)))
            or userInput == 9):
    print("You must have entered:  8, 9, or 10")
```

# Concept Check!

Which of the following expressions evaluate True?

1. 
```
>>> userInput = 8
>>> (userInput > 0) and (userInput < 5)
```
2. 
```
>>> userInput = 8
>>> (userInput > 0) or (userInput < 5)
```
3. 
```
>>> userInput = 8
>>> not((userInput > 0) or (userInput < 5))
```

# Warning!
# Advanced Stuff Ahead

# Short-circuiting evaluations

# Short-circuiting evaluations

```
userInput = float(input("Enter a number:  "))
if 1 / userInput < 4:
    print("That number was > 1/4")
else:
    print("That number was <= 1/4")
```

# Short-circuiting evaluations

```
userInput = float(input("Enter a number:  "))
if 1 / userInput < 4:
    print("That number was > 1/4")
else:
    print("That number was <= 1/4")
```

Produces an error when user enters: 0 as input.

# Short-circuiting evaluations

If the result of a Boolean expression can be determined from the beginning, it stops early

# Short-circuiting evaluations

If the result of a Boolean expression can be determined from the beginning, it stops early

```
userInput = float(input("Enter a number:  "))
if userInput != 0 and 1 / userInput < 4:
    print("That number was > 1/4")
else:
    print("That number was <= 1/4")
```

# Short-circuiting evaluations

If the result of a Boolean expression can be determined from the beginning, it stops early

```
userInput = float(input("Enter a number:  "))
if userInput != 0 and 1 / userInput < 4:
    print("That number was > 1/4")
else:
    print("That number was <= 1/4")
```

No error when user enters: `0` as input.