Chapter 3 Selection

Which door is safe?

Selection in a program

- Ask a user to enter something and response with different actions
  - Account authentication
  - Control the movement of a player
- Do calculations and then make decisions
  - Compute and interpret BMI

A simulation of lottery system

- A player selects a number (0-99) and notifies the lottery system
- The lottery system generates a winning number (0-99).
- The lottery system determines whether the user wins according to the following rule:
  - If player's number matches the lottery in exact order, the award is $10,000.
  - If player's number matches the lottery, the award is $3,000.
  - If one digit in player's number matches a digit in the lottery, the award is $1,000.

```python
import random

# Generate a lottery
lottery = random.randint(0, 99)

# Prompt the user to enter a guess
guess = int(input("Enter your lottery pick (two digits): "))

# Get digits from lottery
lotteryDigit1 = lottery // 10
lotteryDigit2 = lottery % 10

# Get digits from guess
guessDigit1 = guess // 10
guessDigit2 = guess % 10

print("The lottery number is", lottery)

# Check the guess
if guess == lottery:
    print("Exact match: you win $10,000")
elif (guessDigit2 == lotteryDigit1 and \
  guessDigit1 == lotteryDigit2):
    print("Match all digits: you win $3,000")
elif (guessDigit1 == lotteryDigit1
        or guessDigit1 == lotteryDigit2
        or guessDigit2 == lotteryDigit1
        or guessDigit2 == lotteryDigit2):
    print("Match one digit: you win $1,000")
else:
    print("Sorry, no match")
```

What to be covered next

- Boolean Expressions
  - Boolean Type
  - Relational Operator
- Conditional Execution
  - One-way Decisions
  - Two-way Decisions
  - Multi-way
  - Nested decision
- Logic operators
- Short-circuit Evaluation of Logical Expressions

Boolean Expressions

- **Boolean expressions** ask a question and produce a Yes or No result which we use to control program flow

- Boolean expressions using **relational operators** evaluate to True / False or Yes / No

Relational Operator

- Boolean expressions using **relational operators** evaluate to True / False or Yes / No
- Relational operators look at variables but do not change the variables

Python	Meaning
<	Less than
<=	Less than or Equal to
==	Equal to
>=	Greater than or Equal to
>	Greater than
!=	Not equal

Remember: "=" is used for assignment.

Relational Operator

```
>>> 42 == 42
True
>>> 42 == 99
False
>>> 2 != 3
True
>>> 2 != 2
False
```

```
>>> 42 < 100
True
>>> 42 > 100
False
>>> 42 < 42
False
>>> eggCount = 42
>>> eggCount <= 42
True
>>> myAge = 29
>>> myAge >= 10
True
```

```
>>> 'hello' == 'hello'
True
>>> 'hello' == 'Hello'
False
>>> 'dog' != 'cat'
True
>>> True == True
True
>>> True != False
True
>>> 42 == 42.0
True
>>> 42 == '42'
False
```

Boolean Type

- ***Boolean*** data type has only two values: True and False
- No quotes around *True* and *False*
- Start with a capital **T** or **F**, with the rest of the word in lowercase

```
>>> spam = True
>>> spam
True
>>> true
Traceback (most recent call
last):
 File "<pyshell#2>", line 1,
in <module>
   true
NameError: name 'true' is not
defined
>>> True = 2 + 2
SyntaxError: can't assign to
keyword
```

Conditional Execution

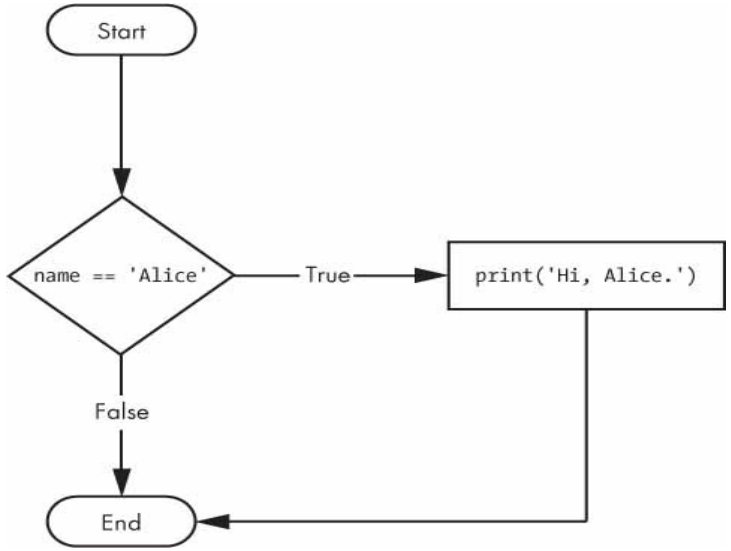
- One-Way Decisions
- Two-Way Decisions
- Multiple-Way Decisions

One-Way Decisions

- *if Statements*
- An **if statement's clause** will execute if the statement's condition is *True*.
- The clause is skipped if the condition is *False*.

```
if name == 'Alice':
    print('Hi, Alice.')
```

- The *if* keyword
- A condition
- A colon
- Starting on the next line, an indented block of code (if clause)

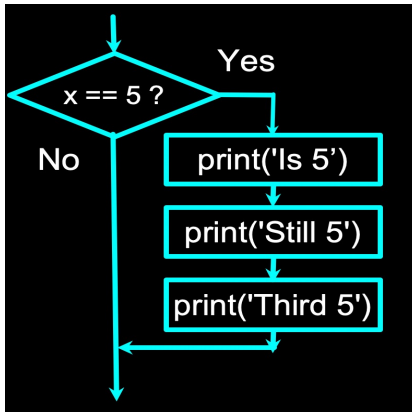


```
x = 5
print('Before 5')
if  x == 5 :
    print('Is 5')
    print('Is Still 5')
    print('Third 5')
print('Afterwards 5')
print('Before 6')
if x == 6 :
    print('Is 6')
    print('Is Still 6')
    print('Third 6')
print('Afterwards 6')
```

```
Before 5

Is 5
Is Still 5
Third 5
Afterwards 5
Before 6

Afterwards 6
```



Indentation

- **Increase indent** after an if statement or for statement (after :)
- **Maintain indent** to indicate the **scope** of the block (which lines are affected by the if/for)
- **Reduce indent** back to the level of the if statement or for statement to indicate the end of the block
- Blank lines are ignored - they do not affect **indentation**
- Comments on a line by themselves are ignored with regard to **indentation**

increase / maintain after *if* or *for*

decrease to indicate end of block

```
x = 5
if x > 2 :
    print('Bigger than 2')
    print('Still bigger')
print('Done with 2')

for i in range(5) :
    print(i)
    if i > 2 :
        print('Bigger than 2')
    print('Done with i', i)
print('All Done')
```

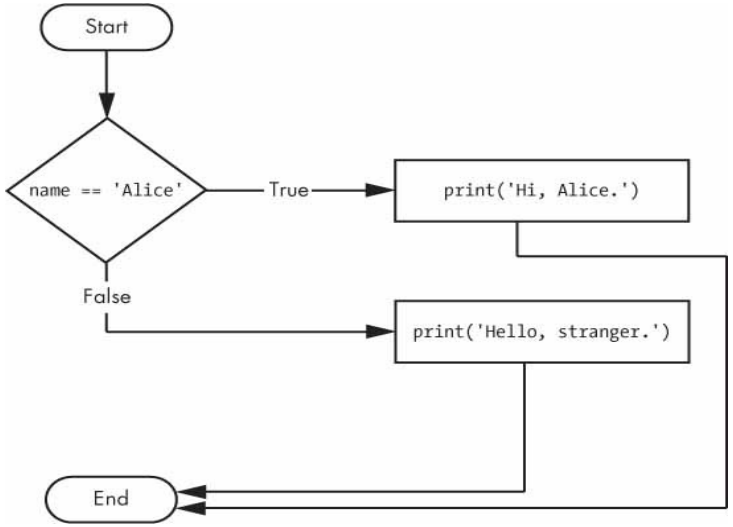
Two-way decisions

- Sometimes we want to do one thing if a logical expression is true and something else if the expression is false
- It is like a fork in the road - we must choose one or the other path but not both
- An **if clause** can be followed by an **else statement**. The else clause is executed only when the if statement's condition is *False*.

- The *if* keyword
- A condition
- A colon
- Starting on the next line, an indented block of code (if clause)

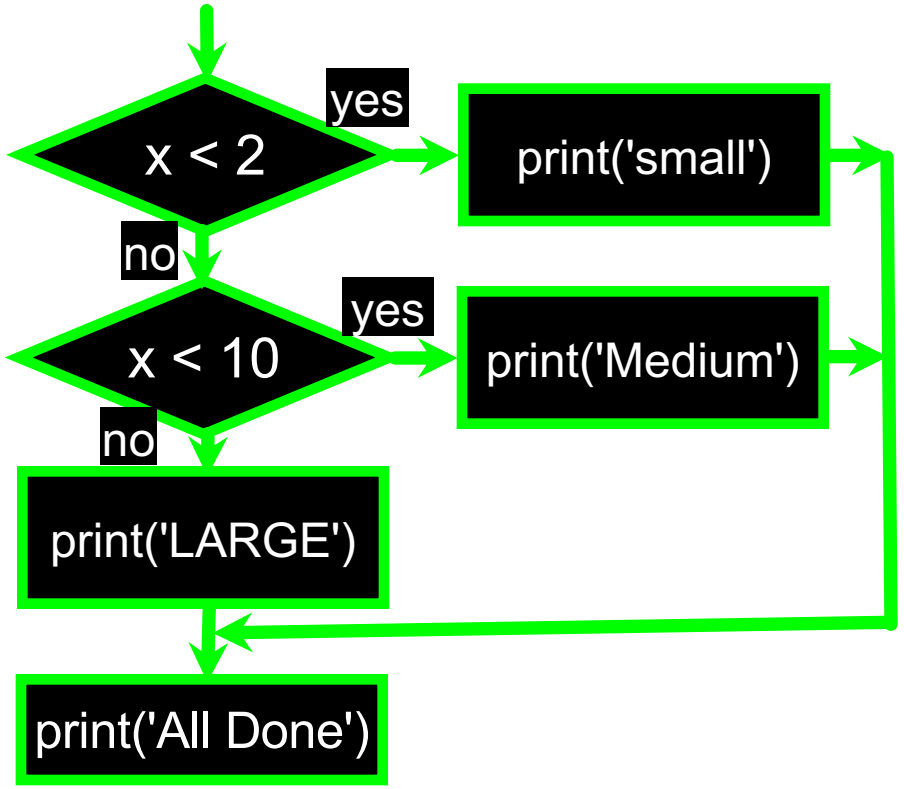
- The *else* keyword
- A colon
- Starting on the next line, an indented block of code (called the else clause)

```
if name == 'Alice':
    print('Hi, Alice.')
else:
    print('Hello, stranger.')
```



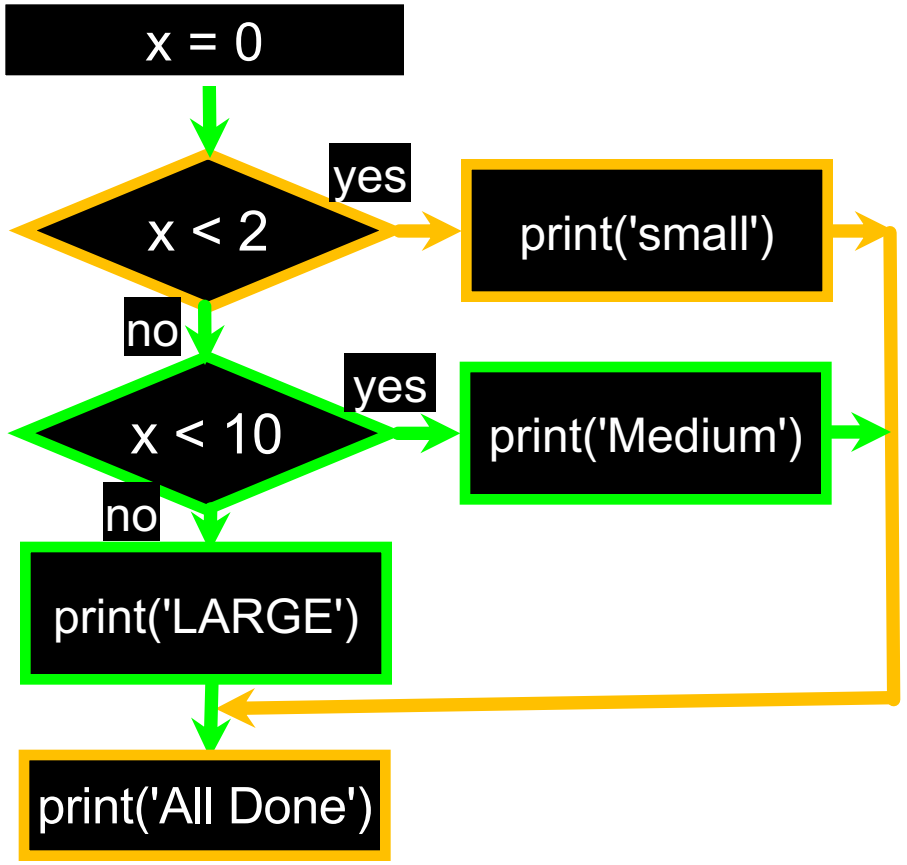
Multi-way

```
if x < 2 :
    print('small')
elif x < 10 :
    print('Medium')
else :
    print('LARGE')
print('All done')
```



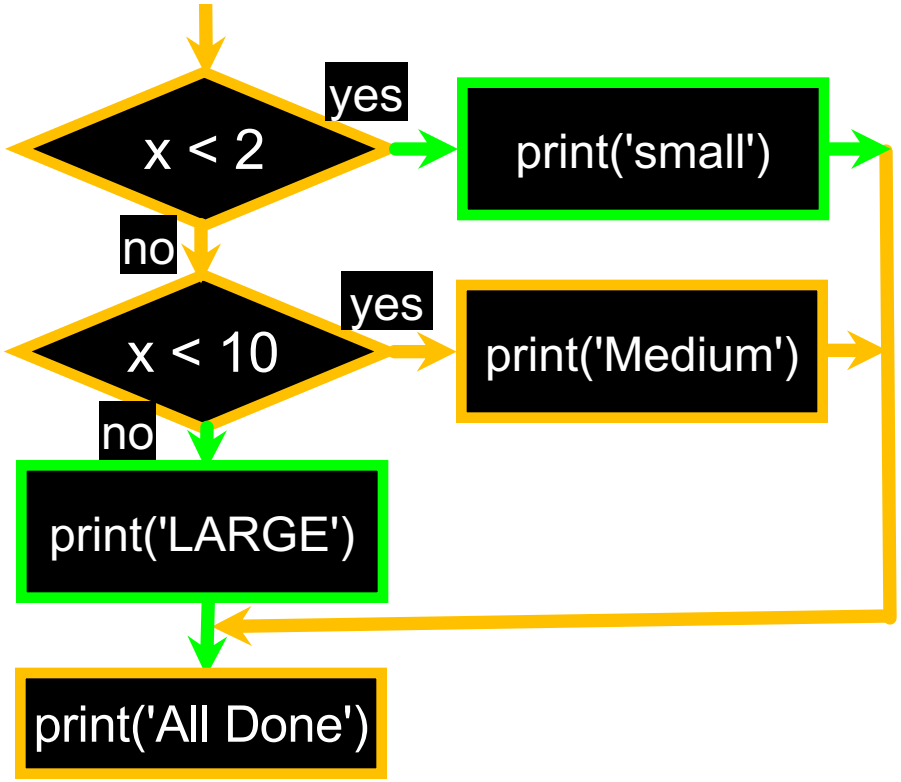
Multi-way

```
x = 0
if x < 2 :
    print('small')
elif x < 10 :
    print('Medium')
else :
    print('LARGE')
print('All done')
```



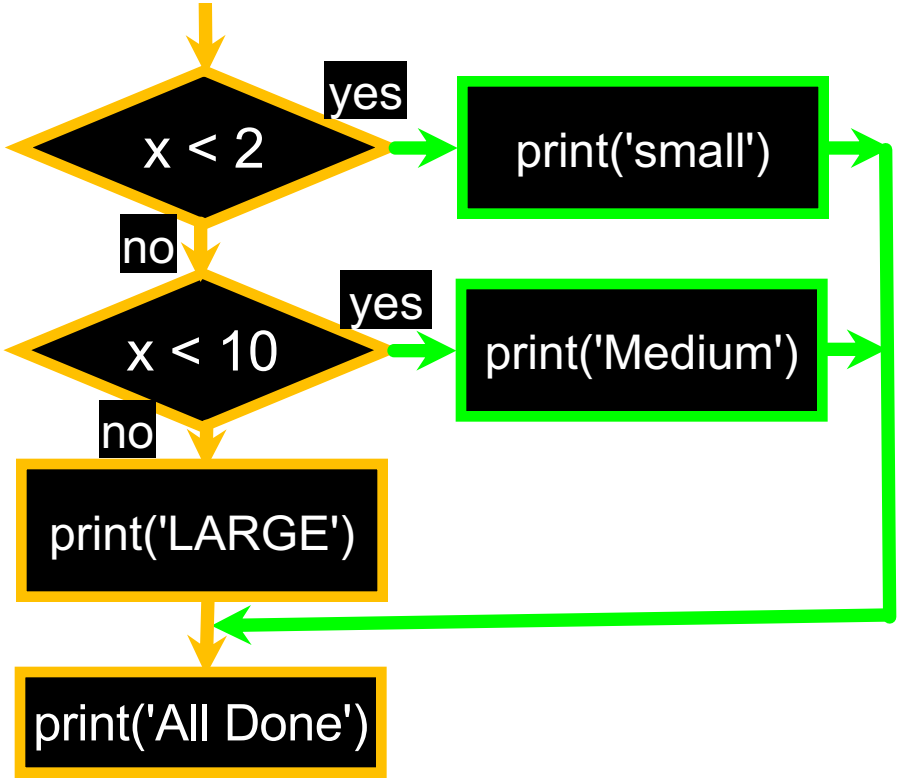
Multi-way

```
x = 5
if x < 2 :
    print('small')
elif x < 10 :
    print('Medium')
else :
    print('LARGE')
print('All done')
```



Multi-way

```
x = 20
if x < 2 :
    print('small')
elif x < 10 :
    print('Medium')
else :
    print('LARGE')
print('All done')
```



Multi-way Puzzles

Which will never print regardless of the value for x?

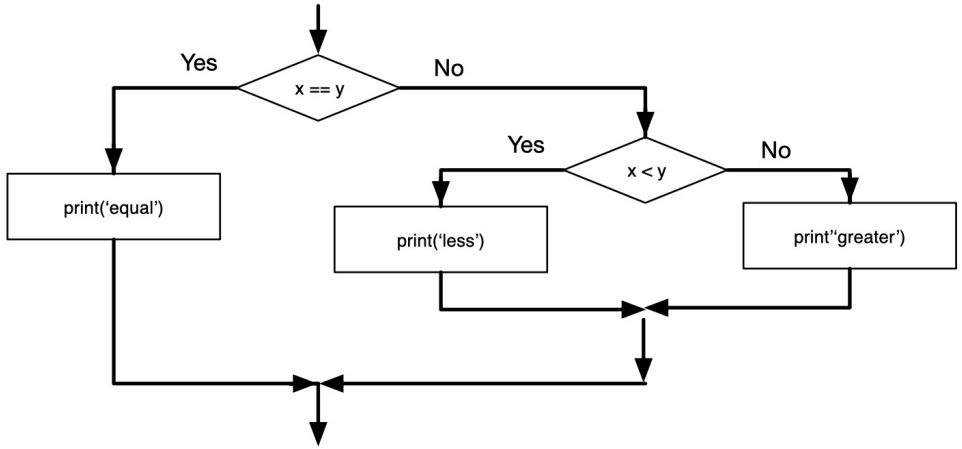
```
if x < 2 :
    print('Below 2')
elif x >= 2 :
    print('Two or more')
else :
    print('Something else')
```

```
if x < 2 :
    print('Below 2')
elif x < 20 :
    print('Below 20')
elif x < 10 :
    print('Below 10')
else :
    print('Something else')
```

Nested Decision

- One conditional can also be nested with another.

```
if x == y:
    print('x and y are equal’)
else:
    if x < y:
        print('x is less than y’)
    else:
        print('x is greater than y')
```



Nested Decision

- Nested conditionals become difficult to read very quickly.
- It is a good idea to avoid them when you can.

```
if  0 < x:
    if x < 10:
        print('x is a positive single-digit number.')
```

```
if 0 < x and x < 10:
    print('x is a positive single-digit number.')
```

Logic Operators

- Three logic operators
- Meaning of them like their meaning in English
  - x > 0 **and** x < 10 is true only if x is greater than 0 *and* less than 10.
  - n%2 == 0 **or** n%3 == 0 is true if *either* of the conditions is true, that is, if the number is divisible by 2 *or* 3.
  - **not** (x > y) is true if x > y is false; that is, if x is less than or equal to y.

Operator	Description
not	logical negation
and	logical conjunction
or	logical disjunction

Logic Operators

- The operand of the logical operators can also be any nonzero number.
  - Any nonzero number is interpreted as "true."

```
>>> 17 and True
True
```

Operator Precedence and Associativity

Operator Precedence Chart	PrecedenceOperator
↓	+, – (Unary plus and minus) ** (Exponentiation) not *, /, //, % (Multiplication, division, integer division, and remainder) +, – (Binary addition and subtraction) **<, <=, >, >= (Relational)** ==, != (Equality) **and** **or** =, +=, -=, *=, /=, //=, %= (Assignment operators)

If operators with **the same precedence are next to each other**, we evaluate them from left to right.

Problem: Determining Leap Year?

This program first prompts the user to enter a year as an int value and checks if it is a leap year.

A year is a leap year if it is divisible by 4 but not by 100, or it is divisible by 400.

(year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)

```
year = int(input("Enter a year: "))

# Check if the year is a leap year
isLeapYear = (year % 4 == 0 and year % 100 != 0) or \
   (year % 400 == 0)

# Display the result
print(year, "is a leap year?", isLeapYear)
```

A simulation of lottery system

- A player selects a number (0-99) and notifies the lottery system
- The lottery system generates a winning number (0-99).
- The lottery system determines whether the user wins according to the following rule:
  - If player's number matches the lottery in exact order, the award is $10,000.
  - If player's number matches the lottery, the award is $3,000.
  - If one digit in player's number matches a digit in the lottery, the award is $1,000.

```python
import random

# Generate a lottery
lottery = random.randint(0, 99)

# Prompt the user to enter a guess
guess = int(input("Enter your lottery pick (two digits): "))

# Get digits from lottery
lotteryDigit1 = lottery // 10
lotteryDigit2 = lottery % 10

# Get digits from guess
guessDigit1 = guess // 10
guessDigit2 = guess % 10

print("The lottery number is", lottery)

# Check the guess
if guess == lottery:
    print("Exact match: you win $10,000")
elif (guessDigit2 == lotteryDigit1 and \
  guessDigit1 == lotteryDigit2):
    print("Match all digits: you win $3,000")
elif (guessDigit1 == lotteryDigit1
        or guessDigit1 == lotteryDigit2
        or guessDigit2 == lotteryDigit1
        or guessDigit2 == lotteryDigit2):
    print("Match one digit: you win $1,000")
else:
    print("Sorry, no match")
```