Objectives

- Conditional statements

Computational Thinking

- Learning how to think
  - Learning how to learn
  - Learning how to solve problems
- Process
  - Practice!
    - Review slides and examples after class
      - Run them in Python visualizer
  - Finding answers
    - Previous labs, handouts, wiki, …
  - Asking questions
    - We talk you through our process

> Drill good practice in early on smaller problems so that you are well-poised to handle the big problems!

Parts of an Algorithm

- Input, Output
- Primitive operations
  - What data you have, what you can do to the data
- Naming
  - Identify things we're using
- Sequence of operations
- Conditionals
  - Handle special cases
- Repetition/Loops
- Subroutines
  - Call, reuse similar techniques

Making Decisions

- Sometimes, we do things only if some condition holds (i.e., "is true")
- Examples
  - If the PB is new (has a safety seal)
    - Then, I will take off the safety seal
  - If it is raining and it is cold
    - Then, I will wear a raincoat
  - If it is Saturday or it is Sunday
    - Then, I will wake up at 9 a.m.
    - Otherwise, I wake up at 7 a.m.
  - If the shirt is purple or the shirt is on sale and blue
    - Then, I will buy the shirt

Conditionals

- Sometimes, we only want to execute a statement in certain cases
  - Example: Finding the absolute value of a number
    - |4| = 4
    - |-10| = 10
  - To get the answer, we multiply the number by -1 *only if it's a negative number*
  - Code:

```
if x < 0 :
    abs = x*-1
```

Typical Execution

```
fahr = eval(input("…"))
```
↓
```
celsius = 5/9*(fahr-32)
```
↓
```
print("celsius=", celsius)
```

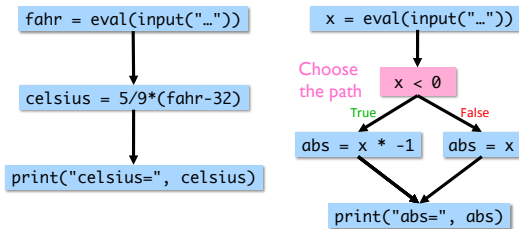So far, we've thought of programs as a *sequence* of statements.

Statements execute *in order*.

1

`if` Statements

- Change the ***control flow*** of the program

```
fahr = eval(input("…"))
```
↓
```
celsius = 5/9*(fahr-32)
```
↓
```
print("celsius=", celsius)
```

```
x = eval(input("…"))
```
↓
Choose the path
```
x < 0
```
True / False
```
abs = x * -1      abs = x
```
↓
```
print("abs=", abs)
```

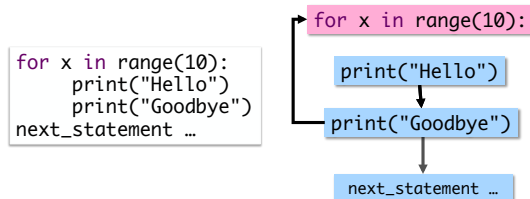
Other Constructs That Change Control Flow

- **`for`** loops
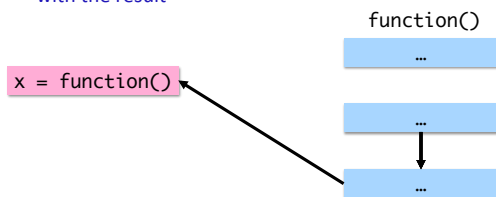 - Repeats a loop body a fixed number of times before going to the next statement after the **`for`** loop

```
for x in range(10):
    print("Hello")
    print("Goodbye")
next_statement …
```

```
for x in range(10):
```
```
print("Hello")
```
```
print("Goodbye")
```
```
next_statement …
```

Other Constructs That Change Control Flow

- Function calls
  - "Go execute some other code and then come back with the result"

```
function()
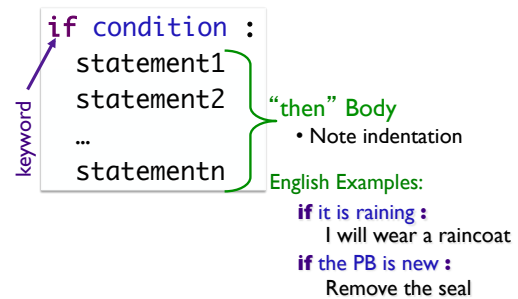```

```
x = function()
```
→
```
…
```
```
…
```
↓
```
…
```

---

## Syntax of `if` statement:
### Simple Decision

keyword →
```
if condition :
    statement1
    statement2
    …
    statementn
```
"then" Body
- Note indentation

English Examples:
**if** it is raining **:**
    I will wear a raincoat
**if** the PB is new **:**
    Remove the seal

---

## Conditions

- Syntax (typical, others later):
  - `<expr> <relational_operator> <expr>`
- Evaluates to either `True` or `False`
  - Boolean type

---

## Relational Operators

- Syntax:
  - `<expr> <relational_operator> <expr>`

*Low precedence*

| Relational Operator | Meaning |
|---|---|
| < | Less than? |
| <= | Less than or equal to? |
| > | Greater than? |
| >= | Greater than or equal to? |

Use Python interpreter

2

## Examples: Using Conditionals

- Determine if a number is even or odd

```python
x = eval(input("Enter a number: "))
remainder = x%2
if remainder == 0 :
    print(x, "is even")
if remainder == 1:
    print(x, "is odd")
```

Common Mistake:
Assignment Operator vs. Equality Operator

- Assignment operator: =
- Equality operator: ==

```python
x = eval(input("Enter a number: "))
remainder = x%2          Syntax error
if remainder = 0 :
    print(x, "is even. ")
```

Syntax of `if` statement:
Two-Way Decision

English Example:

if it is Saturday or Sunday :
  I wake up at 9 a.m.
else :
  I wake up at 7 a.m.

```
if condition :
    statement1
    statement2
    …
    statementn
else :
    statement1
    statement2
    …
    statementn
```

keywords

"then" Body

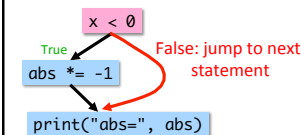"else" Body

---

If-Else statements (absolute values)

```python
abs = x
if x < 0 :
    abs *= -1
print("abs=", abs)
```
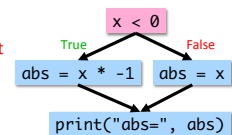
```python
if x < 0 :
    abs = x * -1
else :
    abs = x
print("abs=", abs)
```

If statement

x < 0
True
abs *= -1
False: jump to next statement
print("abs=", abs)

If-else statement

x < 0
True   False
abs = x * -1   abs = x
print("abs=", abs)

Examples: Using Conditionals

- Determine if a number is even or odd
- More efficient implementation
  - Don't need to check if remainder is 1 because if it's not 0, it *must* be 1

```python
x = eval(input("Enter a number: "))
remainder = x % 2
if remainder == 0:
    print(x, "is even")
else:
    print(x, "is odd")
```

Practice: Draw the Flow Chart

```python
print("This program determines your birth year")
print("given your age and current year")
print()
age = eval(input("Enter your age: "))

if age > 120:
    print("Don't be ridiculous, you can't be that old.")
else:
    currentYear = eval(input("Enter the current year: "))
    birthyear = currentYear - age
    print()
    print("You were either born in", birthyear, end='')
    print("or", birthyear-1)
print("Thank you.  Come again.")
```

What does this code do?

Practice: Speeding Ticket Fines

- Any speed clocked over the limit results in a fine of at least $50, plus $5 for each mph over the limit, plus a penalty of $200 for any speed over 90mph.

- Our program
  - Input: speed limit and the clocked speed
  - Output: either (a) that the clocked speed was under the limit or (b) the appropriate fine

`speedingticket.py`

Midterm Review

- Linux
- Very short answer, short answer, programs
  - No essay, no multiple choice
- Practice
  - Worksheets
  - Problems from class from scratch
  - Problems from lab from scratch
  - Problems from text book
- Review review questions :)

Nondeterministic Decisions

- Sometimes, we don't want to necessarily know that a specific decision is always made
- For example, games often use randomness to make decisions
  - Roll dice
  - Coin flips
  - Location and behavior of baddies

> How can we simulate coin flips?

Flipping Coins

- Simulate by randomly selecting between 0 (heads) and 1 (tails)
- Program: `coinFlip.py`

```python
from random import randint

HEADS=0
TAILS=1

# flip the coin
if randint(0,1) == HEADS:
    print("heads")
else:
    print("tails")
```

Syntax of **if** statement: Multi-Way Decision

```python
if condition :
    <then-body1>
elif condition :
    <then-body2>
elif condition :
    <then-body3>
    …
else :
    <default-body>
```

keywords

English Example:

if it is Saturday**:**
  I wake up at 10 a.m.
elif it is Sunday**:**
  I wake up at 9 a.m.
else :
  I wake up at 7 a.m.

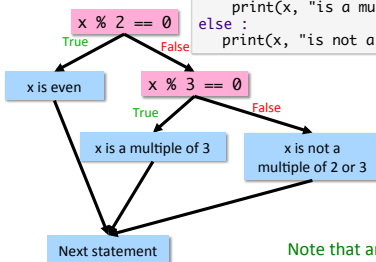
If-Else-If statements

```python
if x % 2 == 0 :
    print(x, "is a multiple of 2")
elif x % 3 == 0 :
    print(x, "is a multiple of 3")
else :
    print(x, "is not a multiple of 2 or 3")
```

x % 2 == 0 — True → x is even
x % 2 == 0 — False → x % 3 == 0 — True → x is a multiple of 3
x % 3 == 0 — False → x is not a multiple of 2 or 3

Next statement

Note that arithmetic is being performed in the conditional.
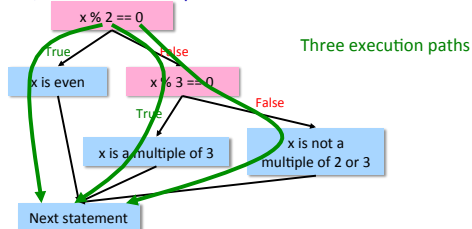
Testing with If Statements

- Make sure have test cases that execute each branch in control flow diagram
  - ➢ i.e., Each execution path is "covered"

Three execution paths

Using the building blocks: nesting if-else statements

```
if condition :
    if condition :
        statements
    else :
        statements
else:
    statements
```

if-else statement is **nested** inside the if

Using the building blocks: nesting if-else statements

```
if condition :
    statements
else:
    if condition :
        statements
    else :
        statements
```

if-else statement is **nested** inside the else

This structure can also be written as an **if-elif-else** statement

---

Practice: Numeric to Letter Grade

- Determine a given numeric grade's letter grade (A, B, C, D, or F)

Numeric Grade	Letter Grade
90 and above	A
80 to below 90	B
70 to below 80	C
60 to below 70	D

Enhanced Lottery Game

- Check if user's pick matches the number you generated

pick4winner.py

VA Lottery: Pick 4

- To play: you pick 4 numbers between 0 and 9
- To win: select the numbers that are selected by the magic ping-pong ball machine

- Your job: Simulate the magic ping-pong ball machines
  - ➢ Revision: display number as #-#-#-#

pick4.py

Discussion Questions

- What challenges did the Watson developers have to overcome?
  - How did they solve them?
  - Which were the hardest?
  - Were they what you expected?
- Any questions that you think would be particularly difficult for Watson?
- How can IBM/we/others use this new technology?
  - Anything you would not trust to Watson?
- What would you like to see added to Watson or how to improve it?

Questions

- Had anyone heard about this before reading the article?
- Any Jeopardy! fans?
- What are your thoughts/opinions of IBM before and after reading the article?
- Could the development of Watson go horribly, horribly wrong?

Changes to pick4.py

- Comments
  - Clarify what the program is doing
  - We wrote the program Wednesday
    - Already unclear on the details
- Constants
  - Give meaning to "magic numbers"
    - What were 0, 9, 3?

What Does This Program Do?

```python
import random

winningNum = ""

for x in xrange(3):
    numChosen = random.randint(0,9)
    winningNum += str(numChosen) +"-"

numChosen = random.randint(0,9)
winningNum += str(numChosen)

print "The number is", winningNum
```