Lecture 11: if / else

Adapted from slides by Marty Stepp and Stuart Reges
The `if` statement

*Executes a block of statements only if a test is true*

```python
if test:
    statement
...
statement
```

- Example:
  
  ```python
gpa = float(input("gpa? "))
if gpa >= 2.0:
    print("Application accepted.")
  ```
The if/else statement

Executes one block if a test is true, another if false

```python
if test:
    statement(s)
else:
    statement(s)
```

• Example:
  ```python
gpa = float(input("gpa? "))
if gpa >= 2.0:
    print("Welcome to Mars University!")
else:
    print("Application denied.")
```
Relational expressions

- if statements use logical tests.
  
  ```python
  if i <= 10: ...
  ```
  
- These are boolean expressions

- Tests use relational operators:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Example</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td>equals</td>
<td><code>1 + 1 == 2</code></td>
<td>True</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>does not equal</td>
<td><code>3.2 != 2.5</code></td>
<td>True</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>less than</td>
<td><code>10 &lt; 5</code></td>
<td>False</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>greater than</td>
<td><code>10 &gt; 5</code></td>
<td>True</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>less than or equal to</td>
<td><code>126 &lt;= 100</code></td>
<td>False</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>greater than or equal to</td>
<td><code>5.0 &gt;= 5.0</code></td>
<td>True</td>
</tr>
</tbody>
</table>
Misuse of if

• What's wrong with the following code?

```python
percent = float(input("What percentage did you earn? "))

if percent >= 90:
    print("You got an A!")

if percent >= 80:
    print("You got a B!")

if percent >= 70:
    print("You got a C!")

if percent >= 60:
    print("You got a D!")

if percent < 60:
    print("You got an F!")

...
Nested `if/else`  

*Chooses between outcomes using many tests*

```python
if test:
    statement(s)
elif test:
    statement(s)
else:
    statement(s)
```

- Example:
  ```python
  if x > 0:
    print("Positive")
  elif x < 0:
    print("Negative")
  else:
    print("Zero")
  ```
Nested `if/elif/elif`

- If it ends with `else`, exactly one path must be taken.
- If it ends with `if`, the code might not execute any path.

```python
if test:
    statement(s)
elif test:
    statement(s)
elif test:
    statement(s)
```

- Example:

```python
if place == 1:
    print("Gold medal!")
elif place == 2:
    print("Silver medal!")
elif place == 3:
    print("Bronze medal.")
```
Nested if structures

- exactly 1 path  (mutually exclusive)
  ```python
  if test:
      statement(s)
  elif test:
      statement(s)
  else:
      statement(s)
  ```

- 0 or 1 path  (mutually exclusive)
  ```python
  if test:
      statement(s)
  elif test:
      statement(s)
  elif test:
      statement(s)
  ```

- 0, 1, or many paths  (independent tests; not exclusive)
  ```python
  if test:
      statement(s)
  if test:
      statement(s)
  if test:
      statement(s)
  ```
Which nested \texttt{if/else}?

• (1) \texttt{if/if/if}  
  • (2) \texttt{nested if/else}  
  • (3) \texttt{nested if/elif/elif}

  • Whether a user is lower, middle, or upper-class based on income.
    • (2) \texttt{nested if / elif / else}

  • Whether you made the dean's list (GPA $\geq 3.8$) or honor roll (3.5-3.8).
    • (3) \texttt{nested if / elif}

  • Whether a number is divisible by 2, 3, and/or 5.
    • (1) \texttt{sequential if / if / if}

  • Computing a grade of A, B, C, D, or F based on a percentage.
    • (2) \texttt{nested if / elif / elif / elif / else}
Nested if/else question

Write a program that produces output like the following:

This program reads data for two people and computes their basal metabolic rate and burn rate.

Enter next person's information:
height (in inches)? 73.5
weight (in pounds)? 230
age (in years)? 35
gender (male or female)? male

Enter next person's information:
height (in inches)? 71
weight (in pounds)? 220.5
age (in years)? 20
gender (male or female)? female

Person #1 basal metabolic rate = 2042.3
high resting burn rate
Person #2 basal metabolic rate = 1868.4
moderate resting burn rate

• Basal Metabolic Rate Formula:

\[
\text{male BMR} = 4.54545 \times \text{(weight in lb)} + 15.875 \times \text{(height in inches)} - 5 \times \text{(age in years)} + 5
\]

\[
\text{female BMR} = 4.54545 \times \text{(weight in lb)} + 15.875 \times \text{(height in inches)} - 5 \times \text{(age in years)} - 161
\]

<table>
<thead>
<tr>
<th>BMR</th>
<th>Burn Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>below 12000</td>
<td>low</td>
</tr>
<tr>
<td>1200 to 2000</td>
<td>moderate</td>
</tr>
<tr>
<td>above 2000</td>
<td>high</td>
</tr>
</tbody>
</table>
# Nested if/else answer

# This program finds the basal metabolic rate (BMR) for two individuals. This variation includes several functions other than main.

# introduces the program to the user
def give_intro():
    print("This program reads data for two")
    print("people and computes their basal")
    print("metabolic rate and burn rate.")
    print()

# prompts for one person's statistics, returning the BMI
def get_bmr(person):
    print("Enter person", person, "information:")
    height = float(input("height (in inches)? "))
    weight = float(input("weight (in pounds)? "))
    age = float(input("age (in years)? "))
    gender = input("gender (male or female)? ")
    bmr = bmr_for(height, weight, age, gender)
    print()
    return bmr

...
# this function contains the basal metabolic rate formula for
# converting the given height (in inches), weight
# (in pounds), age (in years) and gender (male or female) into a BMR

def bmr_for(height, weight, age, gender):
    bmr = 4.54545 * weight + 15.875 * height - 5 * age
    if gender.lower() == "male":
        bmr += 5
    else:
        bmr -= 161
    return bmr

# reports the overall bmr values and status

def report_results(bmr1, bmr2):
    print("Person #1 basal metabolic rate =", round(bmr1, 1))
    report_status(bmr1)
    print("Person #2 basal metabolic rate =", round(bmr2, 1))
    report_status(bmr2)

# reports the burn rate for the given BMR value

def report_status(bmr):
    if bmr < 1200:
        print("low resting burn rate");
    elif bmr <= 2000:
        print("moderate resting burn rate")
    else:  # bmr1 > 2000
        print("high resting burn rate")

def main():
    give_intro()
    bmr1 = get_bmr(1)
    bmr2 = get_bmr(2)
    print(bmr1, bmr2)
    report_results(bmr1, bmr2)

main()