CS 160
CS Orientation

Data Types, Conditionals, and Repetition
Hierarchy of Operations

- Functions, i.e. `math.sqrt()`
- Power
- Mod
- Mult, Div
- Add, Sub
- Relational
- Logical
Python Examples

• $5 \times 2 + 3 - 10$
• $5 \times (2 + 3) - 10$
• $3 > 2 + 4$
• Etc.
Data Types/Converting

- string - “string of text”
- character – ‘a’
- integer – 79
- boolean – True
- float – 79.0
How about storing values?

- ch = ‘a’
- num = 79
- cont = True
- num = num + 1
- Can we mix types?
  - num = num + continue
  - num = num + ch
  - ch = ch + num
Python **Data Type** Demo...
Sequential Logic Structure

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Flowchart</th>
<th>Pseudocode</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \vdots ]</td>
<td>[ \vdots ]</td>
<td>[ \vdots ]</td>
</tr>
<tr>
<td>5. Instruction</td>
<td>Instruction</td>
<td>Instruction</td>
</tr>
<tr>
<td>6. Instruction</td>
<td>Instruction</td>
<td>Instruction</td>
</tr>
<tr>
<td>7. Instruction</td>
<td>Instruction</td>
<td>Instruction</td>
</tr>
<tr>
<td>8. [ \vdots ]</td>
<td>Instruction</td>
<td>Instruction</td>
</tr>
</tbody>
</table>

\[ \vdots \]
Python Sequential Logic

print("   *   ")
print("  ***  ")
print(" ***** ")
print(" *******")
print("*********")
Decision Logic Structure

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Flowchart</th>
<th>Pseudocode</th>
</tr>
</thead>
<tbody>
<tr>
<td>\vdots</td>
<td></td>
<td>\vdots</td>
</tr>
<tr>
<td>5. If &lt;decision&gt;</td>
<td></td>
<td>If &lt;decision&gt;</td>
</tr>
<tr>
<td>then</td>
<td></td>
<td>then</td>
</tr>
<tr>
<td>Instruction</td>
<td></td>
<td>Instruction</td>
</tr>
<tr>
<td>else</td>
<td></td>
<td>else</td>
</tr>
<tr>
<td>Instruction</td>
<td></td>
<td>Instruction</td>
</tr>
<tr>
<td>6. \vdots</td>
<td></td>
<td>\vdots</td>
</tr>
<tr>
<td>Instruction</td>
<td></td>
<td>Instruction</td>
</tr>
<tr>
<td>Decision Instruction</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>Instruction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>\vdots</td>
<td></td>
<td>\vdots</td>
</tr>
</tbody>
</table>
Relational Operators and Symbols

- >
- >=
- <
- <=
- ==
- !=
Logical Operators and Symbols

- not
- and
- or
Python Decision Logic:
Print 1, 3, 5, or 7 stars
Differences/Similarities in these?

```python
x=int(input("Print 1, 3, 5, 7 stars?"));

if(x==1):
    print("   *   ")
if(x==3):
    print("  ***  ")
if(x==5):
    print(" ***** ")
if(x==7):
    print("*******")
```

```python
if(x==1):
    print("   *   ")
elif(x==3):
    print("  ***  ")
elif(x==5):
    print(" ***** ")
eelif(x==7):
    print("*******")
```
Python Decision Logic:
Print 1, 3, 5, or 7 (for any other #) stars
Differences/Similarities in these?

```python
x = int(input("Print 1, 3, 5, 7 stars?"));

if(x==1):
    print("   *   ")
elif(x==3):
    print("  ***  ")
elif(x==5):
    print(" ***** ")
else:
    print("*******")
```

```python
x = int(input("Print 1, 3, 5, 7 stars?"));

if(x==1):
    print("   *   ")
elif(x==3):
    print("  ***  ")
elif(x==5):
    print(" ***** ")
else:
    print("*******")
```
Python Conditional Demo...
## Loop Logic Structure

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Flowchart</th>
<th>Pseudocode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image" alt="Flowchart Diagram" /></td>
<td><img src="image" alt="Pseudocode" /></td>
</tr>
<tr>
<td>5. Loop</td>
<td><img src="image" alt="Flowchart Diagram" /></td>
<td><img src="image" alt="Pseudocode" /></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Flowchart Diagram" /></td>
<td><img src="image" alt="Pseudocode" /></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="Flowchart Diagram" /></td>
<td><img src="image" alt="Pseudocode" /></td>
</tr>
<tr>
<td>6.</td>
<td><img src="image" alt="Flowchart Diagram" /></td>
<td><img src="image" alt="Pseudocode" /></td>
</tr>
</tbody>
</table>

- Loop
  - Instruction
  - Instruction
  - Instruction
  - Until <logical expression>

- Loop
  - Instruction
  - Instruction
  - Instruction
  - Until <logical expression>
Python Loop Logic

for x in range(7):
    print("*", end="")

OR

x=1
while(x<=7):
    print("*", end="")
    x+=1