Control Flow

Chris Piech and Mehran Sahami

CS106A, Stanford University
Housekeeping

• Class website: [http://cs106a.stanford.edu](http://cs106a.stanford.edu)
• Section sign-ups (sections start next week)
  – Sign-up at: [http://cs198.stanford.edu](http://cs198.stanford.edu) (will be on cs106a page)
  – Sign-ups start Thursday, September 17 at 5pm (PDT)
  – Sign-ups end Sunday, September 20 at 5pm (PDT)
  – Not first-come, first-served, but make sure to sign-up
• Application open for CS100A (link on CS106A website)
  – 1-unit supplementary section for stronger foundation
Install PyCharm

Please follow instructions *closely*. Post on Ed if you have problems.
• Reading: Should read the “Karel Reader” on class website
• Handout #3: “Honor Code”
• Handout #4: “Using Karel with PyCharm”
  – Tells you how to get started with writing Karel programs
• Handout #5: “Assignment 1”
  – Set of Karel programs for you to write
  – Due 1:00pm (PDT) on Friday, September 25th
• Only use features of Karel in the course reader
  – No other features of Python may be used in Karel programs!
Recall, Karel’s World

- Grid, where “corner” is intersection of each street/avenue
- Karel is currently on corner (1, 1)
- If Karel moved forward, Karel would be on corner (2, 1)
- Karel’s beeper bag can have 0, 1, or more (up to infinite) beepers
from karel.stanfordkarel import *

File: StepUpKarel.py

Karel program, where Karel picks up a beeper, jumps up on a step and drops the beeper off.

```python
def main():
    move()
    pick_beeper()
    move()
    turn_left()
    move()
    turn_right()
    move()
    put_beeper()
    move()

    # Karel turns to the right
```

```python
def turn_right():
    turn_left()
    turn_left()
    turn_left()
```

SOFTWARE ENGINEERING PRINCIPLE:
Aim to make programs readable by humans

Descriptive names (snake_case)
Today’s Goal

1. Code using loops and conditions
2. Trace programs that use loops and conditions
Today’s Route

You are here

Control Flow

The River of Control Flow

for Loops

while Loops

if

if/else

Piech and Sahami, CS106A, Stanford University
for loop

```python
for i in range(count):
    statements    # note indenting

def turn_right():
    for i in range(3):
        turn_left()  # note indenting
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```python
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```

First time through the loop.
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```python
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```python
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
```

Third time through the loop
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
def main():
    for i in range(4):
        put_beeper()
        move()
        turn_left()
Today’s Route

The River of Control Flow

- for Loops
- while Loops
- if
- if/else

Control Flow
def move_to_wall():
    while front_is_clear():
        move()  # note indenting
### Conditions Karel Can Check For

<table>
<thead>
<tr>
<th>Test</th>
<th>Opposite</th>
<th>What it checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>front_is_clear()</td>
<td>front_is_blocked()</td>
<td>Is there a wall in front of Karel?</td>
</tr>
<tr>
<td>left_is_clear()</td>
<td>left_is_blocked()</td>
<td>Is there a wall to Karel’s left?</td>
</tr>
<tr>
<td>right_is_clear()</td>
<td>right_is_blocked()</td>
<td>Is there a wall to Karel’s right?</td>
</tr>
<tr>
<td>beepers_present()</td>
<td>no_beepers_present()</td>
<td>Are there beepers on this corner?</td>
</tr>
<tr>
<td>beepers_in_bag()</td>
<td>no_beepers_in_bag()</td>
<td>Any there beepers in Karel’s bag?</td>
</tr>
<tr>
<td>facing_north()</td>
<td>not_facing_north()</td>
<td>Is Karel facing north?</td>
</tr>
<tr>
<td>facing_east()</td>
<td>not_facing_east()</td>
<td>Is Karel facing east?</td>
</tr>
<tr>
<td>facing_south()</td>
<td>not_facing_south()</td>
<td>Is Karel facing south?</td>
</tr>
<tr>
<td>facing_west()</td>
<td>not_facing_west()</td>
<td>Is Karel facing west?</td>
</tr>
</tbody>
</table>

This is in Chapter 10 of the Karel course reader
Task: Place Beeper Line

Before

After
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
def main():
    while front_is_clear():
        put_beeper()
        move()
    put_beeper()  # add final put_beeper

Not in **while** loop

Fixed!
def main():
    while front_is_clear():
        put_beeper()
        move()
    put_beeper()  # add final put_beeper
Fence Post Problem

Also sometimes called an “Off By One Error”
A program executes one line at a time.

The **while** loop checks its condition only at the start of the code block and before repeating.
Which Loop

Repeat Process

Know how many times (definite loop)

for Loop

Don’t know how many times (indefinite loop)

while Loop
Actual Bug from Marc II

1100
1525
1545

Started Cosine Tape (Sine
Started Multi Adder Test.
Relay #72 (moth) in re

First actual case of bug

1630 and agent started.
1700 closed down.
Today’s Route

The River of Control Flow

for Loops

while Loops

if

if/else

Control Flow
if condition:
    statements  # note indenting

def safe_pick_up():
    if beepers_present():
        pick_beeperer()  # note indenting
Today’s Route

The River of Control Flow

for Loops

while Loops

if

if/else

Control Flow
if condition:
    statements  # note indenting
else:
    statements  # note indenting

def invert_beepeers():
    if beepers_present():
        pick_beeperer()  # note indenting
    else:
        put_beeperer()  # note indenting
You just learned most of programming “control flow”
Today’s Goal

1. Code using loops and conditions
2. Trace programs that use loops and conditions
Putting it all together
SteepChaseKarel.py
Steeple Chase
Focus on One Steeple
Focus on One Steeple

turn_left()
Focus on One Steeple

\texttt{turn\_left()}

Piech and Sahami, CS106A, Stanford University
turn_left()
while right_is_blocked():
    move()
Focus on One Steeple

```python
import roboscript

# Initial position
position = (1, 1)

# Define the steeple
steeple = (2, 5)

# Function to check if the right is blocked
def right_is_blocked():
    return position[0] == steeple[0] and position[1] == steeple[1] - 1

# Main loop
while not right_is_blocked():
    position[1] += 1
    roboscript.move()

# Turn left
roboscript.turn_left()
```

Piech and Sahami, CS106A, Stanford University
Focus on One Steeple

```python
turn_left()
while right_is_blocked():
    move()
```
Focus on One Steeple

turn_left()
while right_is_blocked():
  move()
Focus on One Steeple

```
turn_left()
while right_is_blocked():
    move()
```
focus_on_one_steeple:

turn_left()
while right_is_blocked():
    move()
turn_left()
while right_is_blocked():
    move()
    turn_right()
Focus on One Steeple

```
turn_left()
while right_is_blocked():
    move()
    turn_right()
```
```
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
```
Focus on One Steeple

```python
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
```
turn_left()
while right_is_blocked():
    move()
    turn_right()
    move()
    turn_right()
```
turn_left()
while right_is_blocked():
    move()
    turn_right()
    move()
turn_right()
```
Focus on One Steeple

```python
turn_left()
while right_is_blocked():
    move()
turn_right()
movemove()
move()
turn_right()
movemes_to_wall()
```
turn_left()
while right_is_blocked():
    move()
    turn_right()
    move()
    turn_right()
move_to_wall()

def move_to_wall():
    while front_is_clear():
        move()
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()

def move_to_wall():
    while front_is_clear():
        move()
Focus on One Steeple

```python
def move_to_wall():
    while front_is_clear():
        move()
    turn_right()
    move()
    turn_right()
    move_to_wall()

turn_left()
while right_is_blocked():
    move()
    turn_right()
    move()
Focus on One Steeple

turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()

def move_to_wall():
    while front_is_clear():
        move()
def move_to_wall():
    while front_is_clear():
        move()
turn_left()
while right_is_blocked():
    move()
turn_right()
move()
turn_right()
move_to_wall()

def move_to_wall():
    while front_is_clear():
        move()
turn_left()
while right_is_blocked():
    move()
    turn_right()
move()
    turn_right()
    move_to_wall()
    turn_left()

def move_to_wall():
    while front_is_clear():
        move()
Focus on One Steeple

turn_left()
while right_is_blocked():
    move()
    turn_right()
move()
    turn_right()
move_to_wall()

def move_to_wall():
    while front_is_clear():
        move()
turn_left()
while right_is_blocked():
    move()
    turn_right()
move()
    turn_right()
move_to_wall()
    turn_left()

def move_to_wall():
    while front_is_clear():
        move()
Focus on One Steeple

```python
turn_left()
while right_is_blocked():
    move()
    turn_right()
move()

turn_right()
move_to_wall()
turn_left()

ascend_hurdle()

descend_hurdle()
```
Focus on One Steeple

turn_left()
while right_is_blocked():
    move()
turn_right()
move()

turn_right()
move_to_wall()
turn_left()

ascend_hurdle()

descend_hurdle()
def ascend_hurdle():
    turn_left()
    while right_is_blocked():
        move()
    turn_right()

ascend_hurdle()
move()
turn_right()
move_to_wall()
turn_left()

descend_hurdle()
def ascend_hurdle():
    turn_left()
    while right_is_blocked():
        move()
    turn_right()
ascend_hurdle()

def descend_hurdle():
    turn_right()
    move_to_wall()
    turn_left()
descend_hurdle()
Focus on One Steeple

def ascend_hurdle():
    turn_left()
    while right_is_blocked():
        move()
    turn_right()

def descend_hurdle():
    turn_right()
    move_to_wall()
    turn_left()

def jump_hurdle():
    ascend_hurdle()
    move()
    descend_hurdle()
A Whole Program:
SteepChaseKarel.py