CS 5 Today

indefinitely nested structure...

CS Midterm

Tuesday, Apr. 2
In-class, written
Page of notes is OK

• Recursion in Python
• Function composition
• Circuit design
• Hmm assembly code
• Loops in Python

See online practice...

Homework 8 Loops! due 1/1

homework 8 accommodations...

A week from Tuesday, 4/2 will be the CS 5 in-class midterm

Un-warnings:

five problems, written
worth 1 hw assignment
score worries? Extra extra-credit in hw9 and beyond

Suggestions:

go over in-class exercises and hwk problems
create a page of notes, 2-sided is OK
consider small variations of the problems – and how they would change the solutions...

that’s our approach...

Homework 9 preview

When Algorithms Discriminate...

The Mandelbrot Set

Lots of loops!

Pi from Pie

TTS Securities

(Extra)

Thinking in Loops...

PythonBat loop practice...

google for “PythonBat” then...

5 points required, up to 11 points available...
Thinking in *loops*

**for**

```python
for x in range(42):
    print(x)
```

**while**

```python
x = 1
while x < 42:
    print(x)
    x *= 2
```

What are the design differences between these two types of Python loops?

---

Loop design...

Is this a for or a while loop?

---

**definite iteration**

For a known list or # of iterations

**indefinite iteration**

For an unknown number of iterations
Homework 9 preview

When Algorithms Discriminate...

The Mandelbrot Set

Lots of loops!

Pi from Pie

TTS Securities

Loopy thinking

ASCII Art

CSS: Cascading Style Sheets

---

Pi from Pie?

Estimating π from pie?

What if we just throw darts at this picture?

Box (1,1)

(0,0)

(-1,-1)

Hints

How big is a side of the square? Its area?
How big is the radius of the circle? Its area?
How do these help?

Box

(1) Suppose you throw 100 darts at the square (all of them hit the square)
(2) Suppose 80 of the 100 hit inside the circle.
(3) How could you estimate π from these throws?

Name(s) ________________________

Estimating π from pie?

(Extra) CSS: Cascading Style Sheets

(Extra) ASCII Art

(Web extra) CSS: Cascading Style Sheets
Estimating $\pi$ from pie!

\[ \frac{\pi}{4} = \frac{\text{area}}{\text{area}} \]

$\pi \sim 4 \times \frac{\text{hits}}{\text{hits}}$

Loops: `for` or `while`?

\[
\begin{align*}
\text{pi}_\text{one} (e) &\quad e = \text{how close to } \pi \text{ we need to get} \\
\text{pi}_\text{two} (n) &\quad n = \text{number of darts to throw}
\end{align*}
\]

Which function will use which kind of loop?

Homework 9 preview

- #0: When Algorithms Discriminate...
  - #1 ~ lab: The Mandelbrot Set
  - #2: Lots of loops!
  - #3: Pi from Pie
  - #4: TTS Securities
  - (Extra): ASCII Art

Not just loops... Nested loops
Nested loops are familiar, too!

```python
for mn in range(60):
    for s in range(60):
        tick()
```

```python
for y in range(84):
    for m in range(12):
        for d in range(f(m,y)):
            for h in range(24):
                for mn in range(60):
                    for s in range(60):
                        tick()
```

Creating 2d structure ~ in ASCII

```
for row in range(3):
    for col in range(4):
        print("#")
```

Life clock

```
for mn in range(60):
    for s in range(60):
        tick()
```

Nested loops’ 2d structure

One hour ~ 3600 seconds

42nd minute’s 59th second

42

for mn in range(60): for s in range(60):
    tick()
Match! What code creates the fourth one? * and ** are extra!

for r in range(3):
    for c in range(6):
        if c > r:
            print('#', end='')
        else:
            print(' ', end='')
    print()

for r in range(3):
    for c in range(6):
        if c%2 == 1:
            print('#', end='')
        else:
            print(' ', end='')
    print()

for r in range(3):
    for c in range(6):
        if c%2 == r%2:
            print('#', end='')
        else:
            print(' ', end='')
    print()

Match! What code creates the fourth one?

What code creates the fourth one? * and ** are extra!

Name(s): ____________________ Try it!

for r in range[3):
    for c in range[6):
        if c > r:
            print('#', end='')
        else:
            print(' ', end='')
    print()

for r in range[3):
    for c in range[6):
        if c%2 == 1:
            print('#', end='')
        else:
            print(' ', end='')
    print()

for r in range[3):
    for c in range[6):
        if c%2 == r%2:
            print('#', end='')
        else:
            print(' ', end='')
    print()
from cs5png import *

def testImage():
    """ image demonstration """
    WD = 300
    HT = 200
    im = PNGImage( WD, HT )
    for row in range(HT):
        for col in range(WD):
            if col == row:
                im.plotPoint( col, row )
    im.saveFile()

Imagining Images

Complex #s!

i can't believe this!

$\sqrt{-1} = i$

$1j \times 1j = -1$

In[]: c = -2+1j

In[]: c**2

(3-4j)

In[]: (-2+1j)*(-2+1j)

Lab 9: the Mandelbrot Set

Consider an update rule for all complex numbers $c$

$z_0 = 0$

$z_{n+1} = z_n^2 + c$

$z = z**2 + c$ ; print(z)
Mandelbrot Definition

Consider an update rule for all complex numbers $c$

\[
\begin{align*}
  z_0 &= 0 \\
  z_{n+1} &= z_n^2 + c
\end{align*}
\]

Small values of $c$ keep the sequence near the origin, $0+0j$. Small values of $c$ keep the sequence near the origin, $0+0j$.

\[z = z^2 + c; \text{print}(z)\]

Lab 9: the Mandelbrot Set

Consider an update rule for all complex numbers $c$

\[
\begin{align*}
  z_0 &= 0 \\
  z_{n+1} &= z_n^2 + c
\end{align*}
\]

The shaded area are points that do not diverge for $z = z^2 + c$.
Higher-resolution M. Set

The black pixels are points that do not diverge for $z = z^2 + c$

Complex things always consisted of simple parts...

Before the M. Set, complex things were made of simple parts:

This was a "naturally occurring" object where zooming uncovers more detail, not less:

Atlas of the M. Set

Numbers in yellow indicate the number of dendrites or spiral arms found in each region, and in the corresponding Julia fractals for each region.
In the *Seahorse Valley*...
for $r$ in range(3):
    for $c$ in range(6):
        if $c > r$:
            print('#', end='')
        else:
            print(' ', end='')
    print()

for $r$ in range(3):
    for $c$ in range(6):
        if $c$%2 == 1:
            print('#', end='')
        else:
            print(' ', end='')
    print()

for $r$ in range(3):
    for $c$ in range(6):
        if $c$%2 == $r$%2:
            print('#', end='')
        else:
            print(' ', end='')
    print()