## As you arrive:

1. Start up your computer and plug it in
2. Check out today's project:

`Session11_WhileLoops`

*Plus in-class time working on these concepts AND practicing previous concepts, continued as homework.*

## Exam 1 preview

- Date and time of exam
- Exam location
- Format of exam (paper part + programming part)
- How to prepare for the exam

## Indefinite Loops

`while` statements

`break` statements

# Definite Loops (review)

- ***Definite*** loop:
  - Knows *before the loop starts to execute* the number of iterations of the loop body
  - Usually implemented by using a `for` statement
- Syntax: `for` *loop-variable* `in` *sequence*`:`
  *body*
- Examples: **m1_definiteLoops.py** in today's project
  - ***Counted*** loop: A loop where the sequence is a range expression
  - ***Loop through a sequence***:
    - ***Directly***
    - ***Using indices*** generated by a ***range*** statement

# Indefinite Loops

- Number of iterations is *not known* when loop starts
- Is typically a conditional loop
  - Keeps iterating as long as a certain condition *remains* `True`
  - The conditions are *Boolean expressions*
- Typically implemented using a `while` statement

```
sum = 0
for k in range(10):
    sum = sum + (k ** 3)
```

***Definite*** loop

```
sum = 0
k = 0
while k < 10:
    sum = sum + (k ** 3)
    k = k + 1
```

***Indefinite*** loop that computes the same sum as the definite loop

# While Loop

- A *pre-test loop*
  - Condition is tested at the top of the loop
- Example use of ***while*** loops

  Nadia deposits $100 in a savings account each month. Each month the account earns 0.25% interest on the previous balance. How many months will it take her to accumulate $10,000?
- Open **m2_moneyDeposit.py** in today's project.

# Infinite loops on purpose

- Simple while loop that runs forever.
- One answer:

```
while True:
    pass
```

# Break statement

- Useful if you want to break out of a loop in the *middle* of the loop, like this pattern:

```
while True:
    # Do some processing
    if processingSaysToStop:
        break
    # Do some more processing
```

# Break statement – Useful if you want to break out of a loop in the *middle* of the loop

```
def breakOutOfMiddleOfLoop():
    ''' Demonstrates a reasonable use of a BREAK statement '''
    while True:
        number = int(input("Enter a number bigger than 10: "))
        if number > 10:
            break   # User entered valid input, great!

        print("You idiot!  Your number was",
              number,
              "which is NOT bigger than 10.")
        print("Try again!")

    print()
    print("OK, now that I have your number that is")
    print("bigger than 10, let's boogie!")
    print("The base 10 log of your number is",
          math.log10(number))
```

# Exercise: While Loops

- Open m3_guessMyNumber.py in today's project.
- Follow the instructions there and demo your program to your instructor or an assistant when you finish.
- Commit your work

# Exam 1 information

- **Monday, January 10, 7 p.m. to 9 p.m.**
  - **Olin 267 (Fisher) and Olin 269 (Mutchler)**
- **Format: 2 hours.**
  - Paper part. Resources:
    - Zelle book, zellegraphics handout, create handout
    - 1 double-sided sheet of notes that you prepare
  - On-the-computer part. Resources:
    - Zelle book
    - Any written notes that you bring
    - Your computer and the files on it
    - Your own Subversion resources
    - **Any resources you can reach from the course web site by clicking only!**

# Possible topics for Exam 1

- Input/compute/output programs
  - Variables, assignment
  - Arithmetic and other expressions
  - input / print,  int / float
- Comments, testing
- Functions:
  - Calling
  - Defining
  - With parameters
  - Returning values
- Definite (for) loops:
  - Through a range
  - Through a sequence
  - Accumulating
    - Summing, Factorial
    - Counting
    - Appending to a sequence
- Operations on sequences
  - Lists, Strings, Tuples.  Indexing.
- Objects
  - Constructing
  - Using methods
  - Accessing instance variables
- Libraries, import
  - math, zellegraphics, time, create
- Decision structures
  - if … elif … else …
  - Relational and Boolean operators
- Files
  - open, read/write, close, parse input

# For your 1-page back-and-front sheet, be sure you have what you need on:

- Operators, including mod (`%`)
- `range` statements with 1, 2 or 3 arguments
- List operations, including `append`
- Writing and calling functions with parameters that return values
- Constructing objects and invoking methods on them
- Looping through lists, strings and tuples, with and without using indices. `for` loops and `while` loops
- Accumulator Loops: summing, counting, appending
- Other things from the list of topics that you might forget

# How to prepare for Exam 1

- Paper part:
  - Make a good 1-page, back and front, sheet with notes to remind you what you need.
  - Work some/all of the paper-part practice problems provided today (pages 9 through 17 – but there are quite a few problems that are NOT relevant, see your instructor for details)
- Computer part:
  - Work some of the computer-part practice problems provided today, perhaps 16, 17 (but not the 3rd bullet), 20, 22 and the House problem on page 25
  - As time permits, review your homework problems, making sure that you understand them
  - As time permits, review the slides (from the Schedule Page), making sure that you understand them