Outline Nov 14:

- Sorting algorithms
- Programming sorting algorithms
- Next time:
  - "is_sorted" function
  - Runtime of sorting algorithms

Notes

- Lab 8 due Saturday night (read BEFORE coming to lab!)
- Lab 9 due Monday after Thanksgiving
- Quiz 4 this Friday (let me know if you have conflicts)
Lab 5 extensions
Welcome to Tic-Tac-Toe!

Enter your choice like the numbers below:

|0|1|2|
|3|4|5|
|6|7|8|

AI chooses: 4

Enter 0-8 for your choice: 3

AI chooses: 2

Enter 0-8 for your choice: 6

AI chooses: 0

Enter 0-8 for your choice: 8

AI chooses: 1

AI wins!
Mega-tic  (Mirabai)

How many rows/columns do you want?: 6

*** Get ready to play tic-tac-toe! ***

Please enter your choices like the numbers below:
[0|1|2|3|4|5|
|6|7|8|9|10|11|
|12|13|14|15|16|17|
|18|19|20|21|22|23|
|24|25|26|27|28|29|
|30|31|32|33|34|35|]

AI moves:

Choose an empty spot (input 0-15): 15

|X|O| |O|
|---|---|---|
| | | | |
| | | | |

AI moves:

Choose an empty spot (input 0-15): 12

|X|O| |O|
|---|---|---|
| |X|X|O|
| | | | |

AI moves:

Choose an empty spot (input 0-15): 8

|X|O|O|O|
|---|---|---|
|X|X|O|
|X| |O|

AI moves:

Choose an empty spot (input 0-15): 4

|X|O|O|O|
|---|---|---|
|X|X|X|O|
|X|O|O| |
|X| |O|X|

You win!
Sorting
Sorting with cards

1) With a partner, set up a series of ~10 cards (out of order)

2) Try to come up with a sorting algorithm that only involves comparing and swapping elements

3) Check your algorithm with me or a ninja

4) Begin implementation in sorts.py

5) Here is our swap function from Week 6:

```python
def swap(i, j, lst):
    """This function swaps the ith and jth values of the lst."""
    temp = lst[i]
    lst[i] = lst[j]
    lst[j] = temp
```
Selection Sort idea

Swap(0, 5)
Swap(1, 4)
Swap(2, 4)
Swap(3, 3)
Swap(4, 4)

i = where min should go
m = where min is now

0 1 2 3 4 5
3 sorts for today

- **Selection Sort**: iteratively find the minimum element and place it in the correct position

- **Bubble Sort**: move through the list, swapping adjacent elements if they are out of place (repeat until sorted)

- **Insertion Sort (didn’t get to)**: for each element of the list, move it down until it is in the correct position
Types of sorting

- **Out-of-place**: leaves the original list alone and creates a new sorted list (returns new list)

- **In-place**: modifies (mutates) the original list via swaps so that it is now sorted

- **Pros of in-place sort**: no new data structure needed (saving space)

- **Cons of in-place sort**: original order destroyed (in some cases it might be important), can be more difficult to implement