CS1 Lecture 43 May 3, 2017

- HW 10 due Friday night, 11:59pm.
- HW 8 scores have been posted
- Final exam: Monday, May 8, 3:00-5:00pm, W10 PBB (this room)

Today

- HW 10 questions?
- More about the *limits of computation* mentioned throughout the semester, including
  - Turing Machines
  - start informal proof of the *halting problem*

Friday

Review for final exam

HW10 questions?

- *recommendation*: use Text widget rather than Label for displaying tweets.
  - in GUI initialization

    tweetText = tkinter.Text(width=…, height = …)

    tweetText.configure(state=tkinter.DISABLED)
  - in code to display tweet:

    tweetText.configure(state=tkinter.NORMAL)

    tweetText.delete(1.0, tkinter.END)

    tweetText.insert(…)

    tweetText.configure(state=tkinter.DISABLED)
- Few tweets have non-null 'coordinates' field, so many of your pins will likely be in the middle of the maps. That's okay.
  - *recommendation*: to get more tweets *with* non-null coordinates field, it seems to help to use small search radius – e.g. 2km rather than 5 or 10.

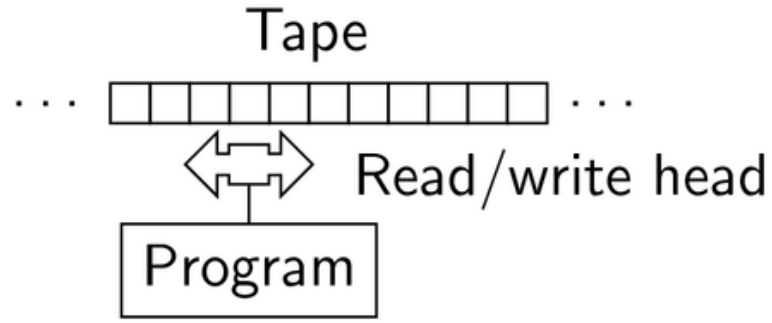
HW 10 Checklist

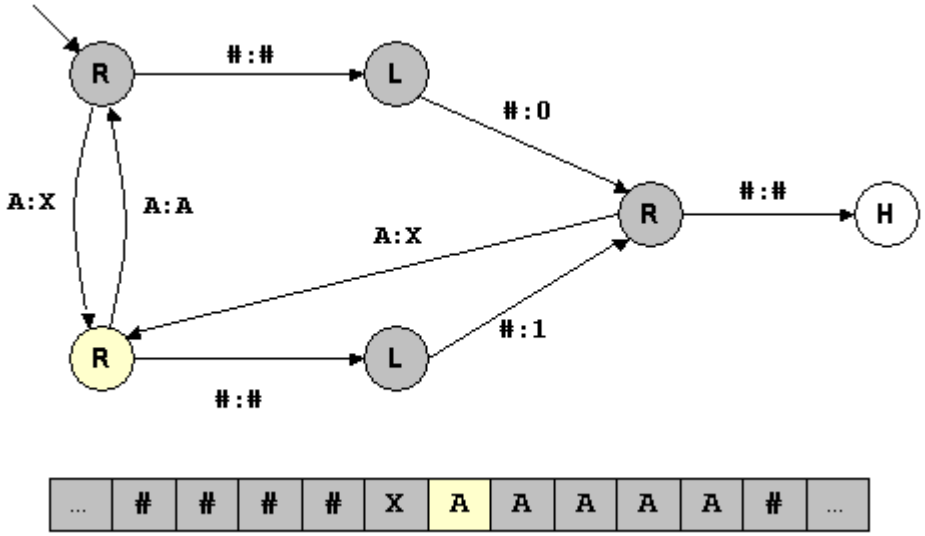
- ❑ update searchTwitter to return tweets
- ❑ add new Entry for search term (don't add a new button for this! Original map button should now read both entries, execute Twitter search, and show map)
- ❑ add new widget where tweet text can be displayed
- ❑ Rename readEntryAndShowMap (-> readEntriesSearchTwitterAndShowMap ?)
- ❑ update readEntriesSearchTwitterAndShowMap
  - ❑ retrieve tweets. I.e. call searchTwitter(…)
  - ❑ set "current tweet" to first one (*recommend:* currentTweetNum variable with index)
  - ❑ display current tweet
  - ❑ create and display map
- ❑ extract locations from tweets
- ❑ update getMapURL to display pins for tweets
- ❑ *recommended*: add new function, setCurrentTweet(num), to set and display current tweet
- ❑ widgets to change current tweet (if have setCurrentTweet, this is simple, like changing zoom level)
- ❑ new widget where tweet URL can be displayed
- ❑ new widget that will open browser to display web page for current URL
- ❑ *recommended*: add function, setCurrentTweetURL, to set and display current URL
- ❑ update setCurrentTweet function to set current tweet URL (via setCurrentTweetURL if you wrote it) to first URL associated with current tweet
- ❑ widgets to change current tweet URL

Some additional theory tidbits

- What if Python didn't have for/while loops, but had something else – e.g. "goto line i"
  - how does this affect what we can compute?
- Turing machines: important theoretical "universal computer"
  - memory: infinitely long "tape" of cells that can be blank or contain 0 or 1
  - program: a set of "states" and one fundamental operation:

    if state == q and tape cell == x, set cell to y, state to q', move





This simple computer can compute anything that is computable!

- A simulator
- A "real" one ☺

while vs. goto plus a simpler form of if statement that has no “body” – just a possible goto *linenum*

```
n = 0
while n < 100:
    sum = sum + n
    n = n + 1
print sum
```

```
1. n = 0
2. If ??? goto ???
3. sum = sum + n
4. n = n + 1
5. ???
6. print sum
```

while vs. goto

```
n = 0
while n < 100:
    sum = sum + n
    n = n + 1
print sum
```

```
1. n = 0
2. if n >= 100 goto 6
3. sum = sum + n
4. n = n + 1
5. goto  2
6. print sum
```

More CS theory

- P vs. NP - the biggest unsolved problem in computer science
  - there are many problems that we don't know *efficient* algorithms for, and don't even know whether or not efficient ones even exist
  - Hundreds of real-world have been shown to be *NP-complete*
    - this means that if you solve any one of them, we can efficiently convert that solution into a solution for the rest of them. *Solve one efficiently → solve all efficiently!*
    - Longest path (but not shortest path), graph coloring, subset sum, many puzzles (Sudoku (!, ?, ?, solvers, hardest??, solver???, solver!, thoughts), minesweeper, etc.), satisfiability of logic formulas, Traveling Salesperson (movie!?)
  - Solve it for extra practice this week == $1 million?

The Halting Problem

- it's important to know what we can and can't compute
- It turns out that we cannot create program that can check *all* other programs for infinite loops
- see, e.g., http://en.wikipedia.org/wiki/Halting_problem

- Informal proof next time
  - first: programs can analyze/create other programs: testProgramOnInput.py
  - why can't we create fully correct doesItHalt function? (doesItHalt.py)
  - To see why, consider function test in doesItHaltTest.py

Consider what happens when you do >>> test(“def test …”)

```
def test(programString):

  result = doesItHalt(programString, programString)

  if result == "No":
    print("I'm done (hey, in fact, I halt)")

  else:
    loopFinished = False
    while(not loopFinished):
      print ("I'm gonna live forever ...")
```