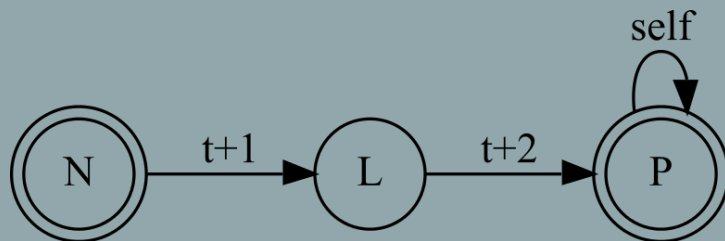


LING-362

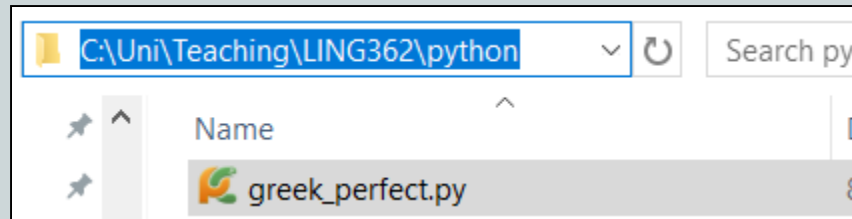
# Introduction to Natural Language Processing

Dictionaries &  
Eliza



# Files and file paths

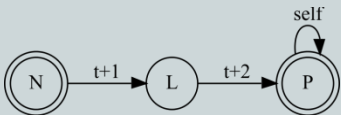
- ◎ Python scripts are always in some directory
- ◎ Can be executed in terminal when you are in that directory:



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\az364>cd c:\Uni\Teaching\LING362\python\

c:\Uni\Teaching\LING362\python>python greek_perfect.py
leluka
```



# Reading from a hard-wired file

---

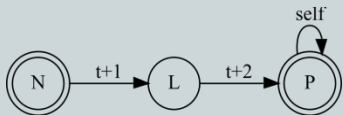
```
# text_about_phones = """ Let's talk ... """
```

```
text_about_phones = open("C:\\Uni\\phone_example.txt",encoding="utf8").read()
```

```
lines = text_about_phones.split("\n")
```

```
for line in lines:
```

```
...
```



# Recap: reading files

---

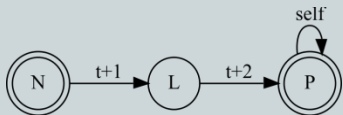
```
from argparse import ArgumentParser
```

```
parser = ArgumentParser()  
parser.add_argument("file")
```

```
options = parser.parse_args()  
phone_file = options.file
```

```
text_about_phones = open(phone_file).read()
```

...



# Recap: functions

---

```
def clean_number(number_text):
```

```
    """
```

docstring

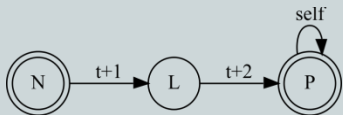
```
    :param number_text: string containing a phone  
                        number and special characters: (, +, )
```

```
    :return: cleaned: string without special characters
```

```
    """
```

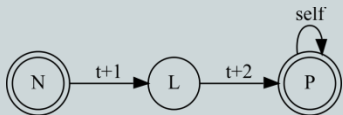
```
    cleaned = re.sub(r"[\+ \(\)]", "", number_text)
```

```
    return cleaned
```



# Phones!!

---



# How about recognizing the country?

- ⊙ A list of numbers is nice but...
- ⊙ In some cases, lists are not enough:
  - Suppose you want to know what country these numbers come from
  - You can recognize the prefix using `r'\+[0-9]+'`
  - It would be neat if we could '**look up**' what country this comes from
  - We'd need a **dictionary**



# Dictionaries

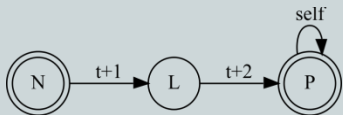
---

⊙ Dictionaries are like lists, but they don't map from index position to value:

- shopping\_list[1]

⊙ Instead they map **keys** to values:

- grocery\_type['broccoli']  
'vegetable'





# Dictionaries

---

- ◎ To make a dictionary, we specify the mapping of keys to values:

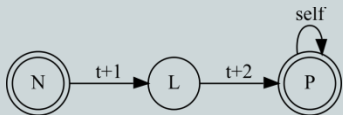
```
prefixes = {"+1":"US", "+972":"IL", "+63":"PH"}
```

- ◎ We can also add some later:

```
prefixes["+52"] = "MX"
```

- ◎ And access them:

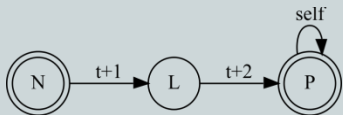
```
some_pref = "+52"  
country = prefixes[some_pref]
```



# Exercise for home

---

- ◎ You can try adding functionality to the number scraper (no need to submit):
    - Start by creating a dictionary of some prefixes:
      - `prefixes = {}`
      - `prefixes["52"] = "Mexico"`
    - Capture the +123 part of phone numbers if it appears
    - If you captured a prefix for the current number, output the country instead of the prefix, then the rest of the number:
      - `this_prefix = matcher2.group(1)`
      - `print(prefixes[this_prefix] + " " + number_without_prefix)`
- `'Mexico 015512345678'`



# How to check if you know the prefix?

- ◎ This can cause an error if you've never seen +345:

```
print(prefixes["345"])
```

Traceback (most recent call last):

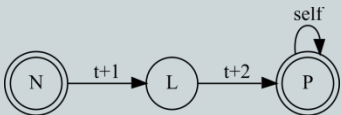
File "<stdin>", line 1, in <module>

KeyError: '345'

- ◎ Instead check first:

```
if "345" in prefixes:
```

```
    print(prefixes["345"])
```



# What else can we do with regex?

---

- ⦿ Regular expressions are a very powerful way of defining and manipulating patterns of strings
- ⦿ For some applications regex can get you very far
- ⦿ Is regex enough to make a talking computer?
- ⦿ What would constitute one in your opinion?



# The Turing Test

## ◎ Test devised by Alan Turing (1950)

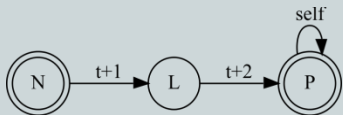
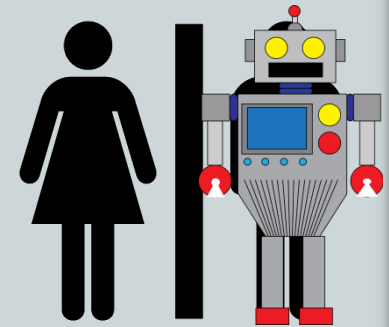
- See whether computers can communicate
- By proxy whether they are intelligent

## ◎ Original test:

- Computer substitutes participant in the "imitation game", in which a man/woman try to fool a judge about their gender

## ◎ Revised test:

- Person and computer try to convince judge that they are a human



# The Turing Test

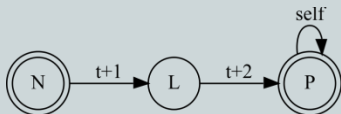
---

- ◎ The best known implementation is the **Loebner Prize** ([https://aisb.org.uk/new\\_site/?page\\_id=2](https://aisb.org.uk/new_site/?page_id=2))
- ◎ A little like a mini-nobel prize for NLG/dialogue systems
  - Running since 1990 (last year on hiatus)
  - Small prizes for 'best' program
  - Larger one time prizes:
    - \$25,000: judges can't distinguish program from human
    - \$100,000: same including A/V input



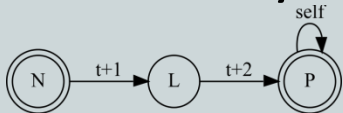
# Spot the not (blue is human)

- Hello, my name is Andrew. What's your name?
  - Hello Andrew nice to meet you, my name is Amos
  - Hello there. Your name is Andrew. My name is Millie.
- Why don't you tell me a little more about yourself ?
  - No, I am rather shy
  - My name is Millie and I am a human volunteer for the Loebner Prize. I live in Leeds and am single. I am interested in AI but don't have a chatbot myself. What about you?
- I am a researcher in Artificial Intelligence at Goldsmiths University, do you know what that is?
  - Of course, it is designation title is self explanatory!
  - Do you enjoy your research? I am a volunteer who has agreed to help with a Turing test.
- Will you tell me about your dreams?
  - Sure, what would you like to know?
  - I dream that I will pass my university exams.
- Can you rephrase that?
  - Yes I will, what would interest you about my dreams?
  - No, not really. What don't you understand?



# Spot the not (blue is human)

- What is my occupation?
  - You are one for changing topics. Aren't you a researcher?
  - Well you said your job was RESEARCHER.
- Can you tell me about a film you haven't seen?
  - If I haven't seen it, how would I tell you about it?
  - you have not seen = Are you asking about my have not seen?
- What do you think of Trump?
  - He will make a nice addition to the zoo
  - I read that over 50% were either concerned or scared about Donald Trump being President. if that's the case, why did they vote for him?

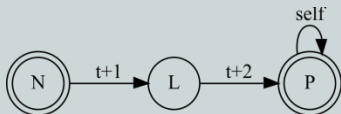




# Having conversations with a computer

- **Eliza** was the first 'chatbot' to convince some humans of its intelligence

- Basic 'Rogerian' therapist (*What do YOU think about that?*)
- Some basic linguistic awareness – recognize user expressions, conjugate and reflect them back
- Let's talk to Eliza and figure out what she knows and doesn't know about speaking English



```

=====
EEEEEE L      IIIIII ZZZZZZ   AAA
E       L      I      Z      A   A
E       L      I      Z      A   A
EEEEEE L      I      Z      A   A
E       L      I      Z      AAAAAA
E       L      I      Z      A   A
EEEEEE LLLLLL IIIIII ZZZZZZ   A   A
=====

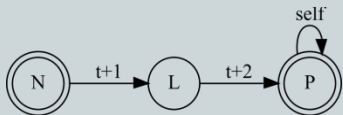
```

# Having conversations with a computer

---

## ◎ You can try it yourself!

- Download **eliza\_simplified.py** from Canvas
- In the terminal console run:  
*> python eliza\_simplified.py*



# Eliza: sample conversation

---

Therapist

-----

Talk to the program by typing in plain English, using normal upper-and lower-case letters and punctuation. Enter "quit" when done.

=====

Hello. How are you feeling today?

>I'm a little tired.

Why do you think you're a little tired?

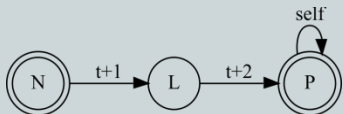
>I probably should have gone to bed earlier.

Why do you say that you probably should have gone to bed earlier?

>I think I don't sleep enough during the week.

Do you doubt you don't sleep enough during the week?

>No, definitely.



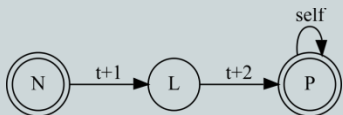
# How does Eliza work?

◎ Eliza is looking for certain patterns:

```
#-----  
# pats, the main response table. Each element of the list is a  
# two-element list; the first is a regexp, and the second is a  
# list of possible responses, with group-macros labelled as %  
#-----
```

```
pats = [  
    [r'I need (.*)',  
    [ r"Why do you need %?" ,  
      r"Would it really help you to get %?" ,  
      r"Are you sure you need %?" ] ] ,
```

...

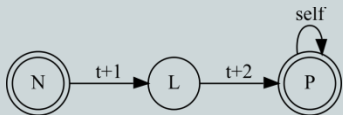


Reimplementation of Weizenbaum's code by Joe Strout, Jeff Epler, Jez Higgins in Python

# How does Eliza work?

## ◎ Uses certain substitutions:

```
#-----  
# reflections, a translation table used to convert things you say  
# into things the computer says back, e.g. "I am" --> "you are"  
#-----  
reflections = {  
    "am" : "are",  
    "was" : "were",  
    "i" : "you",  
    "i'd" : "you would",  
    "i've" : "you have",  
    "i'll" : "you will",  
    ...  
}
```



Reimplementation of Weizenbaum's code by Joe Strout, Jeff Epler, Jez Higgins in Python

# How does Eliza work?

---

```
#-----  
# command_interface  
#-----  
print("Therapist\n-----")  
print("Talk to the program by typing in plain English, using normal upper-")  
print('and lower-case letters and punctuation. Enter "quit" when done.')  
print('='*72)  
print("Hello. How are you feeling today?")  
s = ""
```



Reimplementation of Weizenbaum's code by Joe Strout, Jeff Epler, Jez Higgins in Python

# How does Eliza work?

*# Make an empty dictionary called mappings*

```
mappings = {}
```

```
for pattern in pats: # Go through patterns
```

```
    find = pattern[0] # The first item in each pattern is the regex to find
```

```
    replacements = pattern[1] # The second item is itself a list of possible replacements
```

```
    mappings[find] = replacements
```

```
while s != "quit": # While loop is a bit like a for loop
```

```
    s = input(">")
```

```
    if len(s) < 1:
```

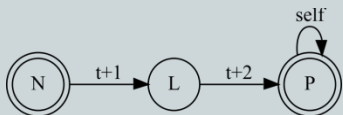
```
        s = "quit"
```

```
        print(s)
```

```
    s = re.sub(r"[\.!]$", "", s) # Remove trailing punctuation
```

```
    response = respond(s, mappings, reflections)
```

```
    print(response)
```



Reimplementation of Weizenbaum's code by Joe Strout, Jeff Epler, Jez Higgins in Python

# Did you find any problems with Eliza?



## Problems:

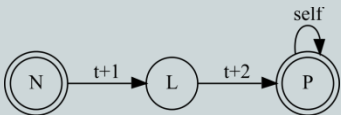
Morphology  
Syntax  
Semantics  
Pragmatics  
Memory?

## Solutions:

More patterns  
More reflections  
Completely new mechanisms

>I want to challenge **myself**

If you got to challenge **myself**, then what would you do?





# Making Eliza better

---

## ◎ Some ideas:

- Talking about brothers and sisters (be conservative!)
- Support for reflexive pronouns to fix this:

>I want to kill **myself**

If you got to kill **myself**, then what would you do?

- A pattern to make Eliza repeat what you said:

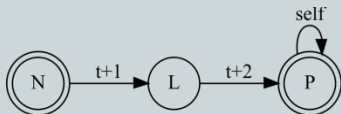
> Just tell me "you're beautiful".

you're beautiful

- ... more clever stuff??

## ◎ Brainstorm at:

<https://corpling.uis.georgetown.edu/etherpad/p/eliza>



# At home

---

- ◎ Make sure you've submitted the Pig Latin assignment!
- ◎ Please read the text about **finite-state methods** from Jurafsky & Martin (2008)
- ◎ Consider tackling the phone script challenge to extract international prefixes! (no need to submit, solution will be discussed on Wednesday)

