# Basic Programming Terminology

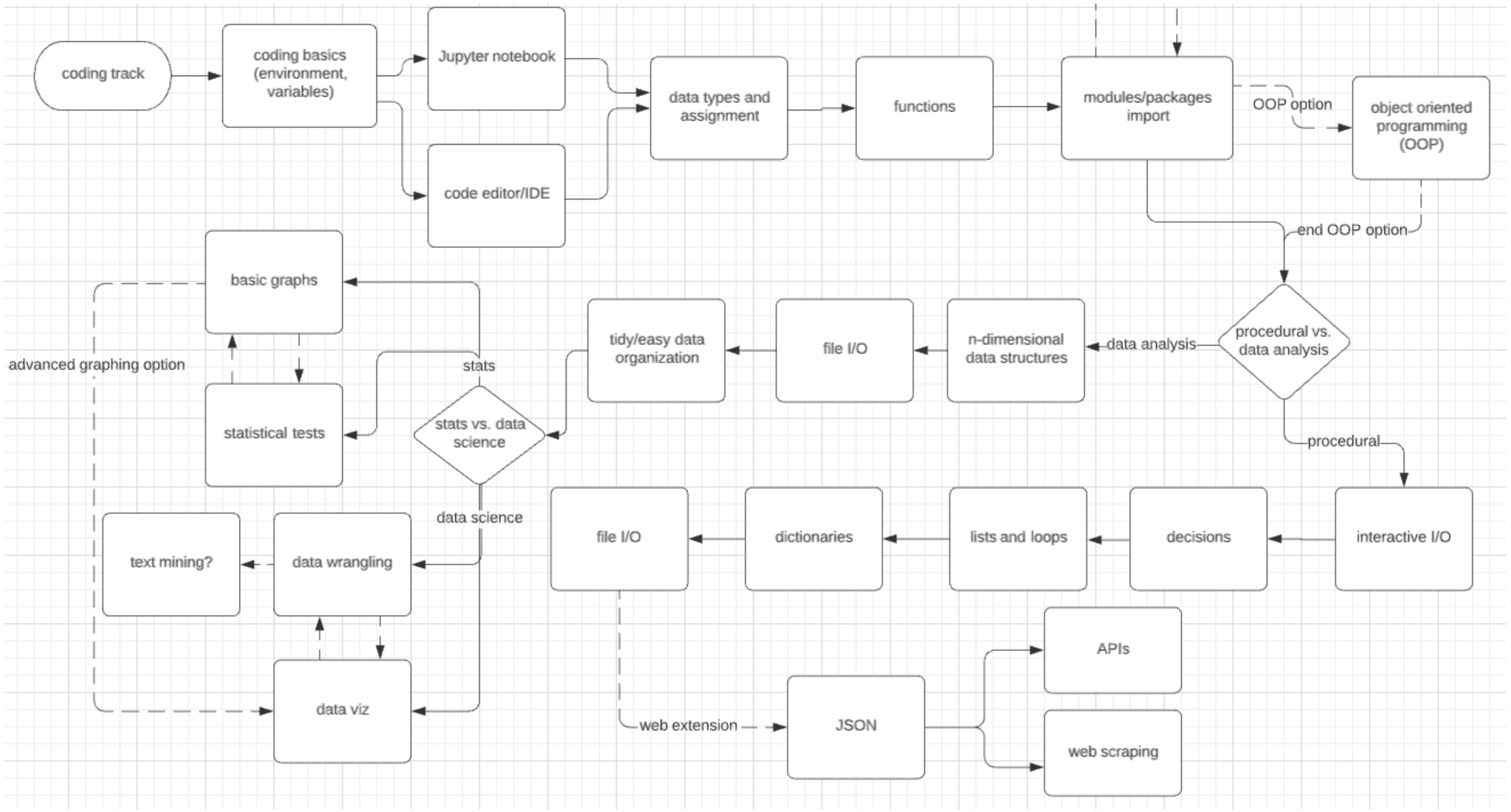
Presenter: Steve Baskauf
steve.baskauf@vanderbilt.edu

## Digital Scholarship and Communications Office (DiSC)

- Unit of the Vanderbilt Libraries
- Support for data best practices  (DMP tool, repositories), GIS, copyright, Linked Data (including Wikidata), tools (GitHub, ORCID, Open Science Framework, etc.), and Open Access publishing.
- Offers on-demand educational programming, consultations, web resources
- Typically offering lessons on Python, R, and GIS
- More online at: **vanderbi.lt/disc**
- Email: disc@vanderbilt.edu

vanderbi.lt/codegraf

# What is CodeGraf?

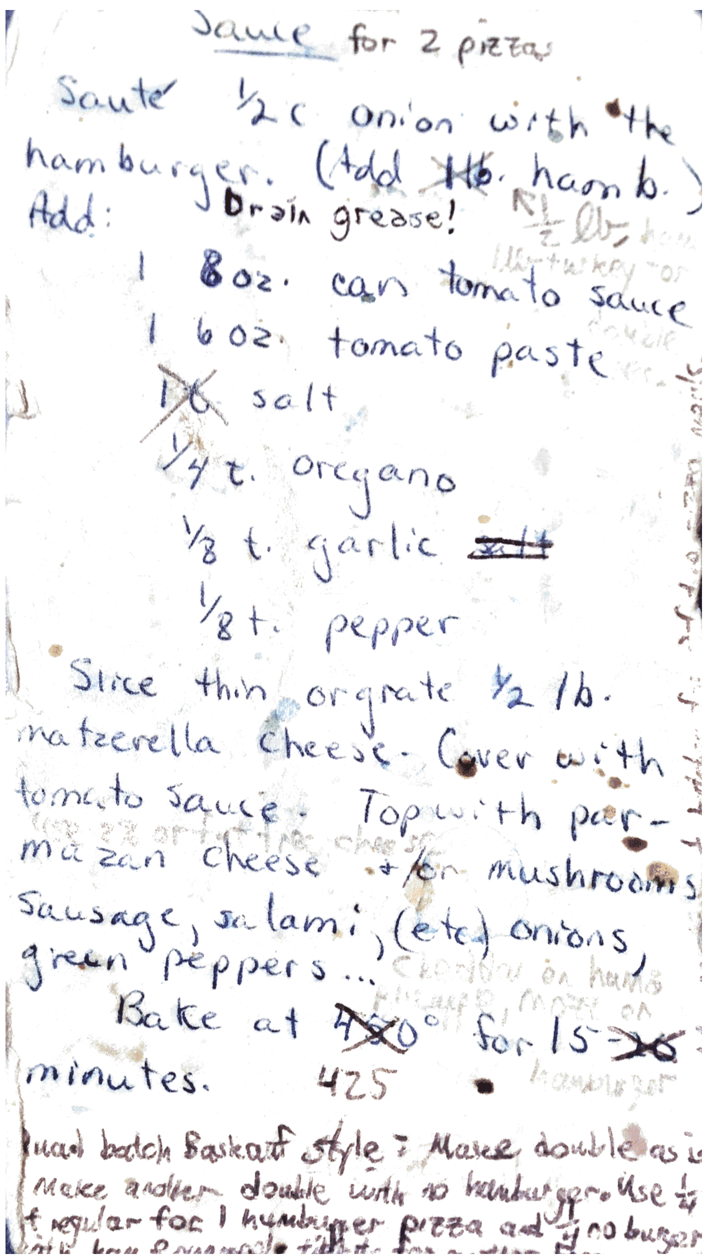


# Terms for programs

DiSC DIGITAL SCHOLARSHIP AND COMMUNICATIONS

Jean & Alexander Heard LIBRARIES

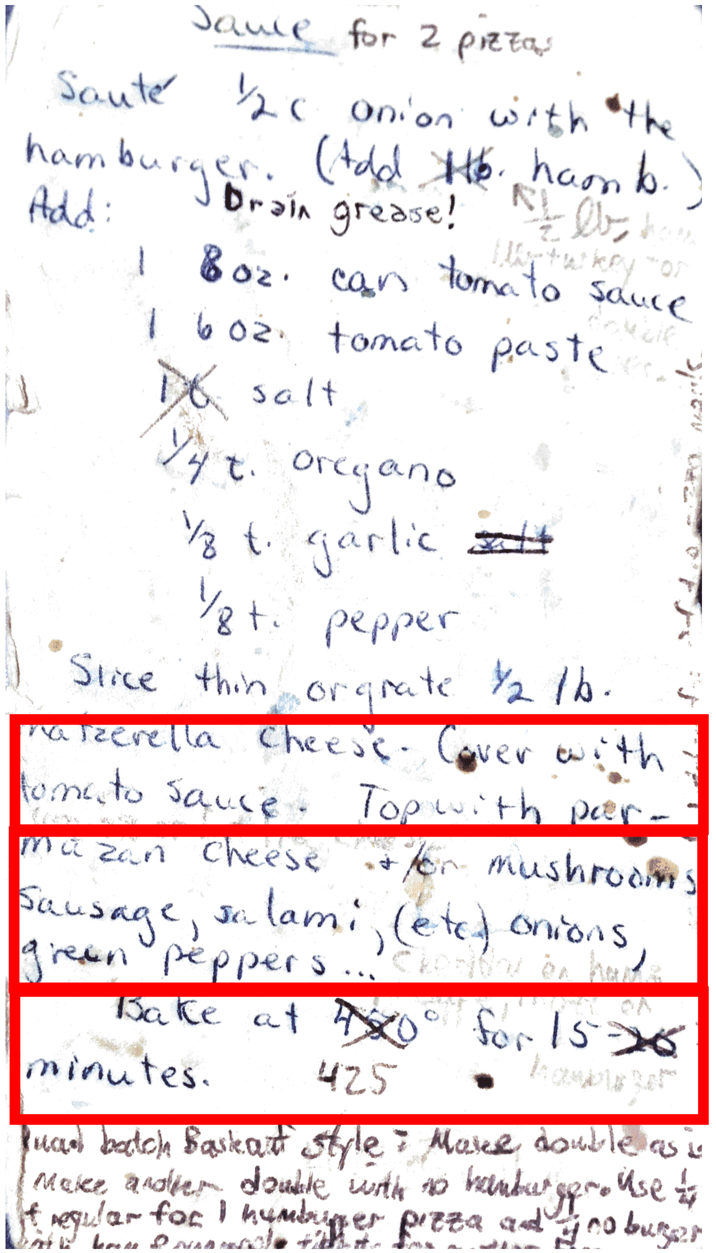
# a program


Sauce for 2 pizzas
Sauté ½ c onion with the hamburger. (Add ~~1 lb.~~ ½ lb hamb.)
Drain grease!
Add:
1 8 oz. can tomato sauce
1 6 oz. tomato paste
~~1 t.~~ salt
¼ t. oregano
⅛ t. garlic ~~salt~~
⅛ t. pepper
Slice thin or grate ½ lb. matzerella cheese. Cover with tomato sauce. Top with parmazan cheese +/or mushrooms, sausage, salami, (etc) onions, green peppers...
Bake at ~~450~~ 425° for 15 ~~20~~ minutes.

Quad batch Baskauf style: Make double as is
Make another double with no hamburger. Use ½
of regular for 1 hamburger pizza and ½ no burger

- a generic term for a complete set of instructions that does something

```
#data = readDict('vanderbilt_units.csv')
#print(json.dumps(data,indent=2))

while True: # infinite loop
    print('Time checked:',
datetime.datetime.utcnow().isoformat())
    with open('last_run.txt', 'rt',
encoding='utf-8') as fileObject:
        date_last_run = fileObject.read()
    print('Date last run:', date_last_run)

    date_now_utc = generate_utc_date()
    print('UTC date now is:', date_now_utc)

    if date_now_utc > date_last_run:
        run_all_queries()

        # Update the date last run
        with open('last_run.txt', 'wt',
encoding='utf-8') as fileObject:

fileObject.write(generate_utc_date())

        print('done')
    print()

    # wait an hour before checking again
    sleep(3600)
```



Sauce for 2 pizzas

Saute ½ c onion with the hamburger. (Add ~~1 lb.~~ hamb.) ½ lb.
Add: Drain grease!

1 8 oz. can tomato sauce
1 6 oz. tomato paste
~~1 t.~~ salt
¼ t. oregano
⅛ t. garlic ~~salt~~
⅛ t. pepper

Slice thin or grate ½ lb. mazzerella cheese. Cover with tomato sauce. Top with parmazan cheese +/or mushrooms, sausage, salami, (etc) onions, green peppers...

Bake at ~~450~~° 425 for 15-~~20~~ minutes.

quad batch Baskett style: Make double as is. Make another double with no hamburger. Use ¼ regular for 1 hamburger pizza and ¼ no burger

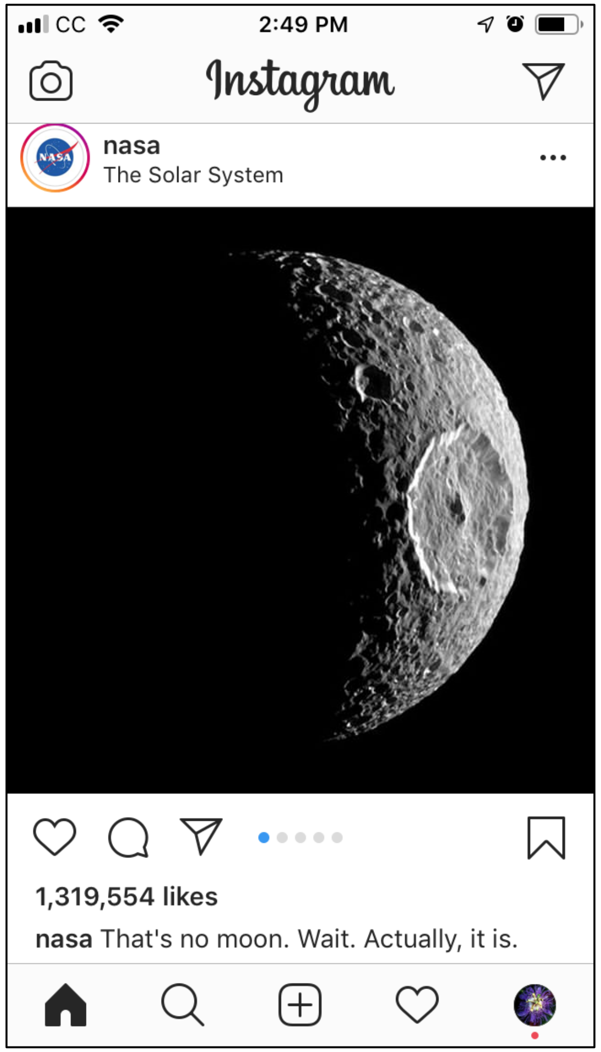
# code

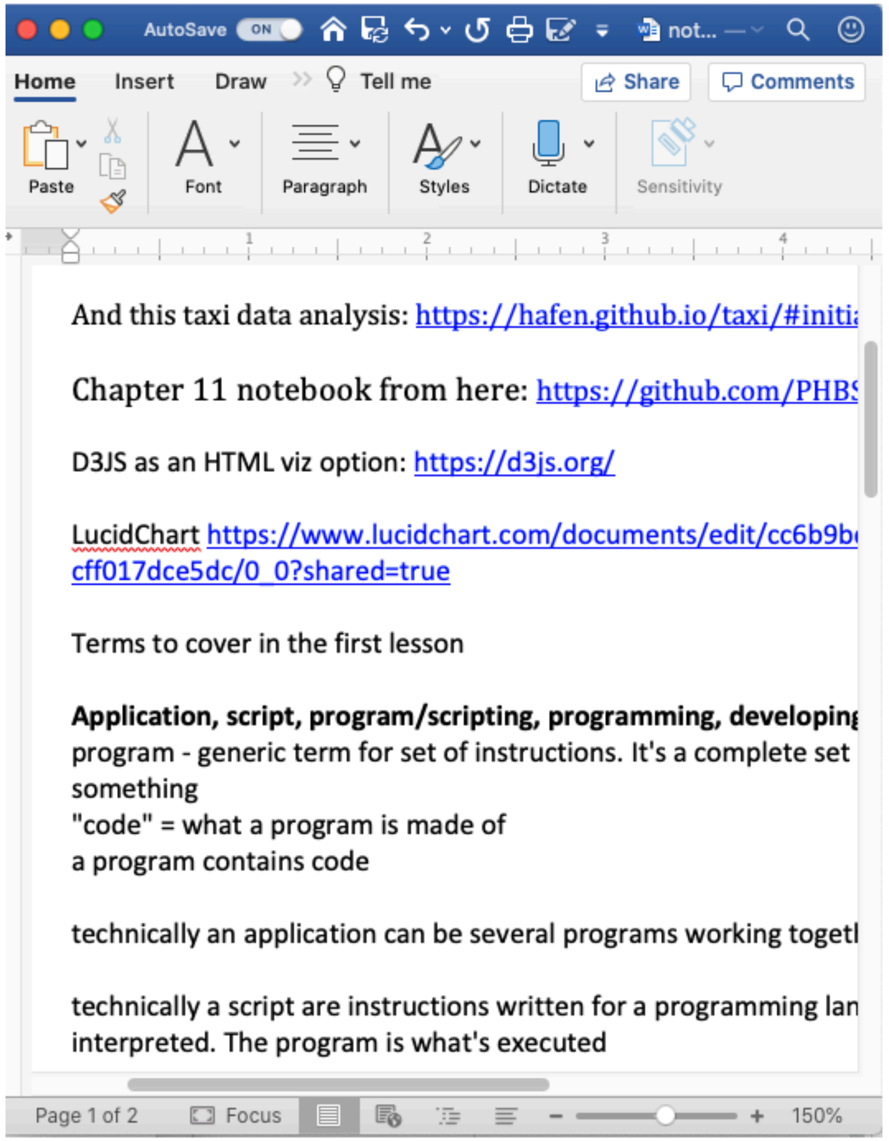
- instructions that make up a program

```
#data = readDict('vanderbilt_units.csv')
#print(json.dumps(data,indent=2))

while True: # infinite loop
    print('Time checked:',
datetime.datetime.utcnow().isoformat())
    with open('last_run.txt', 'rt', encoding='utf-
8') as fileObject:
        date_last_run = fileObject.read()
    print('Date last run:', date_last_run)

    date_now_utc = generate_utc_date()
    print('UTC date now is:', date_now_utc)

    if date_now_utc > date_last_run:
        run_all_queries()

        # Update the date last run
        with open('last_run.txt', 'wt',
encoding='utf-8') as fileObject:

            fileObject.write(generate_utc_date())

        print('done')
    print()

    # wait an hour before checking again
    sleep(3600)
```

## an application (app)

- one or several programs working together for the end user





And this taxi data analysis: https://hafen.github.io/taxi/#initi

Chapter 11 notebook from here: https://github.com/PHBS

D3JS as an HTML viz option: https://d3js.org/

LucidChart https://www.lucidchart.com/documents/edit/cc6b9b cff017dce5dc/0_0?shared=true

Terms to cover in the first lesson

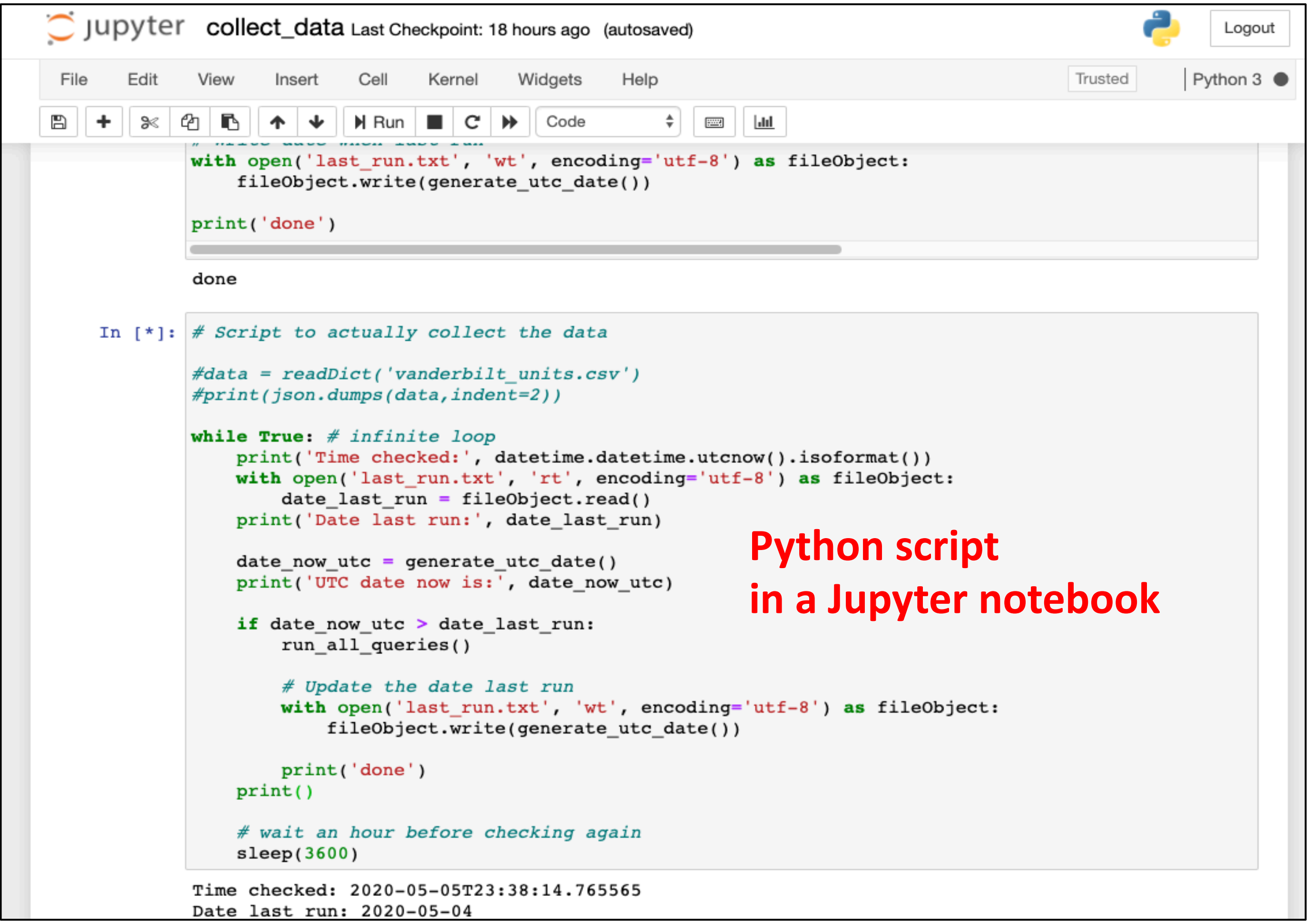
**Application, script, program/scripting, programming, developing**
program - generic term for set of instructions. It's a complete set
something
"code" = what a program is made of
a program contains code

technically an application can be several programs working togetl

technically a script are instructions written for a programming lan
interpreted. The program is what's executed

# a script

- instructions in a programming language that need to be interpreted



```
with open('last_run.txt', 'wt', encoding='utf-8') as fileObject:
    fileObject.write(generate_utc_date())

print('done')
```

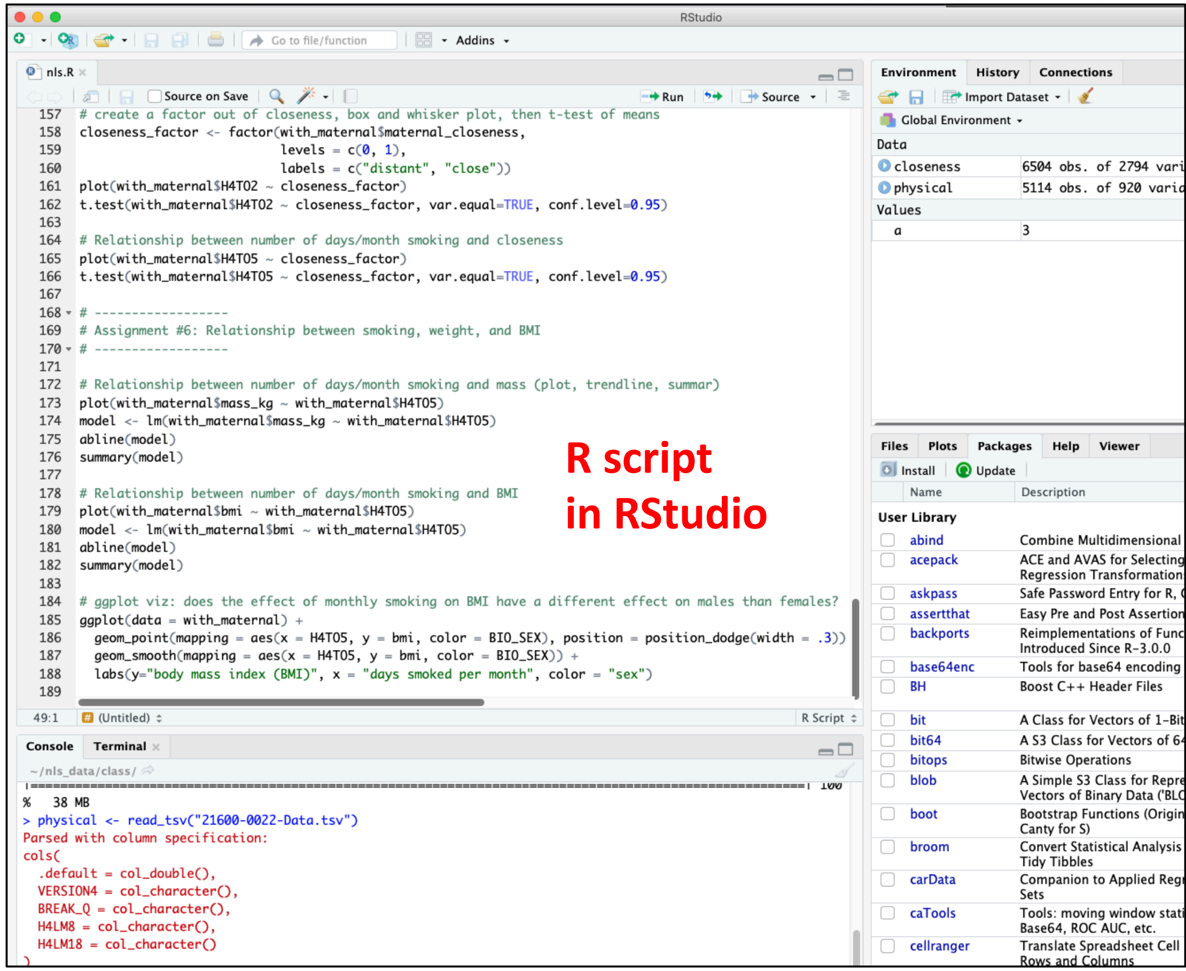
```
done
```

In [*]:

```
# Script to actually collect the data

#data = readDict('vanderbilt_units.csv')
#print(json.dumps(data,indent=2))

while True: # infinite loop
    print('Time checked:', datetime.datetime.utcnow().isoformat())
    with open('last_run.txt', 'rt', encoding='utf-8') as fileObject:
        date_last_run = fileObject.read()
    print('Date last run:', date_last_run)

    date_now_utc = generate_utc_date()
    print('UTC date now is:', date_now_utc)

    if date_now_utc > date_last_run:
        run_all_queries()

        # Update the date last run
        with open('last_run.txt', 'wt', encoding='utf-8') as fileObject:
            fileObject.write(generate_utc_date())

        print('done')
    print()

    # wait an hour before checking again
    sleep(3600)
```

```
Time checked: 2020-05-05T23:38:14.765565
Date last run: 2020-05-04
```

Python script in a Jupyter notebook



```
# create a factor out of closeness, box and whisker plot, then t-test of means
closeness_factor <- factor(with_maternal$maternal_closeness,
                           levels = c(0, 1),
                           labels = c("distant", "close"))
plot(with_maternal$H4TO2 ~ closeness_factor)
t.test(with_maternal$H4TO2 ~ closeness_factor, var.equal=TRUE, conf.level=0.95)

# Relationship between number of days/month smoking and closeness
plot(with_maternal$H4TO5 ~ closeness_factor)
t.test(with_maternal$H4TO5 ~ closeness_factor, var.equal=TRUE, conf.level=0.95)

# -----------------
# Assignment #6: Relationship between smoking, weight, and BMI
# -----------------

# Relationship between number of days/month smoking and mass (plot, trendline, summar)
plot(with_maternal$mass_kg ~ with_maternal$H4TO5)
model <- lm(with_maternal$mass_kg ~ with_maternal$H4TO5)
abline(model)
summary(model)

# Relationship between number of days/month smoking and BMI
plot(with_maternal$bmi ~ with_maternal$H4TO5)
model <- lm(with_maternal$bmi ~ with_maternal$H4TO5)
abline(model)
summary(model)

# ggplot viz: does the effect of monthly smoking on BMI have a different effect on males than females?
ggplot(data = with_maternal) +
  geom_point(mapping = aes(x = H4TO5, y = bmi, color = BIO_SEX), position = position_dodge(width = .3))
  geom_smooth(mapping = aes(x = H4TO5, y = bmi, color = BIO_SEX)) +
  labs(y="body mass index (BMI)", x = "days smoked per month", color = "sex")
```

```
|==========================================================================================| 100
%   38 MB
> physical <- read_tsv("21600-0022-Data.tsv")
Parsed with column specification:
cols(
  .default = col_double(),
  VERSION4 = col_character(),
  BREAK_Q = col_character(),
  H4LM8 = col_character(),
  H4LM18 = col_character()
)
```

| Data | |
|---|---|
| closeness | 6504 obs. of 2794 vari |
| physical | 5114 obs. of 920 varia |
| **Values** | |
| a | 3 |

| Name | Description |
|---|---|
| **User Library** | |
| abind | Combine Multidimensional |
| acepack | ACE and AVAS for Selecting Regression Transformation |
| askpass | Safe Password Entry for R, ( |
| assertthat | Easy Pre and Post Assertion |
| backports | Reimplementations of Func Introduced Since R-3.0.0 |
| base64enc | Tools for base64 encoding |
| BH | Boost C++ Header Files |
| bit | A Class for Vectors of 1-Bit |
| bit64 | A S3 Class for Vectors of 64 |
| bitops | Bitwise Operations |
| blob | A Simple S3 Class for Repre Vectors of Binary Data ('BLC |
| boot | Bootstrap Functions (Origin Canty for S) |
| broom | Convert Statistical Analysis Tidy Tibbles |
| carData | Companion to Applied Regr Sets |
| caTools | Tools: moving window stati Base64, ROC AUC, etc. |
| cellranger | Translate Spreadsheet Cell Rows and Columns |

R script in RStudio

## Command-line interfaces (CLI)

DISC DIGITAL SCHOLARSHIP AND COMMUNICATIONS

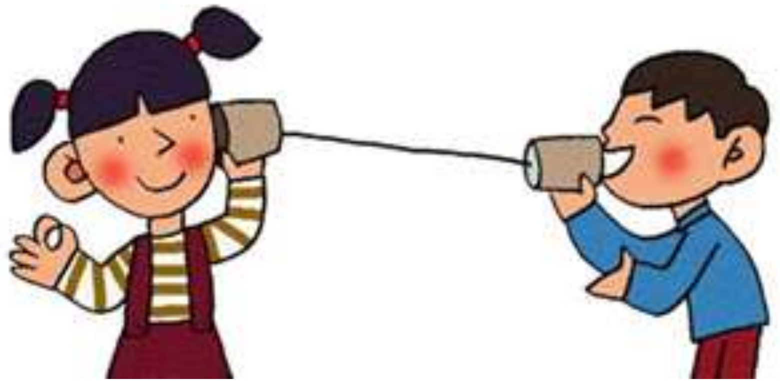
Jean & Alexander Heard LIBRARIES

## consoles

- A **console** is a program that sends text commands and receives text output

- The typical console for Macs is called **Terminal**
- The typical console for Windows is called **Command prompt**

## the shell



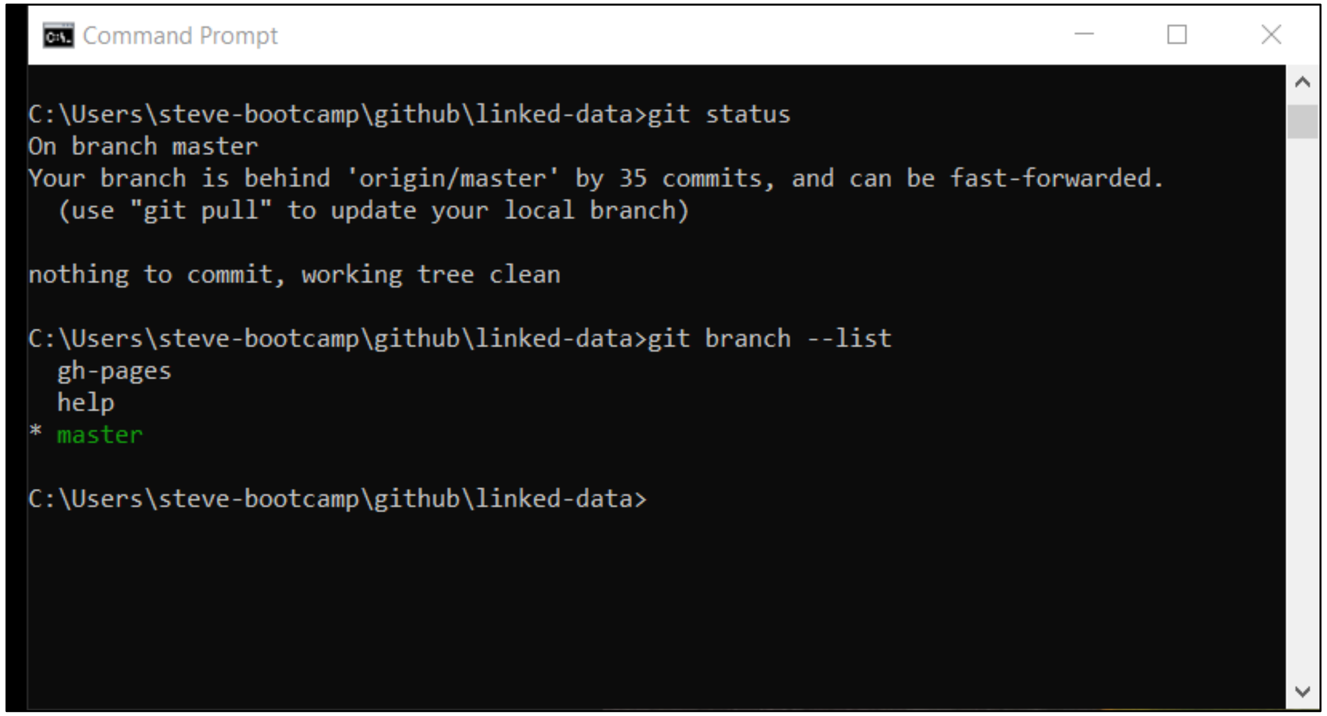
- A **shell** is the program that receives and processes text commands from a console

- **bash** is a shell that processes commands in the Linux operating system
- **Python** and **R** both have shells

Image from https://www.clipart.email/

# CLI vs. GUI

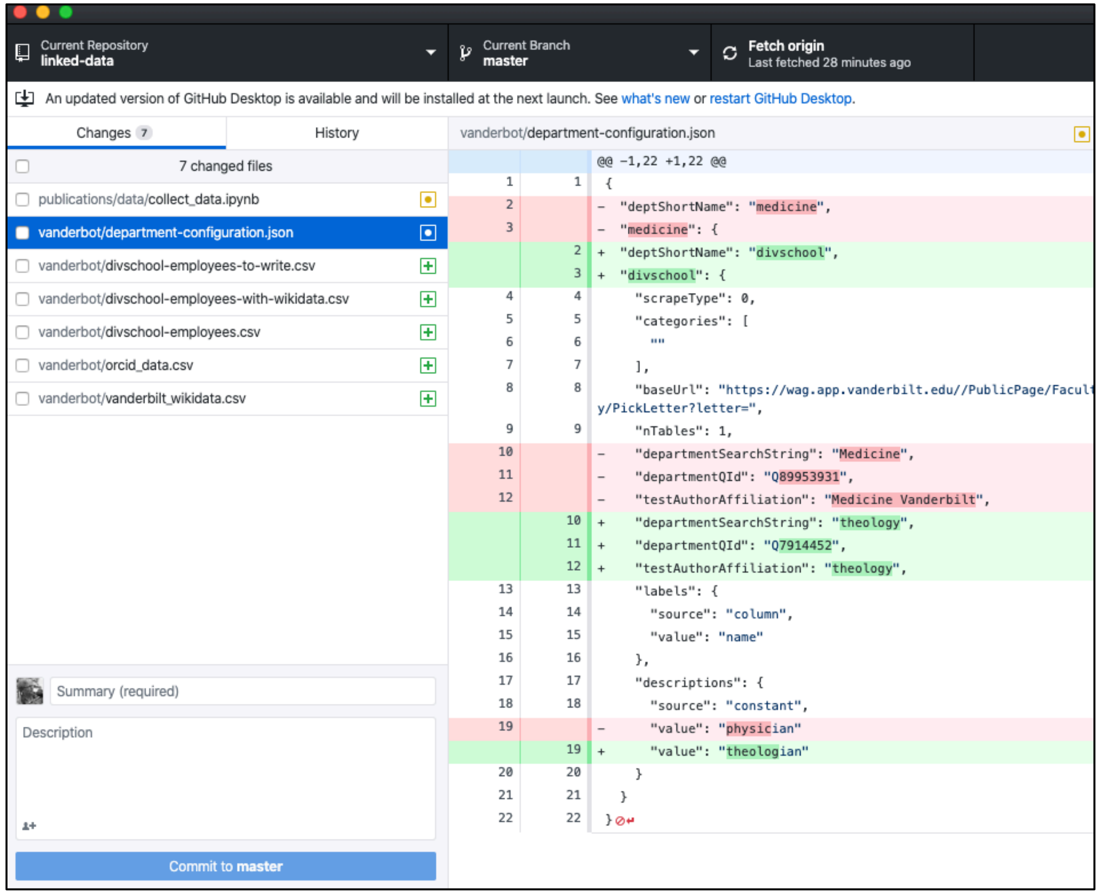
- A **command-line interface** (CLI) is basically synonymous with a shell.
- A CLI is in contrast to a **graphical user interface** (GUI)



Command Prompt

```
C:\Users\steve-bootcamp\github\linked-data>git status
On branch master
Your branch is behind 'origin/master' by 35 commits, and can be fast-forwarded.
  (use "git pull" to update your local branch)

nothing to commit, working tree clean

C:\Users\steve-bootcamp\github\linked-data>git branch --list
  gh-pages
  help
* master

C:\Users\steve-bootcamp\github\linked-data>
```

command-line interface for Git



An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.

publications/data/collect_data.ipynb
vanderbot/department-configuration.json
vanderbot/divschool-employees-to-write.csv
vanderbot/divschool-employees-with-wikidata.csv
vanderbot/divschool-employees.csv
vanderbot/orcid_data.csv
vanderbot/vanderbilt_wikidata.csv

vanderbot/department-configuration.json

```
@@ -1,22 +1,22 @@
 {
-  "deptShortName": "medicine",
-  "medicine": {
+  "deptShortName": "divschool",
+  "divschool": {
     "scrapeType": 0,
     "categories": [
       ""
     ],
     "baseUrl": "https://wag.app.vanderbilt.edu//PublicPage/Faculty/PickLetter?letter=",
     "nTables": 1,
-    "departmentSearchString": "Medicine",
-    "departmentQId": "Q89953931",
-    "testAuthorAffiliation": "Medicine Vanderbilt",
+    "departmentSearchString": "theology",
+    "departmentQId": "Q7914452",
+    "testAuthorAffiliation": "theology",
     "labels": {
       "source": "column",
       "value": "name"
     },
     "descriptions": {
       "source": "constant",
-      "value": "physician"
+      "value": "theologian"
     }
   }
 }
```

graphical user interface for Git

# Variables and objects

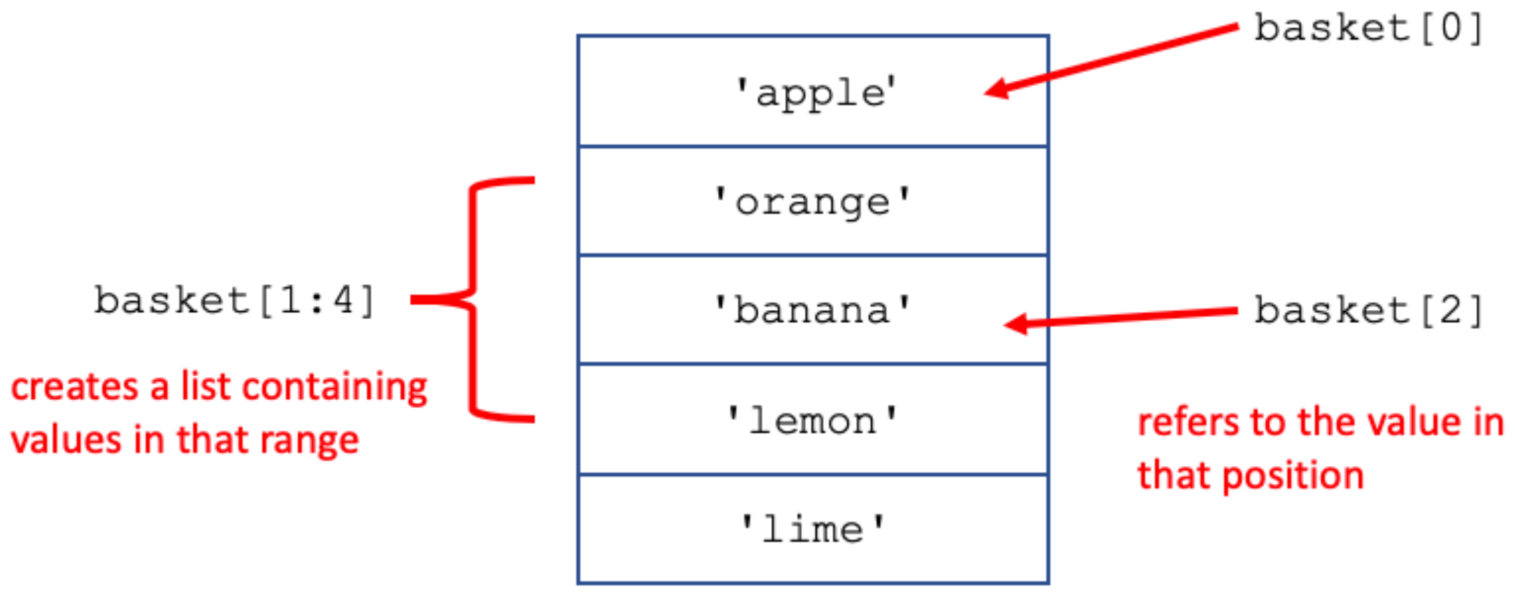
DISC DIGITAL SCHOLARSHIP AND COMMUNICATIONS

Jean & Alexander Heard LIBRARIES

# variables

- **variables** are named locations where we store data
- example a variable named "basket" that is a list that can store multiple alphanumeric strings

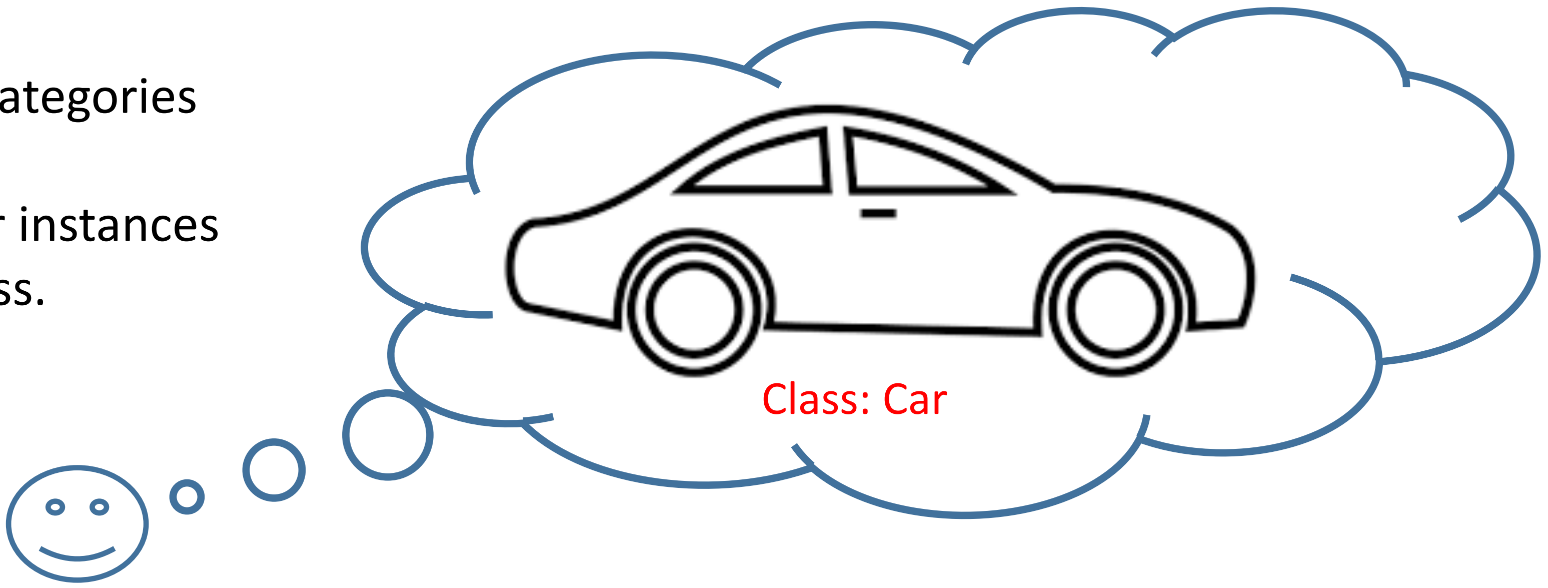
```
basket = ['apple', 'orange', 'banana', 'lemon', 'lime']
```



basket[0]
'apple'
'orange'
basket[1:4]
'banana'
basket[2]
creates a list containing values in that range
'lemon'
refers to the value in that position
'lime'

# classes and objects

**Classes** are abstract categories of things.
**Objects** are particular instances or individuals of a class.



object: toyotaPrius

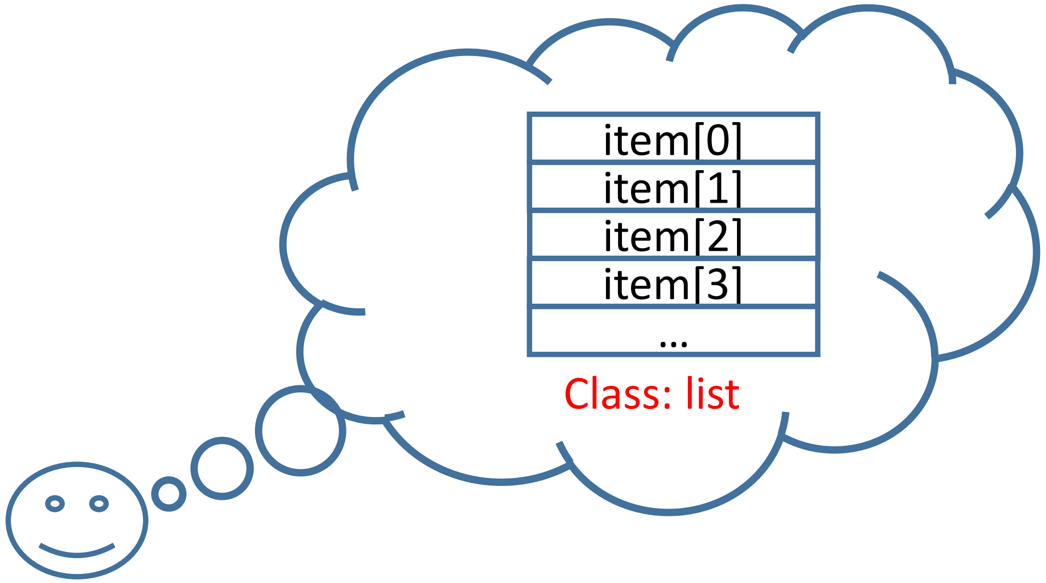
object: ferrari

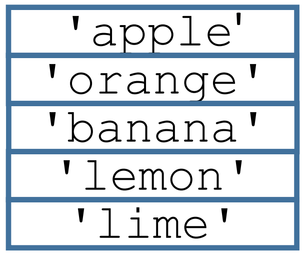
object: volkswagenBeetle

# classes and objects

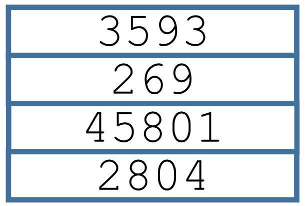
**Classes** are abstract categories of data structures.
**Objects** are particular data structures.
The **type** of an object is the class to which it belongs.

There are technical distinctions between **variables** and **named objects** but we will use them interchangeably.

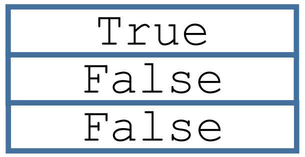




object name: **`fruits`**
type: **`list`**



object name: **`ids`**
type: **`list`**



object name: **`in_stock`**
type: **`list`**

## Executing code

DISC DIGITAL SCHOLARSHIP AND COMMUNICATIONS

Jean & Alexander Heard LIBRARIES

## Statements

- Code is made up of **statements**
- A statement performs a particular action.
- A "line of code" is roughly the same as a statement

```
while True: # infinite loop
    print('Time checked:', datetime.datetime.utcnow().isoformat())
    with open('last_run.txt', 'rt', encoding='utf-8') as fileObject:
        date_last_run = fileObject.read()
    print('Date last run:', date_last_run)

    date_now_utc = generate_utc_date()
    print('UTC date now is:', date_now_utc)

    if date_now_utc > date_last_run:
        run_all_queries()

        # Update the date last run
        with open('last_run.txt', 'wt', encoding='utf-8') as fileObject:
            fileObject.write(generate_utc_date())

        print('done')
    print()

    # wait an hour before checking again
    sleep(3600)
```

**a statement**

**R code**

```
# Relationship between age of starting smoking and closeness
# create a factor out of closeness, box and whisker plot, then t-test of means
closeness_factor <- factor(with_maternal$maternal_closeness,
                           levels = c(0, 1),
                           labels = c("distant", "close"))
plot(with_maternal$H4TO2 ~ closeness_factor)
t.test(with_maternal$H4TO2 ~ closeness_factor, var.equal=TRUE, conf.level=0.95)

# Relationship between number of days/month smoking and closeness
plot(with_maternal$H4TO5 ~ closeness_factor)
t.test(with_maternal$H4TO5 ~ closeness_factor, var.equal=TRUE, conf.level=0.95)
```

**a statement**

**Python code**

# interactive vs. script mode

- In **interactive mode**, one statement is run at a time in the shell. Immediate feedback is given after each line.
- In **script mode**, the entire script is run at once. Feedback is only given when explicitly required by the script.
- Both R and Python can be run in either mode.

## Writing code with an editor

DISC DIGITAL SCHOLARSHIP AND COMMUNICATIONS

Jean & Alexander Heard LIBRARIES

## Code editors

- **Code editors** are text editors on steroids.
- They are "aware" of the language in which you are coding.
- They generally have **syntax checking** and highlighting.
- They may help with automatic **formatting**.
- Some code editors have capabilities for running the code and are essentially integrated development environments (IDEs).

# Remote Support for Teaching and Research Needs

- Access to digital collections 24/7
- Skype consultations with your subject librarian
- Ask a Librarian: an easy way to submit a question via email
- Live chat available from the Library home page

Jean & Alexander Heard
LIBRARIES

## NEED HELP? ASK A LIBRARIAN!

https://www.library.vanderbilt.edu/ask-librarian.php

Jean & Alexander Heard
LIBRARIES