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





man botch Baskart Style 7 Masse double as in Make another double with no handwriger. Use of thegular for 1 humburger preza and if no burger

a program

 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)



code

 instructions that make up a program #data = readDict('vanderbilt_units.csv')
#print(json.dumps(data,indent=2))

```
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



🧧 🔵 🔹 AutoSave 🔍 🏠 🏫 🦕 かくび 🖨 🐼 🔻 🐴 not — - く 🙂
Home Insert Draw >>> Q Tell me
Paste V Font Paragraph Styles Dictate Sensitivity
*
And this taxi data analysis: <u>https://hafen.github.io/taxi/#initia</u>
Chapter 11 notebook from here: <u>https://github.com/PHB5</u>
D3JS as an HTML viz option: <u>https://d3js.org/</u>
LucidChart https://www.lucidchart.com/documents/edit/cc6b9bo 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 toget
technically a script are instructions written for a programming lan interpreted. The program is what's executed
Page 1 of 2 D Focus R 3 = + 150%

a script

• instructions in a programming language that need to be interpreted

Pier Edit Were wert Out Kernel Wioges Heip Tanta Pierter Pier Edit Were wert Out Kernel Wioges Heip Tanta Pierter Pierter Pierter	C JUDYTET collect data Last Checkpoint: 18	hours ago (autosaved)	2	Logout			RStudio		
File East Vew Manum Cold Manum Manu						Go to file/function			
<pre></pre>	File Edit View Insert Cell Kernel	Widgets Help	Trusted	Python 3	Inls.R ×			Environment His	story Connections
<pre></pre>						Source on Save Q / -	🕂 Run 🐤 📑 Source 👻 🗏	🖆 📙 🖙 Impo	ortDataset 👻 🔏
<pre>with open("later_units", 'ty", encoding='ut_6') as fileObject: fileObject.write("generate_ute_date()) print('lose') done In ['1: # Script to actually collect the date #data = readDict('vanderbilt_units.csr') #print('lose characteristic = fileObject: date_late_units, 'ty", encoding='ut_6') as fileObject: date_late_now_ute = generate_ute_date() print('lose') #uti deno('late_units', 'ty", encoding='ut_6') as fileObject: date_late_now_ute = denorse.deta() print('lose') #uti deno('late_units', 'ty", encoding='ut_6') as fileObject: date_late_now_ute = denorse.deta() print('lose') #uti denorse ideno('lose') print('lose') #uti denorse ideno('lose') #uti denorse idenorse idenorse idenorse idenorse idenorse idenorse idenorse idenorse idenorse idenorse idenorse idenorse in a Juppyter notebook #uti denorse idenorse id</pre>	🕒 + ≫ 4?1 🖪 🛧 🕹 Ni Run 🔳 C 🕨	Code 💠 📼 📶			157 # Creater	ss factor <- factor(with maternal\$maternal closeness.	en t-test of means	Global Environm	ient -
<pre>sith open('latt_run,ckt', 'vt', encoding='utf_8') as fileObject: fileObject.write(generate_ute_date()) print('dons') done</pre>					159	levels = c(0, 1),		Data	
<pre>fileObject.write(generate_utc_date()) print('done') done In ('): # Script to actually collideration of don/host scatter and classes: #data = readDict('vanderDilt_units.cw') #dita = readDict('vanderDil</pre>	with open('last_run.txt', 'w	t', encoding='utf-8') as fileObject:			160	<pre>labels = c("distant", "close"))</pre>		Closeness	6504 obs. of 2794 var
<pre>print('done') done In (*): # Script to actually collect the data #data = readbiot('vanderbilt_units.cov') #print('done') print('lime thebedit', thetime.duttime.uktorow().imformat()) while True: # infinite loop print('lime thebedit', thetime.duttime.uktorow().imformat()) print('Disc these thetic, readbiot('vanderbilt_units.cov') #print('lime thebedit', thetime.duttime.uktorow().imformat()) while True: # infinite loop print('lime thebedit', thetime.duttime.uktorow().imformat()) while True: # infinite loop print('Disc these thetic, readbiot('vanderbilt_units.cov') #print('Disc these thetic, trading 'utf=#') as fileobject:</pre>	fileObject.write(generate	e_utc_date())			161 plot(wit	th_maternal\$H4TO2 ~ closeness_factor)		D physical	5114 obs. of 920 varia
<pre>print('done') done In (*): # Script to actually collect the data #data = readDist('vanderbil: units.csv') #finit('son.dup=(data:,indent=2)) while a rune i initiabilitie .csv') #finit('son.dup=(data:,indent=2)) #finit('son.dup=(dat</pre>					162 t.test(w	with_maternal\$H4TO2 ~ closeness_factor, var.equal=TRU	, conf.level=0.95)	Values	
<pre>dome in the print(ignon.dump(data,indent=2)) while frue: # infinite loop print(ignon.dump(data,indent=2)) print(ignon.dump(data,indent=2))</pre>	<pre>print('done')</pre>				164 # Relati	ionship between number of days/month smoking and close	aness.	a	3
done in [*]: # Soript to actually collect the data #data = readDict('vanderbilt_units.csr') #print('line checkedi', data indent=2)) while True: * infinite toop print('line checkedi', data indent=2)) while print(*line * infinite toop print('line checkedi', data indent=2)) while print(*line * infinite toop print('line checkedi', data indent=2)) while print(*line * infinite toop print('line checkedi', data indent=2)) Adta = now_ucc = generate_ucc_date() print('loce last run) id date_now_ucc = generate_ucc_date() print('loce data indent run ''Update the data last run <t< td=""><td></td><td></td><td></td><td></td><td>165 plot(wit</td><td>th_maternal\$H4T05 ~ closeness_factor)</td><td></td><td></td><td></td></t<>					165 plot(wit	th_maternal\$H4T05 ~ closeness_factor)			
done In [*]: f script to actually collect the data fdata = readDict('vanderbilt_units.csv') print('Sine checked:', tanderime.useding'utf-st') as fileobject: print('Sine checked:', date_last_run: run_all_geries() print('Cod date now utc > date_last_run: run_all_geries() print('Cod date now the followject.write(generate_utc_date()) print('Cod date now the sequence_toc, date(last_run: run_all_geries() print('Cod date now the followject.write(generate_utc_date()) print('Cod date now the sequence_toc, date(last_run: run_all_geries() print('Cod date now the last_run: run_all_geries() print('Cod date now the last_run: run_all_geries() print('Cod date now the sequence toc, date(last_run: run_all_geries() print('Cod date now the last_run: run_all_geries() print('Cod date now the last run i', with open last last fastoo bdog to the sector date(last') bdog to th					166 t.test(w	with_maternal\$H4T05 ~ closeness_factor, var.equal=TRUE	, conf.level=0.95)		
<pre>In [1]: # Script to actually collect the data #data = readDict('sanderbil: units.csv') #print(json.dumps(data,indent=2)) while furm: #infinite loop print('infinite loop pri</pre>	done				167				
<pre>In (*): f Soript to actually collect the data #data = readbict('vanderhift_units.csv') #fprint(Son_dmps(data,indent=2)) while frue: #infinite loop print('Time checked: , datetime.datetime.utcnow().imoformat()) while open(last_run,tst', 'rt', encoding='utc=0) date_now_utc > date_last_run: run_all_queries() #Uto years a hour before checking again sleep(360) Time checked: 2020-05-05722138114.1655555 Data last run: 2020-05-0542</pre>					168 - #	#C. D.l.tionship between emploise weight and D	IT.		
<pre>In [] [] # data = readDict('youthet' units.csv') # finit(joon.dumps(data,indent=2)) while True: # infinite loop print('line checkin', datatime.datatime.utcnow().isoformat()) with open('last_run.txt', 'rt', encoding='utf=8') as fileObject: data_now_utc = generate_utc_date() print('Dt date_now_utc = generate_utc_date()) # for date_now_utc > date_last_run: run_all_queries() # Update the date last run with open('last_run.txt', 'xt', encoding='utf=8') as fileObject: fileObject.strite(generate_utc_date()) print('done') print('</pre>	In [*], # Sarint to actually collect	the data			169 # Assign	nment #6: Kelationsnip between smoking, weight, and BM	11		
<pre>//dita = readDict('vanderbilt_units.cs') //print(jeen.dateine.utcnov().isoformat()) while True: # infinite loop print('The checked:', dateime.datetime.utcnov().isoformat()) while True: # infinite loop print('The checked:', dateime.datetime.utcnov().isoformat()) while True: # infinite loop print('The checked:', dateime.datetime.utcnov().isoformat()) print('The checked:', dateime.date() print('The date now utc > date_last_run: run_all_queries()</pre>	In [*]: # Script to actually correct	the data			170 * #				
<pre>print('underlinit_units_u</pre>	#data = woodDist(!woodowhilt	unite envis			172 # Relati	ionship between number of days/month smoking and mass	(plot, trendline, summar)		
<pre>#print(jobdumps(data,indent=2)) while True: # infinite loop print()Time checkedi ', datetime.utcnow().isoformat()) with open()last_run.txt, ', 'tt', encoding='utf=8') as fileObject: date_last_run. date_last_run.txt, ', 'tt', encoding='utf=8') as fileObject: in a Jupyter notebook if date_now_utc > date_last_run: run_all_queries() # Dydate the date last run with open('last_run.txt, ', 'tt', encoding='utf=8') as fileObject: fileObject.write(generate_utc_date()) print('lone ') print('lone ') print('done') print(</pre>	#data = readDict(vanderbilt				173 plot(wit	th_maternal\$mass_kg ~ with_maternal\$H4T05)			
<pre>while True: # infinite loop print('Time checked:', dateime.dateime.dateime.utenow().isoformat()) with open('last_run.txt', 'rt', encoding='utf=0') as fileobject: date_last_run = fileobject.read() print('Date_last_run) date_now_utc = generate_utc_date() print('Utc date now_is:', date_now_utc) if date_now_utc > date_last_run: with open('last_run.txt', 'vt', encoding='utf=0') as fileobject: in a Jupyter notebook if date_now_utc / date_last_run: with open('last_run.txt', 'vt', encoding='utf=0') as fileobject: if loobject.write(generate_utc_date()) print('dome') print('</pre>	<pre>#print(json.dumps(data,inden</pre>	t=2))			174 model <-	 lm(with_maternal\$mass_kg ~ with_maternal\$H4T05) 			
<pre>while True: # Infinite 100p print('line checked: , datetime.datetime.utcnow().isoformat()) with open('last_run.txt', 'rt', encoding='utf-8') as fileObject: date_last_run) date_now_utc = generate_utc_date() print('UTC date_now_utc) if date_now_utc > date_last_run:</pre>					175 abline(m	model)	Decript	Files Plots Pa	ickages Help Viewer
<pre>print('Time checked: , datetime.da</pre>	while True: # infinite loop				176 Summary((model)	K SCHPL	🗿 Install 🛛 🕜 Up	odate
<pre>with open ('last_run = fileObject.run = fileObject: date_last_run = fileObject.run = fileObject.run = fileObject: print('Dt date_last_run) date_now_utc = generate_utc_date() print('UTC date now_utc > date_last_run: run_all_queries()</pre>	print('Time checked:', d	atetime.datetime.utcnow().isoformat())			178 # Relati	ionship between number of days/month smokina and BMI		Name	Description
date_last_run = fileObject.read() print('Data 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('done') <td>with open('last_run.txt'</td> <td>, 'rt', encoding='utf-8') as fileObject:</td> <td></td> <td></td> <td>179 plot(wit</td> <td>th_maternal\$bmi ~ with_maternal\$H4T05)</td> <td>in RStudio</td> <td>User Library</td> <td></td>	with open('last_run.txt'	, 'rt', encoding='utf-8') as fileObject:			179 plot(wit	th_maternal\$bmi ~ with_maternal\$H4T05)	in RStudio	User Library	
<pre>print('Date last run:', date_last_run) date_now_utc = generate_utc_date() print('UTC date now isr: date_now_utc) if date_now_utc > date_last_run: run_all_queries()</pre>	date_last_run = file	Object.read()			180 model <-	- lm(with_maternal\$bmi ~ with_maternal\$H4TO5)	minstudio	abind	Combine Multidimensional
date_now_utc = generate_utc_date() print('UTC date now is:', date_now_utc) Python script in a Jupyter notebook if date_now_utc > date_last_run: run_all_queries() in a Jupyter notebook # Update the date last_run with open('last_run.txt, 'wt', encoding='utf-8') as fileObject: fileObject.write(generate_utc_date()) Print('done') print('done') print('done') print() print('done') Rscript # wait an hour before checking again sleep(3600) Notice checking again sleep(3600) Notice checking again sleep(3600)	print('Date last run:', o	date_last_run)			181 abline(m	model)		acenack	ACE and AVAS for Selecting
<pre>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(</pre>		Python script			182 summary((model)		ucepuck	Regression Transformation
<pre>print('UTC date now is:', date_now_utc) if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: in a Jupyter notebook if date_now_utc > date_last_run: if date_now_utc = date_now_utc = date_last_run: if date_last_run: if date_last_run: if date_last_run: if date_now_utc = date_last_run: if date_last_run:</pre>	<pre>date_now_utc = generate_u</pre>	utc_date()			184 # gaplot	t viz: does the effect of monthly smoking on RMT have	a different effect on males than females?	askpass	Safe Password Entry for R,
<pre>In a Jupyter noteBook if date_now_utc > date_last_run: if update the date last run with open('last_run.txt', 'wt', encoding='utf-8') as fileObject: fileObject.write(generate_utc_date()) print('done') print('done') print('done') print('done') fileObject.write(generate_utc_date()) book dots the date last run with an hour before checking again sleep(3600) Time checked: 2020-05-05T23:38:14.765565 Date last_run: 2020-05-04 WUN8 = col_character(), WUN8 = col_character(). WUN8 = col_character().</pre>	<pre>print('UTC date now is:'</pre>	, date_now_utc)	haale		185 ggplot(d	<pre>data = with_maternal) +</pre>	a arrendre errece on mates chain remates.	assertthat	Easy Pre and Post Assertior
<pre>iff date _ now_utc > date last_run:</pre>		in a Jupyter note	DOOK		186 geom_p	<pre>point(mapping = aes(x = H4T05, y = bmi, color = BI0_SE</pre>	X), position = position_dodge(width = .3))	backports	Reimplementations of Fund
<pre>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('done') print() # wait an hour before checking again sleep(3600) Time checked: 2020-05-05T23:38:14.765565 Date last run: 2020-05-04 HAM8 = col_character(), HAM8 = col_charac</pre>	<pre>if date_now_utc > date_la</pre>	ast_run:			187 geom_s	<pre>smooth(mapping = aes(x = H4T05, y = bmi, color = BI0_5</pre>	SEX)) +	hase64epc	Tools for base64 encoding
<pre># Update the date last run with open('last_run.txt', 'wt', encoding='utf-8') as fileObject: fileObject.write(generate_utc_date()) print('done') print() fileObject.write(generate_utc_date()) % 38 M8 print() fileObject.write(generate_utc_date()) % 38 M8 physical (and open (and open</pre>	<pre>run_all_queries()</pre>				188 labs(y	y="body mass index (BMI)", x = "days smoked per month'	', color = "sex")	BH	Boost C++ Header Files
<pre># Update the date last run with open('last_run.txt', 'wt', encoding='utf-8') as fileObject: fileObject.write(generate_utc_date()) print('done') print() fileObject.write(generate_utc_date()) # wait an hour before checking again sleep(3600) Time checked: 2020-05-05T23:38:14.765565 Date last run: 2020-05-04</pre>					189			0	
<pre>with open('last_run.txt', 'wt', encoding='utf-8') as fileObject: fileObject.write(generate_utc_date()) print('done') print() print() fileObject.write(generate_utc_date()) print() print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) print() fileObject.write(generate_utc_date()) fileObject.write(gener</pre>	# Update the date la	st run			49:1 🗰 (Untitle	ed) 🗧	R Script 🗘	🗌 bit	A Class for Vectors of 1-Bi
fileObject.write(generate_utc_date()) bitops Bitwise Operations print('done') bitops bitops Bitwise Operations print() bitops	with open('last_run.	<pre>txt', 'wt', encoding='utf-8') as fileObject:</pre>			Console Termin	nal ×		bit64	A S3 Class for Vectors of 6
print('done') blob A Simple S3 Class for Reprivations (origin c	fileObject.write	(generate_utc_date())			~/nls_data/class/	1 🗟		bitops	Bitwise Operations
<pre>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</pre>					1		1 TAA	L blob	A Simple S3 Class for Repr
print() > physical <- read_tsy("21000-0022-00td.tsv")	<pre>print('done')</pre>				% 38 MB			boot	Bootstrap Functions (Origin
# wait an hour before checking again sleep(3600) broom Convert Statistical Analysis Tidy Tibbles	<pre>print()</pre>				> physical <- r	edd_tsv("21600-0022-Data.tsv")		0	Canty for S)
# wait an hour before checking again sleep(3600) default = col_double(), VERSING = col_character(), BREAK Q = col_character(), BREAK Q = col_character(), H4LM8 = col_charac					cols(um spectricación.		broom	Convert Statistical Analysis
sleep(3600) VERSION4 = col_character(), sets Sets Time checked: 2020-05-05T23:38:14.765565 Caloba Caloba Date last run: 2020-05-04 Caloba Caloba	# wait an hour before ch	ecking again			.default = co	ul_double(),		CarData	Companion to Applied Per
BREAK_Q = col_character(), H4LM8 = col_character(), Date last run: 2020-05-04	sleep(3600)				VERSION4 = co	l_character(),		CarData	Sets
Time checked: 2020-05-05T23:38:14.765565 Base64, ROC AUC, etc. Date last run: 2020-05-04 cellranger					$BREAK_Q = col$	_character(),		caTools	Tools: moving window stat
Date last run: 2020-05-04	Time checked: 2020-05-05T23:	38:14.765565			$H4LM8 = col_c$	<pre>character(), character()</pre>			Base64, ROC AUC, etc.
	Date last run: 2020-05-04)			Cenranger	Rows and Columns

Command-line interfaces (CLI)



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**



• 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)

			linked-data		P maste	r Branch	✓ C Fetch origin Last fetched 28 minutes ago
			An updated version of GitHub I	Desktop is available and will be i	nstalled at the n	ext launch	n. See what's new or restart GitHub Desktop.
con Command Prompt	—	\times	Changes 7	History	vanderbot	/departme	ent-configuration.json
		~	7 change	ed files			@@ -1,22 +1,22 @@
C:\Users\steve-bootcamp\github\linked-data>git status			publications/data/collect_data.ip	pynb	1	1	<pre>{ "deptShortName": "medicine", ""</pre>
On branch master Your branch is behind 'origin/master' by 35 commits, and can be fast-	forwarded.		vanderbot/department-configur	ration.json s-to-write.csv -	-	2	<pre>- "medicine": { + "deptShortName": "divschool", + "divschool", </pre>
(use "git pull" to update your local branch)			vanderbot/divschool-employees	s-with-wikidata.csv	4 5	4	"scrapeType": 0, "categories": [
nothing to commit working tree clean			vanderbot/divschool-employees	s.csv	6	6	
nothing to commit, working thee clean			vanderbot/vanderbilt_wikidata.csv	:sv [·	8	8	<pre>J, "baseUrl": "https://wag.app.vanderbilt.edu//PublicPage/Facult y/PickLetter?letter=",</pre>
C:\Users\steve-bootcamp\github\linked-data>git branchlist gh-pages help * master C:\Users\steve-bootcamp\github\linked-data>			567		9 10 11 12 13 14 15 16 17	9 10 11 12 13 14 15 16 17	<pre>"nTables": 1, "departmentSearchString": "Medicine", "departmentOld": "089953931", "testAuthorAffiliation": "Medicine Vanderbilt", "departmentSearchString": "theology", "departmentOld": "07914452", "testAuthorAffiliation": "theology", "labels": { "source": "column", "value": "name" }, "descriptions": { "descriptions": { "source": { "source":</pre>
			Summary (required) Description		18 19	18	- "value": "physician"
		\checkmark	<u>2</u> +		20 21 22	20 21 22	+ "value": "theologian" } }⊘≓
			Commit to	master			

command-line interface for Git

graphical user interface for Git

Variables and objects





- 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']
```



classes and objects

Classes are abstract categories of things.

Objects are particular instances or individuals of a class.



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.

	\frown	\mathcal{A}
0000	item[0] item[1] item[2] item[3] Class: list	

'apple'
'orange'
'banana'
'lemon'
'lime'

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

3593
269
45801
2804

True
False
False

object name: ids type: list object name: in_stock
type: list

Executing code

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() a statement # 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)

Python code

R 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

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).

Access to digital collections 24/7

Remote Support for Teaching and Research Needs

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

NEED HELP? ASK A LIBRARIAN!

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