Southern New Hampshire University | School of Engineering, Technology, and Aeronautics

Syllabus
Instructor: brian hogan
https://github.com/bbe2/instructor.brian/tree/it.304.fall.2023

Course Description – it.304 - Systems requirements and implementation planning

Location: on-ground, SETA, 209, Wednesday and Friday at 11:00 – 12:15

Systems analysis and design is an art form, discipline, and science. The 1890s witnessed its formative pillars of speed, quality, and checklists thanks to the efforts of Frederick Taylor, and his methods and stop-watches remain key systems analysis and design tools[1].

```
1890    1930    1950       1960       1970       1980    1990    2000       2020
Scientific mgmt
     Fordism
           Manufacturing automation (MA)
                Statistical process control
                Total quality mgmt (TQM)(Demming)
                      Transistors
                           Microprocessors
                                 Integrated data stores (IDS)<tape>
                                      Personal computers
                                            Information Systems MIS\MES
                                                 Business process reengineering
                                                       Info. factory-servers
                                                            Intelligence systems
                                                                 Data warehouses
                                                                 Artificial Intell.
```

To perform systems analysis and design well, it helps to understand different process models alongside what operation managers need to see, what business leaders want to achieve, and what financiers advise on sustainability. Information technology (IT) facilitates systems design efforts through codification. Leaping forward, Artificial intelligence (AI) identifies potentiality by pairing unseen connections with deep learning neural networks.

In the 1990s, MIT computer science professor Michael Hammer developed the management theory of business process re-engineering (**BPS**). Its tenets are process improvement, process re-design, and process re-engineering. **BPS** emphasizes the application of a holistic view of understanding how business objectives and processes are or are not aligned.

Question: have you stood in line in a coffee shop while the servers are busy doing lots of things but not helping you? IT online ordering has changed business operations, and perhaps customer experience is out of alignment with new transaction processing models. Good design principals may have identified this experience gap a priori by using first in, first out (FIFO) queueing.

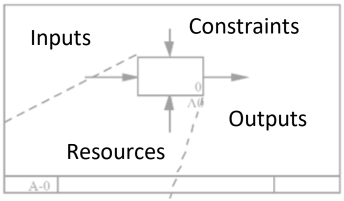
In 2022 **BPS** is alive and well, as witnessed by consultancies like IBM's Business Process Reengineering <IBM-BPRS> and Bain & Company's Business Process Redesign <Bain>. **BPS** names change, such as Accenture's Human + machine intelligence, but its Tayloristic principles are still profitable.

Business requirements, business rules, system specifications, environmental factors, opportunities to tear things apart, reorganize, recodify, and discover new viability vectors. **IT** is essential to this process. Understanding the application of BPS models will help you become a better system's designer through the development of abstraction and looking ahead skills. These skills improve with training and application.

In BPS, an individual's skills express themselves in the selection, testing, and application of BPS models to frame situations. Abstracting systems involves applying process engineering skills to help orchestrate quality engineered improvements, new IT paradigms, and machinery to augment and facilitate change. Measuring change is problematic, and this course is not focused on it. Suffice it to say, sometimes only profit and stock price reflect the systemic effects of an organization's BPS's efforts.

Why do BPS efforts wane? One answer is people and systems "move on." Life flows forward with designers and business champions refocusing and pulling the wind out of BPS sails. Perhaps work was not understood by managers leading to other ineffective, haphazard d outcomes. Developing skills in this arena will help you identify concerns hopefully before a ghoulish nightmare.

The coursework is challenging, accessible, and extremely useful. As such, the expectation is your work will progress naturally in an ongoing fashion driven by self-interest and self-motivation.



Evidence of skills in systems analysis & design skills includes

1. Written examination and diagnostics of systems thinking.
2. Use of 10-15 systems knowledge classificaitons.
3. Use of Python for data modeling and codifify transactional information.
4. Translation of systems analysis and design principles into functional specs., object models, requirement specs., and implementation plans.

Attendance

Your presence in class is instrumental to your learning, peer learning, and skills advancement. **500 of the 1,000 total class points** represent in-class attendance, activities, and participation. Evaluation points accumulate per class for attentiveness, engagement, distraction, level of focus, use of cell phones, and illustration of sustained participation. Abnormal breaks or unexcused time out of class deducts (2.5*total.minutes out of class). For example, and abnormal break of 10 minutes will cost you 10*2.5 = 25/95 = 0.2631*35.71<per class points> = or 9.5 points.

7.pillars.of.python – Required training

In order to have full competence in systems design it's necessary to perform

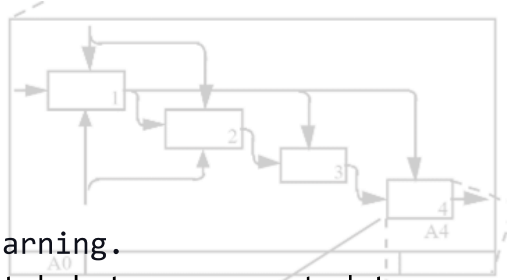
- data transformations and manipulation with Python data objects.

  A customized programming crash course will build your competence with conditionals, functions, iterators, conditionals, and classes to provide you with basic fierce compute skills.

Required Resources

Textbook Notes	Textbook	
You are welcome to use either the 11th or 12th edition. Other details provided in class. • excellent book to have on hand today and years from now.	Tilley, Scott (2022). **Systems analysis and design, 12th Edition**. Shelley Cashman Series. Cengage. Published 2022. ISBN 978-0-357-11781-1.	<amazon> 12th <abebooks> 11th <abebooks> ~$10
Not required to purchase but an • excellent book to have on hand today and years from now.	BKrogerus, M., Tschappeler, R., and Pienning, J. (2018). The decision book: fifty models for strategic thinking. ISBN-10: 0393652378, ISBM-13, 978-0393652376.	<abebooks> ~$7

Tools, technology, and software to facilitate evidence



1. **Index cards and pencils #=> provided.**
2. Installation of **Google Chrome for Colab**
3. Microsoft Visio \ university.download
4. Anaconda – IDE Spyder. Best for data transformation learning.
5. Students are not required to figure out code from scratch but are expected to type code or re-stenograph as necessary.

Required Performance Hardware – Mouse!

1. **Mouse!** – if you don't have one I will find one for you.
    - Use of 3-finger dolls for compute activties is slow and illustrates to an employer ineffectiveness. Points substracted if no mouse in-class.
2. **2nd Monitor** – 2 Monitors help you get your work done are virtually required for effective computer science work. If need one ask and I will find for you.

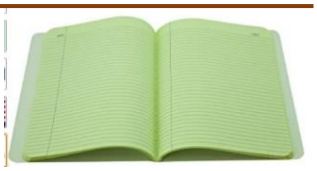
Instructor availability

- Simply reach out to your instructor on Teams and we will schedule and continue your support and progress. b.hogan@snhu.edu

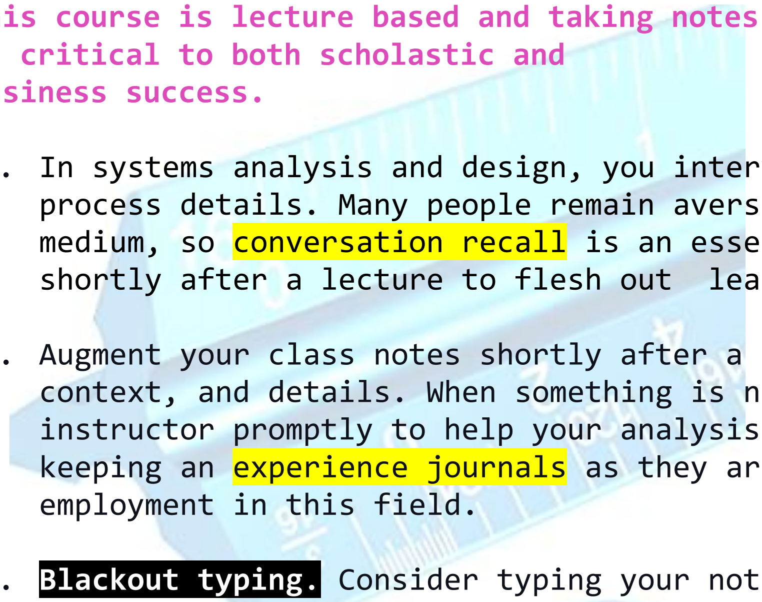
Prior course competencies

- IT-20358: Make ethically informed decisions based on awareness of legal and organization parameters
- IT-20359: Develop a systems requirements specification
- IT-30360: Develop an implementation plan

A brief guide to effective analysis



This course is lecture based and taking notes is critical to both scholastic and business success.



i. In systems analysis and design, you interview customers to learn information and process details. Many people remain averse to recording conversations in any medium, so conversation recall is an essential skill. Augment your class notes shortly after a lecture to flesh out learnings, context, and details.

ii. Augment your class notes shortly after a lecture to flesh out learnings context, and details. When something is not well-understood reach out to your instructor promptly to help your analysis skills advance organically. Consider keeping an experience journals as they are helpful to reflect upon if you seek employment in this field.

iii. **Blackout typing.** Consider typing your notes and ideas with the computer screen blocked or blacked out. Doing so stimulates your abstraction engine flow.

iv. Word spelling/grammar matters. But, for now, focus on **IDEA** generation and design. The Victorians have 1000s of well-written texts nobody reads, and Herman Melville, a Victorian, wrote about a **process --** whale hunting.

v. Ask questions – right away. Your focus should always be
   a. An engineered ***course of action***, and
   b. Clarified thinking.

Good writing is good thinking

Effective analysis items to do first:

1. Write down any ideas about assignment and models that come into you they arrive. Carry index cards, text yourself, keep a moleskin notebook and pencil. Don't put off recording something interesting for even for five minutes else "whooosh-vapor."
   - laboratory bench scientists are required to this day to perform daily journal of their work. It is a skill worth considering.

2. Carefully read every word of the assignment 2x to make sure you consider what lectures, readings, and models you're asked to consider. Cogitate carefully as every assignment is curated to deepen knowledge and focus thinking.
   - Consider reviewing weekly assignment section and re-reading curated course content when your logic is amiss.
   - Between 4-8 will review strategies for librarian type research.

3. If you assignment wants you to use class lectures, then study your lecture notes. Hopefully you have augmented your lecture notes shortly after the lectures to add context.
   - Add to your class notes in another color pen to increase neuroplasticity.

Research Websites

The internet is full of information *and* advertisements. Use your time wisely working with research sites instead of the internet. I can help anyone setup an account.

- Shapiro Library - Research Guides at Southern New Hampshire University (snhu.edu)
- Home Feed | ResearchGate, https://www.researchgate.net/
- Routledge - Publisher of Professional & Academic Books, https://www.routledge.com/

Wikipedia “IS NOT” an academic reference website

Use of Wikipedia is for idea generation and quick information fact sharing.

- Any links to dictionary wikipedia is to help quickly build topic background and.or augment class lectures.
- Wikipedia helps to broadly engage a topic’s context and related info.
- Wikipedia **is not** an academic reference nor a substitute for quality academic media. Some academics argue Wikipedia’s veracity p.e.r.i.o.d.
- **At any time a student may request academic approved learning media to substantiate any reviewed topic.**

Task, activity, assignment, and report Due Date

Anytime the day due day is stil day on world clock.

Missing Class

To recoup class points, present the instructor with a doctor’s note, a campus health department note, or one from your academic advisor or dean.

Grading

All activities are graded via points earned per week per activity.

Category	# weeks or items	Points	Total points
In class participation & JAMs	14	35.71	500
Labs	10	20	200
Quizzes	10	10	100
Project	1	200	200
		Total	1000

University grading system

Grade	Numerical Equivalent	Points
A	93-100	4
A-	90-92	3.67
B+	87-89	3.33
B	83-86	3
B-	80-82	2.67

C+	77-79	2.33
C	73-76	2
C-	a	1.67
D+	67-69	1.33
D	60-66	1
F	0-59	0
I	Incomplete	
IF	Incomplete/Failure	
IP	In progress	
W	Withdrawn	

Diversity, Equity, and Inclusion

As indicated in SNHU’s core value, the university is committed to “embrace diversity where we encourage and respect diverse identities, ideas, and perspectives by honoring difference, amplifying belonging, engaging civilly, and breaking down barriers to bring our mission to life.”

In higher education, you’re expected to think critically while exhibiting a growth mindset. This mindset includes the practice of diversity, equity, and inclusion (DEI) to provide transformative experiences for yourself, peers, faculty, and staff.

Collectively we are an organize learning mechanism. Through our community, compassion, and collaborative interactions we walk with respect towards a greater

SNHU Handbook and University General Guidelines

- https://snhu.sharepoint.com/sites/CAMPUSACADEMICS
- Use your internal resources to access the student handbook detailing all features of attendance, academic honesty et. cetera.
- Perform authentic work.
    - SNHU requires all students adhere to high standards of integrity including avoidance of plagiarism and cheating.
- SNHU adheres to copyright provisions of the Copyright Act.
- Consult the handbook when considering withdrawal or need anything else.

ADA/504 Compliance Statement

SNHU is dedicated to providing equal access to individuals with disabilities in accordance with Section 504 of the Rehabilitation Act of 1973 and with Title III of the Americans with Disabilities Act (ADA) of 1990, as amended by the American’s with Disabilities Act Amendments Act (ADAAA) of 2008.

SNHU prohibits unlawful discrimination on the basis of disability and takes action to prevent such discrimination by providing reasonable accommodations to eligible individuals with disabilities. The university has adopted this policy to provide prompt and equitable resolution of complaints regarding any action prohibited by Section 504, the ADA, and the ADAAA.

For any questions about support services, documentation guidelines, general disability issues, or pregnancy accommodations please email wellness@snhu.edu. See my.snhu.edu and select the wellness tab. And the campus accessibility center at cac@snhu.edu.

For anything regarding discrimination please contact school professionals right away at the emails above and.or see the Disability and Accessibility Services at https:\\my.snhu.edu

Student Support Resources including Tutoring and Instructional Support

It is really amazing to have a careteam@snhu.edu to help students with assistance of all sorts. Again, this is an amazing resource.

- Consider this service if feeling pressured or overwhelmed.
- For instructional support email instructionalsupport@snhu.edu.
- For in class tech support call 603.645.9615

Other Key Resources

- https://snhu.sharepoint.com/sites/thesource
- https://snhu.sharepoint.com/sites/CAMPUSACADEMICS

Lecture and Weekly Assignment Schedule

In this document, every week lecture notes are added and/or assignment tasks in the tables below.

Template format

Wk	Focus & Medium	Weekly Topic & Assignment
x	**~py pkg index~** **https://pypi.org/**	Hands - mediapipe (google.github.io) **note:** Weekly Assignments (1 or 2 pages per week as indicated on left)

Wk	Focus & Medium	Weekly Topic & Assignment
x		

Wk	Focus & Medium	Weekly Topic & Assignment
x		

Wk	Focus & Medium	Weekly Topic & Assignment
x		

```
7.pillars.ofpython

#=> fill in the blanks
     obj_Name    | charcter code      |   explicit code
      --------- | ------------------ | -----------------------------
  i) mytuple =  |  (, )              |=> mytuple = tuple(myobject)
 ii) mylist =   |                    |=> mylist  =
iii) mydict =   |                    |=> mydict  =
 iv) myset  =   |                    |=> myset   =
  v) dataframe =|                    |=> df      =
 vi) mystring = |                    |=> mystring=
```

```
#==================================
                  [({...})],{“” : [ ] }
#==================================
                string data is in a dictionary {key:value}
                   which is inside a tuple
                    which is inside a list
                     separated by a comma to another object
                      which is a dictionary with
                       a string for a key, and
                         a list of its key values
```

Now you have the tools to decipher how data is packed and figure out how to mix and mingle python objects and re-organize as needed.

You have also worked with iterators, conditionals, and variables and can transpose data

```
=============================================
dir(<myObject>) => displays its constructors, methods, and attributes
['__class__', '__delattr__', '__dict__','__dir__', '__doc__',
 '__eq__', '__format__', '__ge__', '__getattribute__', '__gt__',
 '__hash__', '__init__', '__init_subclass__', '__le__', '__lt__',
 '__module__', '__ne__', '__new__', '__reduce__', '__reduce_ex__',
 '__repr__','__setattr__', '__sizeof__','__str__', '__subclasshook__',
 '__weakref__',
 'name',
 'species',====> user defined attributes
 'train']

help(<myobject> or <function)
-------
 |  Data and other attributes defined here:
 |  name = ''
 |  species = '' ==> these are the attributes in our wk7 object
 |  train = ''
```

<table>
<tr><th>Wk</th><th>Focus &amp; Medium</th><th>Weekly Topic & Assignment</th></tr>
<tr><td></td><td>

Python pillars

- core objects
- conditionals
- iterators
- functions
- transposition

PLEASE BE CREATIVE

</td><td>

```
  """# -*- coding: utf-8 -*- Created Sep 15 07:58:23 2022
@author:17574 b.hogan@snhu.edu it.304.fall.22
Objective: import data and apply zipper to transform, iterate,
use conditionals, apply functions, leading to python classes work
Library homebase = Python package index:  https://pypi.org
"""
'''=====================
#==========================
#============================================================
#=>STEP 1 get pip library install path from
#============================================================
#==========================
#====================='''
import pandas as pd              #dataframe library
import numpy as np               #numeric library
import matplotlib.pyplot as plt  #visualization library
import os
os.getcwd()            #where am i? <get working directory>
#os.chdir('c:\\Users\BBE\DATA\') #some op.sys use one slash
os.chdir('c:\\Users\\17574\\Desktop\\data') #microsoft uses 2 \\
os.getcwd()

df0 = pd.DataFrame() #explicitly set the data object
#df0 = pd.read_csv("shakes_corpus_v1.csv")   #ETL method 1
df0 = pd.read_excel("shakes_corpus_v1.xlsx") #ETL method 2
df0.info()
   #             RangeIndex: 37 entries, 0 to 36
   #             Data columns (total 3 columns):
   #              #   Column  Non-Null Count  Dtype
   #             ---  ------  --------------  -----
   #              0   title   37 non-null     object
   #              1   script  37 non-null     object
   #              2   type    37 non-null     object
   #              3   ID      37 non-null     int64
   #             dtypes: int64(1), object(3) memory usage: 1.3+ KB

print(type(df0))  #use type() to always see what an object is
df0.head()
    #                                   title  ...    type
    #             0  Alls Well That Ends Well  ...  Comedy
    #             1            As You Like It  ...  Comedy

#2.1 use pandas df.to_dict() to move data into dictionary object
mydict = df0.to_dict()
print(mydict.keys())    #['title', 'script', 'type', 'ID'])
type(mydict.keys())     # object itself is keys

#2.2 understand what a dictionary and zip is doing
mylist_keys = list(zip(mydict.keys()))
mylist_keys    # [('title',), ('script',), ('type',), ('ID',)]

#Inspect huge data and then break into smaller chunks
mylist_values = list(zip(mydict.values())) #WOW huge !
#point - zip helpful but continue to learn more functions

mylist_values   #=======================> #MEGASAURUS
                     #  35: 'Tragedy',
                     # 36: 'Tragedy'},),
                     #{0: 1,
                     # 1: 2,
                     # 2: 3,
```

</td></tr>
</table>

```
'''=====================
#==========================
#============================================================
#=>STEP 2 - seperate Megasaurus into usuable object chunks
#============================================================
#==========================
#====================='''

'''2.1'''
type(mylist_values) #=> [({...})],

'''======> packed as [({...})], =>list, tuple, dictionary'''

type(type(mylist_values[1]) )#hmm doesn't unpack
len(mylist_values)  #=> 4 columns in spreadsheet, ie data objects

'''megasaurus - all plays and words'''
mylist_values
                        # => format is list[(tuple(dict))]
                        # [  ({id:title}),({id:script}),
                        #    ({id:type}), ({id:id}) ]
                        # zip added an key sequential value
'''==>2.2'''
'''use slicing [0:1], [2] to view next level down'''
type(mylist_values[0])  # tuple
mylist_values[0]        #=> [x] is called slicing

                  Out[23]:
                  ({ 0: 'Alls Well That Ends Well',
                     1: 'As You Like It',

'''now think data like in spreadsheet'''
#  columns
#    0      1     2      3
# |title |script| type |  id  |
#  hamlet,oh joy,tragedy, 29

mylist_values[1] #displays all the script text!

'''==>2.3'''
len(mylist_values[1])  # waits its '1' so need to unpack my data

mylist = []
for i in mydict['title'].values():
    mylist.append(i)
mylist
len(mylist)  #37 - does htat match spreadsheet? always know your bounds

title_total_characters = 0    #how many characters?
for i in mylist:
    title_total_characters = title_total_characters + len(i)
title_total_characters  #do you get 560 ?

'''============================================================
==>2.4 autoBOTs304 - repeat this for total script words
#============================================================
===> moved  this into the graded_assign_wk7'''

#=====================
#==========================
#============================================================
#=>STOP! : view 'Variable Explorer' window
#  use this feature to propel data transformation learning
#============================================================
#==========================
#=====================

'''#=====================
#=> WRAP - UP  Housekeeping
#  delete variables not using; help avoid unnecssary mistakes
#==========
# be mindful how you stage both variable and data names
#   df0 = baseline import
#       df1 = analysis 1
#           df2 = analysis 2
#==================='''

'''==>2.5'''
del mylist_keys    # del removes a variable

'''
mylist2 = []
for i in [mydict.get('title')]:
    mylist.append(i)   #so what happended here a. wrote name list wrong
print(len(mylist2), len(mylist))
#make a note here on what happended...............
mylist    #stacked a list on a dictionary bc meant to use list2
'''
#go back and rest data for part 2
mylist = []
for i in mydict['title'].values():
    mylist.append(i)

'''=====================
#==========================
#============================================================
#=>STEP 3:  Use dir(object) to learn its methods to get work done
#============================================================
#==========================
#====================='''
'''==>3.1'''
#===============> use dir() to get functions available for an object
myset = set()
print(type(myset))
dir(myset)
 # '__xor__',  ==> these are constructors, more later
 #    'add', 'clear',    ==> these are methods
 #   'copy','difference', 'difference_update', 'discard',
 #  'intersection','intersection_update', 'isdisjoint', 'issubset',
 #   'issuperset','pop', 'remove','symmetric_difference',
 #   'symmetric_difference_update','union',  'update']'''

'''==>3.1'''# ====> SETS
mylist2 = mylist
mylist2.append("Winters Tale")  #add one duplicate title
myset = set(mylist2)
print(len(mylist),len(myset))  #so got rid of duplicate
del mylist2

#============> ACTION learn what you need and go find it
mystring = ""
print(type(mystring))
dir(mystring)
#'''__subclasshook__',  'capitalize', 'casefold',, 'center',
#'count', 'encode', 'endswith', 'expandtabs', 'find', 'format',
#'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal',
#'isdigit', isidentifier', 'islower', 'isnumeric', 'isprintable',
#'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip',
#'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
#'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines','startswith',
#'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']'''
```

```
'''====================
#==========================
#============================================================
#=>STEP 4: More dictionary: .keys(), .values(), .get(<key>)
#============================================================
#==========================
#====================='''

'''==>4.1'''
mydict.get('title')  #.get() views one series
play_names = [mydict.get('title')]
play_names
        [{0: 'Alls Well That Ends Well',
          1: 'As You Like It',
          2: 'The Comedy of Errors',
mylist
# Now add titles to a different object with an iterator
mylist2 = []
for i in [mydict.get('title')]: #method returns a dict obj
    mylist2.append(i)
mylist2
    [{0: 'Alls Well That Ends Well',
      1: 'As You Like It',
      2: 'The Comedy of Errors',

#3.2 =>  Learn dictionary key, value, items parameters
mylist_key =  []
mylist_values = []
for k,v in mydict.items():
    mylist_key.append(k)
    mylist_values.append(v)
mylist_key              #['title', 'script', 'type', 'ID']
mylist_values   #'''again megasaurus'''

'''==>4.2''' #=>  Understand and count items in a list
len(mylist_values) #hmm  why is this only four ?
mylist_values[0]
mylist_values[1]
mylist_values[2]
mylist_values[3]

#=====================
#==========================
#============================================================
#=>STEP 3:  Use Functions and get Meta Data
#============================================================
#==========================
#=====================
https://docs.python.org/3/library/functions.html#built-in-functions

sum(mylist_values[3])-1
sum(df0['ID'])-1
len(set(df0['ID']))
```

wk.16 Machine Learning 12.12-12.16

Program / operating parameters:

1. Demonstrate how to create a small Python program, called a script, and generate speech to text and text to audio results.
2. Challenge a user to replicate proper syntax, indenting, and other coding idioms to ensure programs run as intended.
3. Educate on basic data encoding where binary (1 or 0) is used for pictures/voice and nonbinary (byte/collations) is for text.
4. Educate on how libraries simplify program feature engineering making the art of the possible a far less daunting task.

Scenario 1: Generate a working program in a Python integrated development environment (IDE) such as Anaconda. The following example uses the Jupyter notebook program as part of the Anaconda Install.

Scenario 2: Expand code requiring 2 audio requests but deliver a single audio outcome file

Hint: The trick of this scenario is to create 2 separate myWords variables.

- In Python variables are either implicitly or explicitly declared.
- Code line 7 “my Words” is an implicit declaration as its type is not declared, such a character (char) or number
- Add a “_1” to the variable and then duplicate code lines 5-8 with a second variable myWords_2
- Finally, combine the myWords_1 with myWords_2 into myWords to deliver the audio output

```
""" Part 1: Set Computer File Directory os=operating system"""
import os
os.chdir('C:\\Users\\17574\\Desktop')

""" Part 2: Set Google Speech Recognition and Microphone Library Functions
import speech_recognition as sr
import pyaudio

""" Part 3: Ask user to same something use Google speech to parse words"""
with sr.Microphone() as source:
    print("Ready? Say something quick")
    myWords = sr.Recognizer().listen(source)
    print("You Said...: "+ sr.Recognizer().recognize_google(myWords))

>>> Ready? Say something quick
>>> You Said...: Nacho

"""Part 4: Encode words into audio file audio data is binary so add 'wb'
                            for 'write binary data (1 or 0)"""
with open("myAudio.wav", "wb") as file_:
    file_.write(myWords.get_wav_data())

"""Part 5: Import a generic microphone module """
from playsound import playsound
playsound('myAudio.wav')

import os
os.chdir('C:\\Users\\17574\\Desktop')
import speech_recognition as sr
import pyaudio
with sr.Microphone() as source:
    print("Ready? Say something quick")
    myWords_1 = sr.Recognizer().listen(source)
    print("You Said...: "+ sr.Recognizer().recognize_google(myWords))
with sr.Microphone() as source:
    print("Ready? Say something quick")
    myWords_2 = sr.Recognizer().listen(source)
    print("You Said...: "+ sr.Recognizer().recognize_google(myWords))
myWords = myWords_1 + myWords_2
with open("myAudio.wav", "wb") as file_:
    file_.write(myWords.get_wav_data())
from playsound import playsound
playsound('myAudio.wav')

>>> Ready? Say something quick
>>> You Said...: Nacho
>>> Ready? Say something quick
>>> You Said...: Nacho

""" Run like a Pro """
import os
os.chdir('C:\\Users\\17574\\Desktop')
import speech_recognition as sr
import pyaudio
with sr.Microphone() as source:
    print("Ready? Say something quick")
    myWords = sr.Recognizer().listen(source)
    print("You Said...: "+ sr.Recognizer().recognize_google(myWords))
with open("myAudio.wav", "wb") as file_:
    file_.write(myWords.get_wav_data())
from playsound import playsound
playsound('myAudio.wav')

>>> Ready? Say something quick
>>> You Said...: I like cake
```

Wk	Focus & Medium	Weekly Topic & Assignment
8 Oct 17 to 22	(2 of 5) wk8 code git	""" it.304.wk8 (10/16-10/22/22) Created on Sat Oct 15 13:56:24 2022 @author: 17574 b.hogan@snhu.edu """ #======================= > Week 8 #===== Classes - Week 8 #================ #======================== #=> Objective: use the following Classes example to make one of your own #=> new function input("<message>") -> asks user for a value #=== # Part I: Import libraries and source data

8

Oct 17 to 22

(3 of 5)

wk8 code git

```
# Part II:  Draft an object with couple functions
# Part III: Creat a child object and run the function
# Part IV:  Run a report
#=========================================================================
  ''' CARLY! this is not boo scary!
      conditionals (below)are a set of questions, often in your own words.
      if you are stuck, set a timer and spend no more than 20 minutes.
      research says your better phoning or emailing a friend as anything
      after 20 minutes exceeds optimal learning. good luck!

#=> # Part I:
#===============================================================
#=> # Part I:  Import libraries and source data
#===============================================================
Import libraries + data
import pandas as pd                               #dataframe library
import numpy as np                                #numeric library
import matplotlib.pyplot as plt                   #visualization library
import os                                         #operating system library
import sys                                        # sys.exit()
os.chdir('c:\\Users\\17574\\Desktop\\data_it304') #microsoft uses 2\\
df0 = pd.DataFrame()                              #explicitly set datafarme
df0 = pd.read_excel("data_shakes_corpus_v1.xlsx") #ETL method 2
df0.info()# RangeIndex: 37 entries,0 to 36, 4 col=> all data u need to
index
mydict = df0.to_dict()                            #df to dict
'''mydict_shakespeare => {'title':{},'script':{},'type':{},'ID':{} }'''
len(df0) #37                            lenth is always veritical by
default!



#=> # PART - DEtour - was best to add NEW INFO here
'''this will help you create a report for quiz end of wk...'''
#================================>
#=> Function idea and drive a bitchen camero data to excel
#===============================
  # Use if, elif, else 'conditionals' to draft your questions based on data
  # Consider drafting 1-3 questions on an index card before coding
  # detail what information need to perform so you focus vs get stuck on
names
  # remember - objects are the actors and functions are their script

'''Fucntion ideas & examples:
   i) write a function to count total characters in a play or all plays!
   ii) use an iterator, count characters, and put in a list
   iii) use new lists to create a report or write back to excel using'''

mylist = []                 #so this could be a function to count
characters
for i in mydict['title'].values(): mylist.append(len(i))
print(mylist, type(mylist), sum(mylist))

#use the new objects and variables to creat a dictionary
myNewDict = {sum(mylist): mylist}
        #   or {'play-1':[<TitleTotalWords>,<ScriptTotalWords>]}

print(myNewDict)
print(type(myNewDict))
myDF = pd.DataFrame.from_dict(myNewDict)#function create a pandas.DF from
dict
#myDF.info()                               #check it out

''' Send to excel or view here - will review in class'''
mywriter = pd.ExcelWriter('myoutput.xlsx') #create object that writes out
myDF.to_excel(mywriter)
```

8

Oct 17 to 22

(4 of 5)

wk8 code git

```
mywriter.save()
myDF                                #Excel will look exact same !

#=> # Part II:
#============================================================
#=> # Part II:  Draft an object with couple functions
    # We are training with .self notation. write self.<attribute or
variable>
    # are inherent, or part of our instantiated children objects
#============================================================

'''START - HIGHlight all of class and hit F9 from lines 93 to 150 '''

class shakespeare_minion:           #this defines the parent object
    pass
    name = ""
    perform_work = 0                #yes,no switch so could exit terminal
    total_plays_not_read = len(df0) #use an object vs. hardcode a value
    total_plays_read = 0            #increment so you know how much work
done
    num_plays_work_now = 0          # countdown tracker based on user input

    '''Function-1: ask user how many plays to read'''
    def how_much_work_master(self):
                      #int() function here ensures user response encoded as
a #
        perform_work = int(input("Enter greater than 0 to run program =>
"))
        if perform_work <= 0:
            sys.exit()       #On/off switch so can exit program in terminal

        if perform_work > 0:    #NEW - ask user a question with input()
            self.num_plays_work_now = \
                int(input("Enter how many plays you will read today?=> "))
        perform_work = 0    #set back to zero as 1x trigger








    '''Function-2: have minions completed what they said they would do?'''
    def do_work_and_report_status(self):

    #0) for transactions, here would be some kind of wait time to do work


    #1) condition 1 - Did we complete total work yet?
        if self.num_plays_work_now <= 0:
            #after test, then increment/decrement associated variables
            self.total_plays_not_read = self.total_plays_not_read - 1
            self.num_plays_work_now = self.num_plays_work_now - 1
            total_plays_read =+1 #another way to increment variables

            return "Master! {} is done. I finished {} plays today.". \
                            format(self.name,self.total_plays_read)

    #2) condition 2 - Still doing daily work ?
        elif self.num_plays_work_now > 0:
            #after test, then increment/decrement associated variables
            self.total_plays_not_read = self.total_plays_not_read - 1
            self.total_plays_read = self.total_plays_read +1
            self.num_plays_work_now = self.num_plays_work_now -1
            total_plays_read =+1        #another way to increment variables
    #3) condition 3 - this is a NESTED loop b/c now you either no more work
    #       or you report what you have left to do in this batch
            if self.num_plays_work_now == 0:
                return "Master I have {} plays left to read AND no more 
work.\
                           I am 100% done for today so start over!".\
                                       format(self.total_plays_not_read)

            else:
                return "Master I read {} plays today and have {} more plays 
\
                to do in this most egregiousness and unjust batch.".\
                    format(self.total_plays_read,self.num_plays_work_now)


'''END HERE - HIGHlight all of class to define full object'''

#============================================================
# Part III: Creat a child object and run the function
#============================================================

# IIIa: ask user number plays to ready & run the transaction
'''Run these 3 lines together! - This starts to queue up total work'''
minion = shakespeare_minion()
minion.name = "Toothless Harold"
minion.how_much_work_master()         #ask user how much to do!

#=================
'''====>Now run a transaction, that is read a play.
        this program runs these transactions manually.
        The final little program we make will run them all at once.'''

#================= select all 4 lines - keep running to run out of work!
print(minion.do_work_and_report_status())
print(minion.total_plays_not_read)
print(minion.num_plays_work_now)
print(minion.total_plays_read)
```

Wk	Focus & Medium	Weekly Topic & Assignment
	(1 of 4) QUIZ Instructions QUIZ Answer **Objective:** more exercises on python pillars to prepare for creating an object generator. We will review in class but you will need to answer and turn it in when finished. Turn it in by the 19th the	

```
"""# -*- coding: utf-8 -*-
Created on Mon Oct 10 10:59:53 2022
@author: 17574
====================
#==========================
#============================================================
#=> it.304 2nd Graded Assignment
#============================================================
#========================"""
import pandas as pd              #dataframe library
import numpy as np               #numeric library
import matplotlib.pyplot as plt  #visualization library
import os
os.chdir('c:\\Users\\17574\\Desktop\\data') #microsoft uses 2 \\
df0 = pd.DataFrame() #explicitly set the data object
df0 = pd.read_excel("shakes_corpus_v1.xlsx") #ETL method 2
df0.info()
mydict = df0.to_dict()
#===================================
#=>1.0 Pillar: Iterators
'''1.1 Task: use an iterator and produce total words all plays'''
#===================================
```

latest but won’t take you long.

- I will post everyone’s own gradebook this week.

- The 2nd part of the week will review class objects

(2 of 4)

```
#==> ENTER YOUR CODE HERE
mylist = []
for i in mydict['script'].values():
    mylist.append(i)
total_script_characters= 0    #how many characters?
for i in mylist:
    total_script_characters = total_script_characters + len(i)
total_script_characters

# Answer: 1,212,379

'''1.2 Task: what is easiest in code to double total characters'''
#==> ENTER YOUR CODE HERE

total_script_characters*2

# Answer: 2424758

#====================================
#=> 2.0 Pillar: Functions
'''Task: Generate a tuple wth the code provided
   hint: use codebook '''
#====================================
mylist = []
mytuple = ()
for i in range(37):
    mylist.append(i)

#==> ENTER YOUR CODE HERE
mytuple = tuple(mylist)

# Answer:
print(mytuple)         # (0,1,..............36)
print(type(mytuple))   # tuple




#=======================================
#=> 3.0 Pillar: Built-in objects - Sets
#========================================
''' 3.1 Quickly explain what this statement is doing

    random.randint(len(mydict),len(df0['script']))

    3.2 What does the type() function tell you and why is it
        important?

    3.3 Create one set from =mydata1 and mydata2
    3.2 Use the type() function to prove it is a set
    3.5 Why is performing housekeeping a good habit?'''
#====================================

import random  # generates random numbers
               # randint(start value, end value)
mydata1 = random.randint(len(mydict),len(df0['script']))
print(len(mydict),len(df0['script'])) #4, 37

#==> 3.1 ENTER YOUR RESPONSE HERE
'''pulling random value from 4 to 37'''

#==> 3.2 ENTER YOUR RESPONSE here after the 3 lines of code
type(mydata1)
mydata1 = (mydata1,)
type(mydata1)
```

```
'''can only add objects that are the same object type'''

#==> 3.2 ENTER YOUR RESPONSE HERE
mydata2 = 1,2,3,4,3,2,1
myset = set(mydata1 + mydata2)

#...ANSWERS:
#Answer: <your code answers should be the same except m
       #each person will have 1 diff value
print(mydata1,set(mydata2))     # 35, {1,2,3,4}
print(myset)                    # {1, 2, 3, 35, 4}
print(len(myset))   # 3.1 => 4
print(type(myset))  # <class 'set'>

#Answer built in objects only take one parameter.
# BUT you can add objects together as long as they are the same
# object type.

# housekeeping
#Why: so dont absob data you dont need later by accident
del mydata1; del mydata2;del myset

#====================================
#= 4.0 Pillar - interpreting packed built-in objects
'''Task: you have the following object visible to your in your
   'variable explorer' window. if script is in the ... describe
   the object container around it and what you would do to
    unpack it.'''
#====================================
'''             [({...})],   '''
#====================================
the string data is in a dictionary
which is inside a tuple
which is inside a list
```

Wk	Focus & Medium	Weekly Topic & Assignment
7 Oct 10 to 15	3 of 4) classes!	

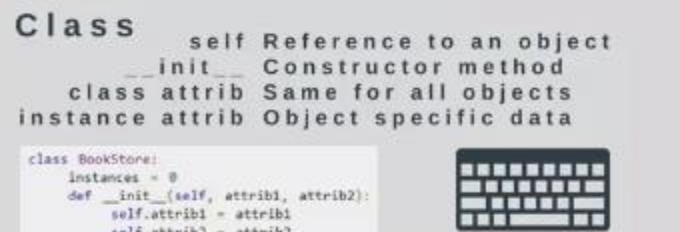
```
"""# -*- coding: utf-8 -*-Created on Mon Oct 12 10:59:53 2022
@author:17574 b.hogan@snhu.edu it.304.fall.22
# WEEK 7 CODE final - including classes """
                        #==========================================
                        #=>week 7  Object Classes Overview
                                #====================================

Lexical Analysis
        always remember about indent \ dedent!
        if you copy and paste and teh spacing is wrong it wont run

https://python.readthedocs.io/en/latest/reference/lexical_analysis.html

#Create a report structure
mydict = {"training done":[], "total animals":0}
class myFarm:    #create parent class object
    pass
    name = ""
    species = ""
    train = ""
def add_train(traintype):    #create a user function to count, sort
    mydict["training done"].append(traintype)
    mydict["total animals"] =+1

#------------->   #children instantiate from parents
a1 = myFarm()     # instantiate children objects from parent, a for animal
a2 = myFarm()     # all object names are user defined
```

7

Oct 10 to 15

```
#update attributes
a1.name = 'mackenzie'  #object.attribute or object.function
a1.species ='dog'
a1.train = 'speak'
add_train(a1.train) #cheCK-OUT!    <only here bc space>

a2.name = 'vinny'
a2.species = 'horse'
a2.train = 'jumping'

add_train(a2.train) #'''function accepts attribute to update dictionary
object'''

#write a simple report using a dictionary data object format
mydict_rpt = {a1.name:a1.species, a2.name:a2.species,"metrics=>":mydict}
mydict_rpt
 '''{'arnold': 'dog','vinny': 'horse','metrics=>':
     {'training done': ['catch', 'jumping'], 'total animals': 1}}'''

#use object's constructors to view its contents
print(a1.__dict__,a2.__dict__)
   ''' {'name': 'arnold', 'species': 'dog', 'train': 'catch'}
     {'name': 'vinny', 'species': 'horse', 'train': 'jumping'}'''

dir(a1)
['__class__', '__delattr__', '__dict__','__dir__', '__doc__',
 '__eq__', '__format__', '__ge__', '__getattribute__', '__gt__',
 '__hash__', '__init__', '__init_subclass__', '__le__', '__lt__',
 '__module__', '__ne__', '__new__', '__reduce__', '__reduce_ex__',
 '__repr__','__setattr__', '__sizeof__','__str__', '__subclasshook__',
 '__weakref__',
 'name',
 'species',  'train']
```

```
#============================================================
                        #=>Week 7 Objects part II
        #============================================================

#==> this is using programmming construct of .self.

class dog_train:
    name = ""
    num_fetch_train = 30
    num_fetched = num_fetch_train
    trainer_ok = 0

    def fetch_train(self, num_balls):
        self.num_fetched = self.num_fetched - num_balls
        if self.trainer_ok == 0 and self.num_fetched <= 0:
            return "sorry! {} not fetch trained. {} balls over a target of
{}".format(self.name,abs(self.num_fetched),self.num_fetch_train)
        elif self.trainer_ok == 1:
            return "Whew! {} passes training after {} balls".format(self.name, abs(self.num_fetch_train-
self.num_fetched-1))
        else:
            return"{} on target to pass fetch train with {} balls
left".format(self.name,self.num_fetch_train-self.num_fetched)

dog1 = dog_train()
dog1.name = "cheeseman"
print(dog1.fetch_train(9))
print(dog1.fetch_train(31))
```



Class

self Reference to an object
__init__ Constructor method
class attrib Same for all objects
instance attrib Object specific data

```
class BookStore:
    instances = 0
    def __init__(self, attrib1, attrib2):
        self.attrib1 = attrib1
        self.attrib2 = attrib2
```

```
dog1.trainer_ok = 1
print(dog1.fetch_train(1))
============================
```

Class, object, and function definitions:

Classes – are a framework or template for creating objects, attributes, and methods.

Objects – are the actors performing work. Child objects instantiate from parent objects and may contain their attributes and methods or have distinct attributes and methods.

Methods - are object instructions detailing how to perform behaviors in a class such as data arrangement, computation, printing, and conditional logic trees, perhaps to test, parse, or look for specific information. Methods do not have to return a value!

Functions – a set of instructions to accomplish a task independent of an object and typically part of a program. They may accept arguments and always return a value.

Class attributes – user-defined names that describe features of a class, and methods can use their values. For example, an object's unique ID, color, name, or numeric value for use in a calculation.

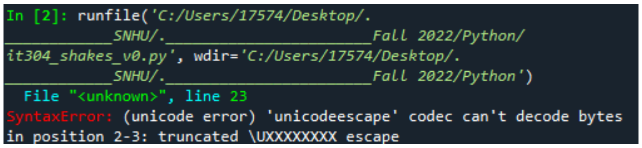
.self <self.attribute> is the first argument in a class function identifying its own attributes.

Essential Python tools associated with objects.,

Built-in types – Python core boolean, comparators, numeric types, and operations like 1+1, iterator types, and operations. REVIEW recommended!

Python Essential Data structures – lists, tuples, sets, dictionary, looping, more on conditionals. Methods and tips and tricks.

Wk	Focus & Medium	Weekly Topic & Assignment

5
9-
26
-
10-
1
	(2 of 3)
	Shakespeare Corpus
	Class Team Coding
	09-28-022
	Step 1: libraries
	#dataframe library
	import pandas as pd
	#numeric library
	import numpy as np
	#visualization library
	import matplotlib.pyplot as plt
	#operating system
	import os
	➢ Reading the data
	➢ Use conditional to loop words
	➢ Make fun graph
		Objective = begin working with 5 pillars of python; create data folder on c:\drive. Code -> Interpret ->
		
		Step 1: change directory, get corpus file path
		import os #operating system library
		os.getcwd() #command to get workiing directory
		q1> What do bad characters in your paths do? A: cant read data
		
		In [2]: runfile('C:/Users/17574/Desktop/.
		___________SNHU/.____________________Fall 2022/Python/
		it304_shakes_v0.py', wdir='C:/Users/17574/Desktop/.
		___________SNHU/.____________________Fall 2022/Python')
		  File "<unknown>", line 23
		SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 2-3: truncated \UXXXXXXXX escape
		os.chdir('c:\\Users\\17574\\Desktop\\data') #msft uses two\\
		os.getcwd()
		df0 = pd.DataFrame() #ensure data going into a dataframe
		#raw_data = pd.read_csv("shakes_corpus_v0.csv") #oops doesn't work
		df0 = pd.read_excel("shakes_corpus_v0.xlsx") #this works!
		df0.info()
		<class 'pandas.core.frame.DataFrame'>
		RangeIndex: 37 entries, 0 to 36
		Data columns (total 3 columns):
		 # Column Non-Null Count Dtype
		--- ------ -------------- -----
		 0 name 37 non-null object

- transpose data between lists, dictionary, string, tuple

OUT[10]: **tip!** going forward will use python [out] to signify output

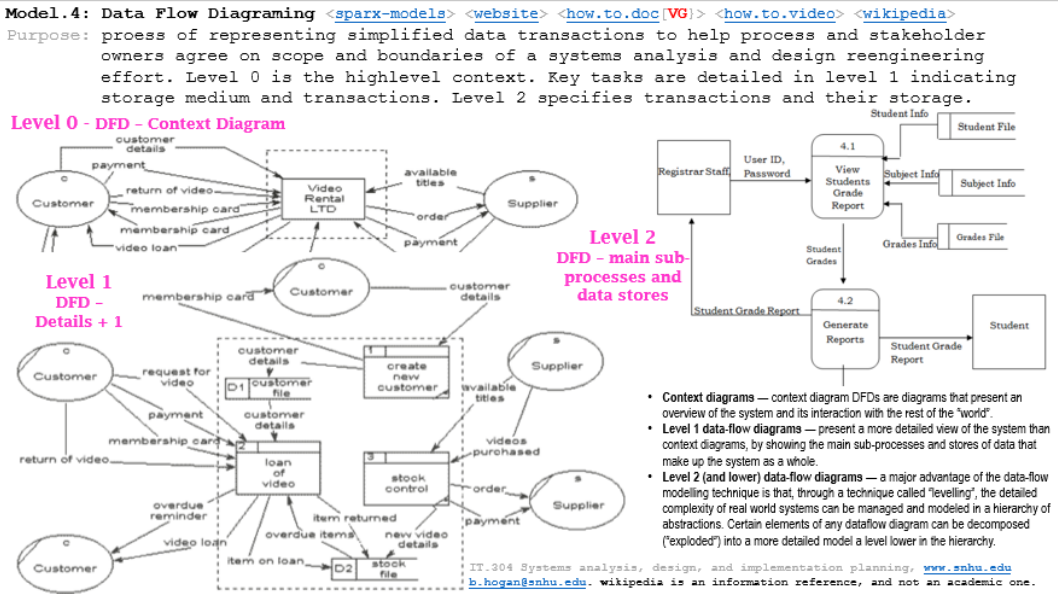
#now as a class we will experiment with cheatsheet

```
             1   script  37 non-null     object
             2   type    37 non-null     object
            memory usage: 1016.0+ bytes
type(df0)                                         #pandas.core.frame.DataFrame
df0.head(2)                           name  ...    type
            0  Alls Well That Ends Well  ...  Comedy
            1             As You Like It  ...  Comedy
```

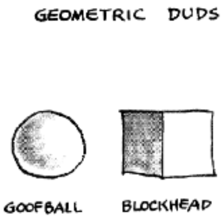
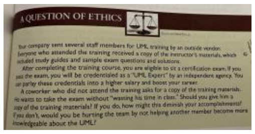
q2> What happens when you dont have a cheatsheet and need to convert a dictionary to a list?[Python Convert Dictionary To List - Python Guides](#)   A: === ACTION = email brian this answer <==========ACTION

```
mydict = df0.to_dict()
print(mydict.keys())
out[10]: dict_keys([‘title’, 'script', 'type'])

mylist_keys = list(zip(mydict.keys())) #hmm my data columns looks good
mylist_keys
OUT[10]: [('name',), ('script',), ('type',)]
#DANGER Will Robinson this is a megasaurus
mylist_values = list(zip(mydict.values())) #holy cow this is huge!
mylist_values=====> this is huge, make sure you undertand

#finally break data into more manageable things to do
mydict.get('title')  #learn a new function
play_names = [mydict.get('title')]
play_names        OUT[10]: [{0: 'Alls Well That Ends Well',
                             1: 'As You Like It',
                             2: 'The Comedy of Errors',

for i in play_names:
    print(i)
                 Out[27]:  [{0: 'Alls Well That Ends Well',
                            1: 'As You Like It',
```

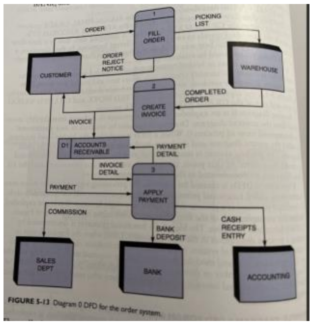
Wk	Focus & Medium	Weekly Topic & Assignment
4 9/19 - 9/24	**Overview** **Python 101 coding**	Orientation to core Python functionality the course will use for system analysis and design projects. The codebook details core data objects, functions, iterators, conditionals, dataframes, and ETL. In short, everything you need to be successful in class and as an entry-level IT professional. Your objective is to "re-type" the code and bring class your learnings and questions for any code you do not understand. You are not learning code from scratch, but you need to understand and intuit the mechanics of iterators,

	wk4 Assignment [Model.4.DFD Data Flow Diagram]()	if.elif.else conditions, and functions to perform work computational work effectively. I am 99.9% confident everyone can complete this work, and I hope everyone will have fun doing so. Good writing is good thinking, and good programming helps make IT work more meaningful and enjoyable. The latest version of the codebook, called the zipper, is in the [bh.github](). Enjoy the printed codebook handouts but ensure to update and print another copy in the upcoming weeks. The latest copy is always on the class git. Thank you for thoughtfully working through all codebook examples. Think about what the code is doing inside the computer. Write down anything that doesn't make sense for class discussion. Over the next few weeks we will learn the 7 pillars of python to build your representation of an data flow diagram.  **Model.4: Data Flow Diagraming** <[sparx-models]()> <[website]()> <[how.to.doc]()[VG]}> <[how.to.video]()> <[wikipedia]()> Purpose: proess of representing simplified data transactions to help process and stakeholder owners agree on scope and boundaries of a systems analysis and design reengineering effort. Level 0 is the highlevel context. Key tasks are detailed in level 1 indicating storage medium and transactions. Level 2 specifies transactions and their storage. Level 0 - DFD – Context Diagram customer details payment Customer return of video membership card membership card video loan Video Rental LTD available titles order payment Supplier Level 1 DFD – Details + 1 membership card Customer customer details request for video customer details D1 customer file create new customer Customer payment customer details membership card return of video loan of video Supplier available titles videos purchased stock control order payment Supplier overdue reminder video loan item returned overdue items new video details item on loan D2 stock file Customer Level 2 DFD – main sub-processes and data stores Registrar Staff User ID, Password 4.1 View Students Grade Report Student Info Student File Subject Info Subject Info Grades Info Grades File Student Grades Student Grade Report 4.2 Generate Reports Student Grade Report Student • **Context diagrams** — context diagram DFDs are diagrams that present an overview of the system and its interaction with the rest of the "world". • **Level 1 data-flow diagrams** — present a more detailed view of the system than context diagrams, by showing the main sub-processes and stores of data that make up the system as a whole. • **Level 2 (and lower) data-flow diagrams** — a major advantage of the data-flow modelling technique is that, through a technique called "levelling", the detailed complexity of real world systems can be managed and modeled in a hierarchy of abstractions. Certain elements of any dataflow diagram can be decomposed ("exploded") into a more detailed model a level lower in the hierarchy. IT.304 Systems analysis, design, and implementation planning, [www.snhu.edu]() [b.hogan@snhu.edu](). wikipedia is an information reference, and not an academic one.

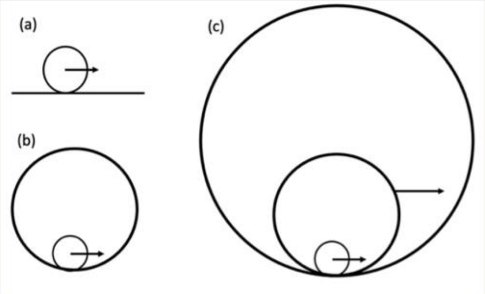
Wk	Focus & Medium	Weekly Topic & Assignment
3 **9/12** **-** **9/17**	**Focus** **Overview**	• Perform hands-on activities in Python to learn [object-oriented programming]()(OOP) working with strings, dictionary, tuple, list, set, function, and objects. • As a team, outline system and code objects to simulate system analysis exercises. • Code is provided for you to re-type and learn. • Use cases will grow your confidence. Tilley details old and new techniques for systems modeling, like business process modeling (BPM) (ch1-2), data flow diagrams (DFD) (ch4), and data and process modeling (ch5). Exercises focus on techniques but with little substantiated in the field outcomes.

	Reading Tilley, Ch6 entire chapter  **ethics discussion text tilley p196**  A QUESTION OF ETHICS Your company sent several staff members for UML training by an outside vendor. Everyone who attended the training received a copy of the instructor's materials, which included study guides and sample exam questions and solutions. After completing the training course, you are eligible to sit a certification exam. If you pass the exam, you will be credentialed as a "UML Expert" by an independent agency. You can parlay these credentials into a higher salary and boost your career. A coworker who did not attend the training asks for a copy of the training materials. He wants to take the exam without "wasting his time in class." Should you give him a copy of the training materials? If you do, how might this diminish your accomplishments? If you don't, would you be hurting the team by not helping another member become more knowledgeable about the UML?	Python hands-on OOP work will replicate varying Tilly processes, such as pg 155-163, with Python data objects (strings, list, etc.), building knowledge of what programmers do. It connects you closely to realistic outcomes of systems analysis and design work. And position you to learn quickly any systems anal. method. A final benefit of the Python OOP work is today's systems analysis, and design do a lot of work extracting and translating information. The result is challenging, but you will know more about it and how not to perform senseless internet searches looking for ideas.approaches to tackle it. **Tilley, Ch6: Overview** • The chapter does an excellent job detailing the components with little to no "geometric duds." • Notice by end of chapter everything you have done to this point is repeated here. Curious! **Python Training:** • By Wed you will be provided with customized training to support this work. It will have all that you need. • Python crash course link below is good to reference and see examples for lists, loops, and similar. Feel free to dig into. • Real world python is super fun training exercises. **Other reference materials** • Matthes, E. (2019), Python Crash Course • Real world Python – FUN training examples • Matthes, Alien Invasion, Ch12. ○ Note: custom materials being provided replace Matthes chapters 1-11. Good to skim by priority: Ch:9,1,3,6 **Nothing due / Reading Only!** Class will start off discussing pg 196 ethics case study so please simply have your thoughts organized on that.

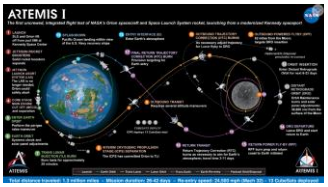
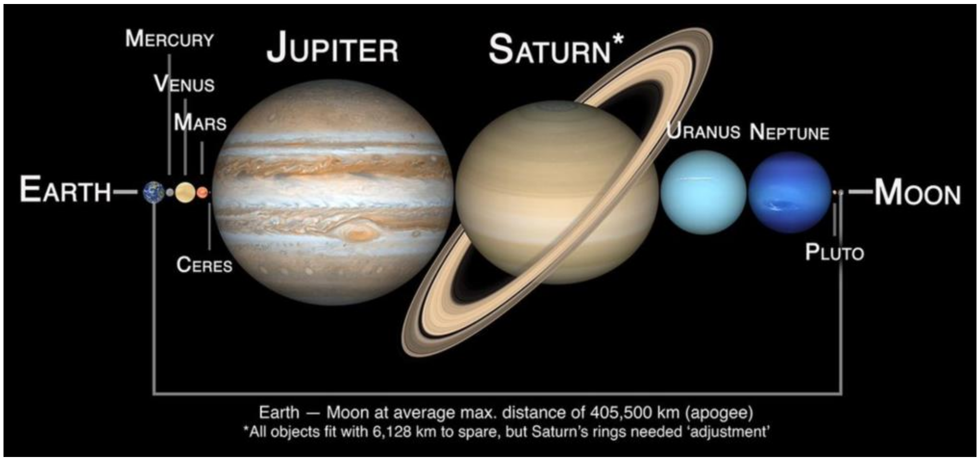
WK	Focus & Medium	Weekly Topic & Assignment
2.2 **9/9**	Focus / Goal lecture notes Reengineering Work: Don't Automate, Obliterate by Michael Hammer **Michael Hammer acticle**	**Goal:** wrap-up historical influence of business process reengineering • **lecture notes:** BPS's evolution with invention of machine learning and data warehousing. The institutionalized game changer of Amazon's kiva robotics **Ch5: data and process modeling** • data flow diagramming uses mostly an agreed upon set of symbols to represent processes, data flows, data stories and entities like transactions or physical items like a deposit ticket and goods.

	 ORDER 1 FILL ORDER PICKING LIST CUSTOMER ORDER REJECT NOTICE WAREHOUSE 2 CREATE INVOICE COMPLETED ORDER INVOICE D1 ACCOUNTS RECEIVABLE PAYMENT DETAIL INVOICE DETAIL 3 APPLY PAYMENT PAYMENT COMMISSION BANK DEPOSIT CASH RECEIPTS ENTRY SALES DEPT BANK ACCOUNTING FIGURE 5-13 Diagram 0 DFD for the order system. **Assignment** **A. Reading** ○ **Tilley,Ch5** **B. Install Python** **Good luck w install!**	• the goal is to represent the information to be encoded by database programmers and develop apps that negotiate the transactions. • this class is less concerned on formality of box symbols but use circles to start and end a process, diamonds for decisions and rectangles for activities. • pg 153, agreed! try not to cross lines when building. • pg 155-159 does a nice job representing an actual system we could easily and realistic code for on hands-on python activities. • Unlike the book are goal is not to "write" about doing this work but actually code it using standard python data objects of lists, strings, dictionaries, tuples, and sets. a)**Reading:** Tilley, ch5, pgs 144-163 b)**Install** **Python** • **Please watch video (i). The best course of action is installation via anaconda b.c it is engineered to auto-fix MANY challenges. However, if done wrong, the 1st time may take => 2-3x more work/time to fix. You "do not" have to figure this out yourself so please reach out with any questions.** i. 1.3M views on YouTube: Install Anaconda Python, Jupyter Notebook And Spyder on Windows 10 - YouTube ii. good start place = jupyter notebook classic home iii. Jupyter :: Anaconda.org **Python cloud** • **online\cloud Jupyter Notebook**: • online alternative – works great ! • https://jupyter.org/try-jupyter/lab/ • JupyterLite — JupyterLite 0.1.0-beta.12 documentation

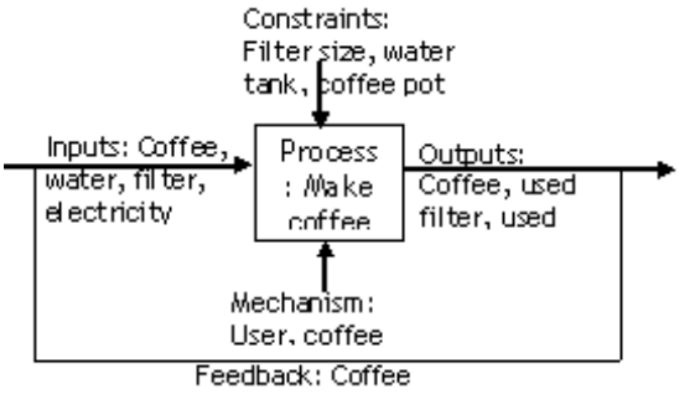
Wk	Focus & Medium	Weekly Topic & Assignment
2.1	Overview Podcast / Video Run videos at speed 1.25 Focus / Goal	**Ch2: Overview** ○ ch2 directs focus to business cases and how to identify a system for analysis. It augments learnings with factors contributing to project success/failure, purpose+ how.to a perform feasibility study, align priorities, and perform an preliminary investigation. ○ Section 2.9, "Preliminary Investigation" (p.26), outlines your revolving course focus building skills and techniques in ○ **Abstraction:** Which tool-kit model will help me quickly assess the situation asked of me? ▪ Quick assessments illustrate your ability to another party to grok salient factors, exercise skill by presenting a visual or data dashboard, and communicate back to manager or stakeholder.

	Model.2:SWOT Model.2:SWOT. Decision.Book perception... cartoon 	▪ *Why should person X trust you*? Your responsible for building trust b/c it gets you access to more resources and what you need most, time. ○ **Data:** What data collection strategy will help me access inputs, outputs, resources, and constraints? ○ **Situational awareness:** After presenting initial response to business owner, what kind of model support, time, and resources do I have? Do I need? ✓ info.Tech resources usually can help get process metrics, source metric data, and any other information to meet your analysis goals. ✓ Data not what you need? Initiate estimation work. ✓ Today, operations often have project planning documents associated with the system workflow you should inspect while applying your abstraction work. ✓ **SWOT.** When in doubt fall back to basics to help assess a situation’s status with strengths, weaknesses, opportunities, and threats(tilley.45, krogerus.tschappelerp.12). **Perception & time <philosophy>:**  **Figure 3.** Illustrating how a hierarchy of specious presents and the passage of time may be represented by a sequence of compact dimensions in relative motion. (a) corresponds to SP_1, (b) to SP_2, (c) to SP_3, etc. link physical space, perceptual space, and memory ○ the course is not designed to dive deep into perception, time, and points of view. For systems modeling, learn to hone your logic representation skills ***and*** figure what you missed. ○ Do individuals experience time similarly? Does time affect perception? Quality of shared information?

Week	Focus & Medium	Weekly Topic & Assignment
1.2	Model.3: Swimlane IT Order Harmonization Example model.3.swimlane <bh.github> <how.to.doc> <wikipedia>	**Model.3.Swimlane** **Purpose:** use horizontal or vertical gradating color bars to demarcate business lines illustrating system inputs, activities, and decisions connected with arrows. **Assignment: Tilley Ch2 + Roughcut Swimlane diagram** ➢ Swimlanes no longer have notoriety as in 1993, and some IT professionals view them as a hindrance to what they need, that is, codified information. ➢ However, swimlanes are super at helping a senior manager or new employees quickly grasp what an organization is doing and how they are doing it. ➢ **“”””You’re the only resource, but you can have and do anything you want to do. Please include,””””**

	sorry! in github you have to download to get link to work or use them here [Artemis I Space Launch System unmanned Moon mission]()  Swimlane Assignment request by 9/6 @6ish PM	➢ You're the only resource but can have, and do, anything you want to do. Please include, ✓ Square(ish) boxes to represent activities ✓ Lines to connect between activities ✓ Line arrowheads to show directionality between shapes ✓ Diamond(ish) boxes to represent decisions ✓ Text in squares + diamonds + on lines to detail happenings ✓ Optional: add a numeric index for each box & feel free to annotate "anyway" you like.  MERCURY JUPITER SATURN* VENUS MARS URANUS NEPTUNE EARTH— —MOON CERES PLUTO Earth — Moon at average max. distance of 405,500 km (apogee) *All objects fit with 6,128 km to spare, but Saturn's rings needed 'adjustment' **Example:** **Earth:Launch ↓** **Mars: Fuel up -> Open solar flares 3 yrs ↓** **Neptune: Turn into nano-space particulates** ❖ **Please email a .jpg, pdf however you build it.** ○ **File\SaveAs\often allows you select type .pdf** **-->'The goal is to be more thoughtful of your logic'<--**

Week	Focus & Medium	Weekly Topic & Assignment
1.1	Reading Podcast / Video What is business process re-engineering? **Run videos at speed 1.25** What is a system? inputs outputs resources constraints	**Tilley, Ch 1.** [Intro to Systems Analysis]() (free link) • 1st chapter is FREE !, use above link • Awareness & Design – Michael Hammer ○ https://www.youtube.com/watch?v=9oxM5JV7H5O • [Business Process Re-engineering explained -]() ○ https://www.youtube.com/watch?v=v-jAf7L2Uak ▪ (10.5min/1.25=8.4min) • IBM Business process Analysis (6.5min/1.25=5.2min) ○ https://www.youtube.com/watch?v=1E6II2U1shY Utilize your abstraction instinct while reading because the name "EMS" isn't important, but the concepts are. https://www.niu.edu/ems/introduction/definition.html 1) definition is page 1 + 8 more pages using <next topic> 2) The EMS model 3) Benefits of EMS

		4) Examples of EMS 5) Systems approach 6) Concept diagram <focus and perform abstraction here> 7) Processes, inputs, outputs a. Example of: inputs, outputs, resources, constraints 8) Summary
	IDEF0 Handout	• IDEFØ – Function Modeling Method – IDEF - website ○ 2nd example of input, output, res., constraint
	Assignment Request for 9/1	Select a process you love or dislike. Define its input, outputs, resources, and constraints (IORC). Logically what goes into the system is either consumed or comes out. Notate ALL you think of. Then, list 5 to 10 high-level activities performed by the IORC. Use paper and pencil and send me a picture **anytime** end of the day tomorrow. I am only asking for a max of 15 min to whip up. Please spend more if having fun. Thank you for considering this fast turnaround, as I will use all work submitted to start Friday's lecture. Perform work as a team as desired or convenient.
	Assignment Example page Assignment example **Model.1:IDEF0**	https://www.niu.edu/ems/introduction/constraints.html  Constraints: Filter size, water tank, coffee pot Inputs: Coffee, water, filter, electricity Process : Make coffee Outputs: Coffee, used filter, used Mechanism: User, coffee Feedback: Coffee

References