Week 3

Day 1

Python

Data Analytics and Python –

*Programs* *and* *algorithms* – algorithm is a series of steps to solve a problem, a program is a series of instructions telling a computer how to solve a problem

*Python* – high level language, meaning mor natural language, takes less time to write, shorter and easier to read, and more likely to be correct and error free when code is ran

Two ways computers process program code –

*Interpreted* *languages* – interpreters process programs a little bit at a time, reading lines and performing computations

*Compiled* *Languages* – see slides for info

Python is an interpreted language – see slides for info

*Types of languages* – natural are mostly spoken; formal are designed by people like programming languages

**Vocab**

*VALUE* – number or string that can be stored in a variable or computed in an expression

*VARIABLE* – name that refers to a value

*Variable* Name – name given to variable

*Str* – python data type that holds a string of characters

*Operators* – special symbols that represent a simple computation

*Data* Type – set of values

*Comment* – information in a program that explains what certain code does

*Input* – command in a program that prompts the user to put in an answer

*Output* – result of the program

Graphical user interface, text

Description automatically generated

Different ways to code – Integrated Dev Environment IDE

Command line interpreter

*IDE* – *REPL* – Reads evaluates prints loops

*Python* *in* *VS* *Code* – integrated dev environment is an app that combines all the functions or REPL as well as an editor with which you can create and modify code to then submit to the interpreter for execution. Notable Features : syntax highlighting, context sensitive help, code completion, debugging

*Debugging* – process of detecting and removing existing and potential errors from software code, its an essential skill, both a science and an art, needs to be done as you code thinking of terms such as what could go wrong and how can I prevent it from happening, complexity is ruins efficiency, issues could be large or small

Tips – regularly test and review codes, chunk code into smaller sections and ensure each section works, explain your code aloud to yourself barney style, work backwards to try finding the issue, take a break and come back to it when you run out of ideas, ask someone to help look

**Common** **Errors**

*Syntax* – mistake in code like spelling punctuation spacing and indentation

*Runtime* – occur during running or execution, might run indefinitely or crash

*Semantic* – code is typically grammatically correct, but its not coming out with the expected results

*Logical* – do not accomplish intended goal, calculations are off

Jupyter Notebook – open source web application used to create and share documents that contain : live code, equations, visualizations, text; browser based tool for interactive authouring of documents combining text, mathematics, code, computations, and rich media output

Notebook Documents: a representation of ----on video

**TO RUN JUPYTER RUN** python -m notebook