Chapter 1

- Create a script that can find all lines in the file pythonbeginners.txt that has the word "python" in it
- Change the script so it can get the search word and filename from commandline

Continuous exercise

We will make a bookmaker system. First step is to construct the commandline interface. The usecases are:

- `python bookie.py --init --initial-asset 100000`
- `python bookie.py --new-race --race-name 'Big Darby'`
- `python bookie.py --add-horse --race-name 'Big Darby' --horse '1:Xerxes' --odds 3.1`
- `python bookie.py --add-horse --race-name 'Big Darby' --horse '2:Troudeau' --odds 2.6`
- `python bookie.py --add-horse --race-name 'Big Darby' --horse '3:Frederix' --odds 4.7`
- `python bookie.py --add-horse --race-name 'Big Darby' --horse '4:Equinox' --odds 11.0`
- `python bookie.py --add-player --player-name 'Jack Norman' --deposit 1000`
- `python bookie.py --add-bet --player-name 'Jack Norman' --race-name 'Big Darby' --horse 1 --amount 75`
- `python bookie.py --race-result --race-name 'Big Darby' --horse 1`
- `python bookie.py --print-status`

Use argparse to handle commandline arguments

Chapter 2

- Create a module that can convert between Celsius and Fahrenheit, inches and millimeters, and gallon and liters
- Create a script that uses the functions in the module
- Convert the module to a package with the three conversions in separate files

Continuous exercise

- Restructure the script `bookie.py` into a package with modules for each functionality.
- Make the remaining of the script into a `main()` function called from a `if __name__ == "__main__":` block.

Chapter 3

- Read in the file passwd.txt and print the lines
- Read the file and place the data in a list of tuples
- Read the file and place the data in two dictionaries of named tuples, one dictionary with login as key, the other with uid as key
- Make an Enum with 1 being bash, 2 being ksh, 3 being tcsh, 4 being zsh. Use it in the data structure

Continuous exercise

Create datatypes for these elements:

- Horse (name, owner, rating)
- Race (name, horses, status, winner)
- Race status (planned, bet-able, started, finish, cancelled)
- Bet (race, horse, amount, redeemed)
- Player (name, balance, bets)
- Bank (balance, players, races)

Chapter 4

- Create a function that can calculate numeric integration.
  - It should take four parameters: the function, lower limit, upper limit, and the width of the segments
- Create a list comprehension to calculate the leap years from 1588 to 2024. Take the hundred years rule and four hundred years rule into account

Continuous exercise

Make a function, `payout(bank)` that ... * Goes through all players and all bets of the player * If a race is finished and the bet isn't redeemed evaluate the bet * If the bet is won, calculate the payout and add to the balance of the player * Set the status of the bet to redeemed

Chapter 5

- Make a class for a book and a library
- Make methods to add a book to a library
- Make a class for periodicals (magazines) and a parent class for publications (books and periodicals)

Continuous exercise

- Rewrite the bookmaker system as object oriented
- The datatypes should be made into classes

- The functions should be made into methods

Chapter 6

- Create a function that reads the `numbers.txt` file, adds the numbers, and returns the sum of the numbers.
- Add error handling so that the lines that are not numbers are ignored.
- Create a main part that asks for the filename and handles if the filename does not exist.

Continuous exercise

What part of the `bookie` project may raise an exception? Add exception handling all necessary places

Chapter 7

- Create a script that finds all python files and prints the filenames and the number of lines in the file

For the next exercises these data can be used:

```
data = [
    {'name': 'Andrew', 'age': 34, 'height': 1.82},
    {'name': 'Ben', 'age': 51, 'height': 1.81},
    {'name': 'Charlie', 'age': 72, 'height': 1.84},
    {'name': 'Dennis', 'age': 31, 'height': 1.78},
    {'name': 'Eric', 'age': 45, 'height': 1.83},
]
```

- Create a script that dumps the above data to three files named `data.json`, `data.csv` and `data.pickle`.
- Create a script that gets a filename from commandline and reads it in as either JSON, CSV, or Pickle depending on the filename extension

Continuous exercise

- Make a table for each of the datatypes/classes
- Make methods for each class to add, retrieve, update, and delete an object to or from the table

Chapter 8

- Create a virtual environment and activate it
- Install the packages `pytest`, `mypy`, and `black`

Continuous exercise

- Create a package called `bookie`
- Move the files from the previous exercise (or the solution) into the package
- Make a `[project.scripts]` section to make a script
- Build the package

Chapter 9

- Write script with a function that counts the number of vowels (a, e, i, o, u, y) in a text
- Write a unittest file that tests the function in the script

Continuous exercise

- Add unit tests to the `bookie` package.
- Run `black`on the files in the project
- Test for type errors with `mypy`