C200 Programming Assignment № 5
Looping, Conditions and Dictionaries
Spring 2025

Dr. M.M. Dalkilic

Computer Science

School of Informatics, Computing, and Engineering

Indiana University, Bloomington, IN, USA

March 1, 2025

In this homework, you'll start mastering your skill in writing functions. Make sure to start working early on this assignment. If we do not explictly mentioned a function/library to use, you are not allowed to use it. For example, if we have not mentioned to use tkinter library then you should not import tkinter in your python file. If you're found cheating...well, you know the consequences by now which includes using tools to solve any portion of any problem.

As always, all the work should be with you and your partner; but *both* of you should contribute. You must complete this before **Friday, March 07 2025 5:00PM EST**. Please remember that

- You will be submitting on Gradescope
- Your highest score of **10 submissions** will be used for your final score.
- Any function that is defined requires a docstring, please refer to lab 03 for what a docstring should have.
- Finally, manual grading will be done, so if you receive a 50 out of 100 on the autograder, it means the lowest grade you can receive is 50. The rest will come from manual grading of the problems.

If your timestamp is 5:01PM or later, the homework will not be graded. So **do not wait until 4:59 PM** to submit. If you have questions or problems with Gradescope, please visit office hours or make a post on Inscribe ahead of time. Since you are working in pairs, your paired partner is shown on canvas.

Problem 1: Come Fly with Me (Sinatra)

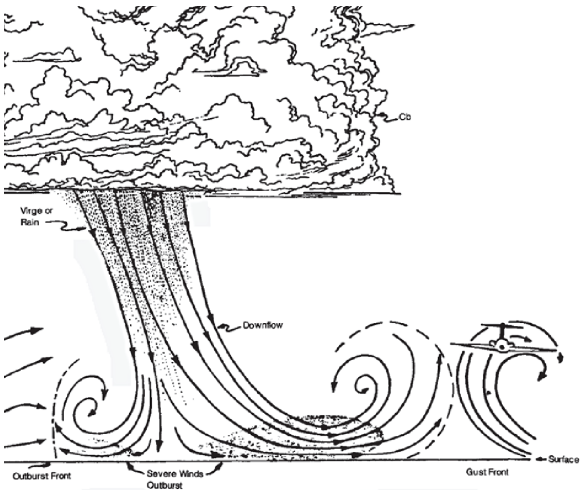

Figure 1: Image taken from FAA

You've been hired by the FAA to calculate wind shear contants. Wind shear is a meterological phenomenon where the velocity of wind v_0 at height h_0 and the velocity of wind v_1 at height h_1 differ significantly. Wind shear affects airplanes as they take-off and land. The formula for verticle wind sheer is:

$$\frac{v_0}{v_1} = (\frac{h_0}{h_1})^P \tag{1}$$

Implement a function P(v0,h0,v1,h1) that takes wind velocities and heights and returns the constant value of P. The following function:

```
v0,h0 = 25,200
v1,h1 = 6,35
print(P_ws(v0,h0,v1,h1))
```

produces

```
0.819
```

Deliverables for Problem 1

- Complete the function
- Round the final answer to three decimal places

Problem 2: Day of the Week

The modulus operator is denoted by the symbol %. For $x\%y$ it returns the remainder after y is divided into x a whole number of times. While you've seen this before, the format is likely different. To refresh your memory you can visit `https://realpython.com/python-modulo-operator/` or do your own search. For this problem, you will also need to use the floor function in python - think of floor as a way to round down a floating point number to the nearest integer. For example, floor(18.90) is 18 and floor(14.25) is 14.

An approximate (works generally okay for a number of dates, but **not** all) formula for computing the day of the week given dlst = [d,m,y] where d is day, m is month, and y is year is:

$$dlst = [d, m, y] \tag{2}$$

$$a(dlst) = y - \frac{(14 - m)}{12} \tag{3}$$

$$x = a(dlst) + \frac{a(dlst)}{4} - \frac{a(dlst)}{100} + \frac{a(dlst)}{400} \tag{4}$$

$$b(dlst) = floor(x) \tag{5}$$

$$c(dlst) = m + 12(\frac{14 - m}{12}) - 2 \tag{6}$$

$$day((d, m, y)) = (d + b(dlst) + (31 \cdot \frac{c(dlst)}{12}))\%7 \tag{7}$$

The function *day* returns a number $1, 2, \ldots, 7$ where $1 = Monday, 2 = Tuesday, \ldots$. If we use this number as the key with the week dictionary, we are able to return the correct day of the week. Here is the dictionary used:

```
week = {1:"Mon", 2:"Tue", 3:"Wed", 4:"Thu", 5:"Fri", 6:"Sat", 7:"Sun"}
```

For example,

$$day([14, 2, 2000]) = \text{Mon} \tag{8}$$

$$print(day([14, 2, 1963])) = \text{Thu} \tag{9}$$

$$print(day([14, 2, 1972])) = \text{Mon} \tag{10}$$

2/14/2000 falls on a Monday; 2/14/1963 falls on a Thursday; 2/14/1972 falls on a Monday.

Deliverables for Problem 2

- Complete the functions described above.
- For calculating b(dlst), you can use the math.floor() function from the math library.
- Remember, the day function utilizes the week dictionary.

Problem 3: Portfolio Valuation

In this problem, you'll implement a function value(portfolio, market) that takes a person's portfolio of stocks and the current market and determines the value as a percent. A stock is a portion of ownership of a business. Since it's a portion, it is a fraction of what the business is worth in total. A stock as a price per share as a dollar amount and the number of shares (how much of the portion is owned). For example, if you have a stock that you purchased for \$2.25/share and you purchased 11 shares then $\$2.25 \times 11 = \$24.75$ in total. A portfolio consists of the stock name, share price that was purchased, and the number of shares. We'll model this a dictionary:

```
portfolios =  {'A':{'stock':{'x':(41.45,45),'y':(22.20,1000)}},
                    'B':{'stock':{'x':(33.45,15),'y':(12.20,400)}}}
```

shows two portfolios, one for 'A' and one for 'B'. 'A' has 45 shares of 'x' valued at \$41.45 and 'y' valued at \$22.20 with 1000 shares. The total value is: $\$41.45(45) + \$22.20(1000) = \$24065.25$. The market value for stocks changes. In this example, our market is:

```
market = {'x':43.00, 'y':22.50}
```

This means that 'A's holdings in 'x' is worth now $\$43.00 \times 45 = \$1935.0$ instead of $\$41.45 \times 45 = 1865.25$. The value of a portfolio is the ratio of $\frac{\text{Market}-\text{Portfolio}}{\text{Portfolio}}$. If we were solely looking at the 'x' stock, then

$$\frac{\text{Market} - \text{Portfolio}}{\text{Portfolio}} = \frac{43.00(45) - 41.45(45)}{41.45(45)} \tag{11}$$

$$\approx 3.7\% \tag{12}$$

which means the value today is 3.7% more. Implement the function value(portfolio, market) that determines the change in percentage of worth. For example,

```
portfolios =  {'A':{'stock':{'x':(41.45, 45),'y':(22.20, 1000)}},'B':{'stock':{'x':(33.45,15),'y':(12.20,400)}}}

market = {'x':43.00, 'y':22.50}

for name, portfolio in portfolios.items():
    print(f"{name} {value(portfolio,market)}")
```

produces

```
A 2.0
B 79.0
```

that gives 2% for A and 79% for B.

Deliverables for Problem 3

- Complete the function.
- The function returns a percent.
- Round to two decimal places.

Problem 4: Smoothing Data

When data is temporal (ordered by time), we sometimes want to "smooth" it by giving a simple average. The simplest is taking two consecutive values and finding their average. Given a (possibly empty) list of numbers $\mathsf{lst} = [x_0, x_1, \ldots, x_n]$ the function smooth(lst) returns:

- The string "AveError: list empty" if the list is empty
- lst if list only contains one number
- a new list of numbers $[y_0, y_1, \ldots, y_{n-1}]$ such that $y_i = \frac{x_i + x_{i+1}}{2}$

The following code

```
1 data = [[], [1],[1,2],[1,2,2,3],[0,2,4,6,8]]
2 for d in data:
3     print(smooth(d))
```

produces

```
1 AveError: list empty
2 [1]
3 [1.5]
4 [1.5, 2.0, 2.5]
5 [1.0, 3.0, 5.0, 7.0]
```

Deliverables for Problem 4

- Complete the function.
- Round to two decimal places.

Problem 5: Maximal Value

In this problem, you're given a list of numbers. The function msi takes a list of numbers and returns a list of three values (in this same order) namely, the interval start, interval stop and the value of the greatest sum. For example if the list is: x = [7, -9, 5, 10, -9, 6, 9, 3, 3, 9] then the greatest interval is from x[2:10] giving 36:

$$sum(x[2,10]) \quad = \quad 5+10-9+6+9+3+3+9 = 36 \tag{13}$$

The function should return a list of values i.e., [2, 10, 36]

```
1 x =  [7, -9, 5, 10, -9, 6, 9, 3, 3, 9]
2 print(msi(x))
```

produces

```
1 [2,10,36]
```

For this problem you are allowed to use sum(lst) which returns the sum of a non-empty list of numbers.

Programming Problem 5: Maximal Value

- Complete the function msi(x)
- You are allowed to use the sum function
- You might find the following from the lecture useful:

```
1 >>> x = [1,2,3,-2]
2 >>> sum(x[1:3])
3 5
```

6: While Loops and Roots

Remember that for a real-valued function $f(x)$, the root $x*$ is a value such that $f(x*) = 0$. We saw, for example, that $ax^2 + bx + c = 0$ has roots that can be analytically determined. Here is a very simple technique to discover roots for $f(x)$ if is a continuous function.

```
#initial estimate
x = alpha
#epsilon is how good we want the estimate
while abs(x - f(x)) > epsilon:
    x = f(x)
```

While it doesn't work all the time, for some problems, it does astonishingly well for such simple code. The main task is to find the reciprocal of the highest power. For example, assume $f(x) = x^2 - 4x - 7$. The highest power is x^2 so we isolate that:

$$0 = x^2 - 4x - 7 \tag{14}$$

$$x^2 = 4x + 7 \tag{15}$$

$$x = \sqrt{4x + 7} \tag{16}$$

We can quickly see that $f(4) < 0$ and $f(7) > 0$ so a root must lie between 4 and 7. So we start with 4 as our initial guess. The following code:

```
def f1(x):
    return math.sqrt(4*x + 7)

def approx_root(f,initial_guess):
    ...
x_star = approx_root(f1,4)
print(x_star, x_star**2 - 4*x_star - 7)
```

produces

```
5.3166247040232015 -5.726630121216658e-07
```

We know one of the roots is $\frac{4+\sqrt{16+28}}{2} \approx 5.3166247903554$ which is very close. We do one more example. Let $f(x) = x - e^{-x}$. Then

$$x = e^{-x} \tag{17}$$

Let's start with an initial guess as x = .5. Then

```
def f0(x):
    return math.exp(-x)

```

```
4 def approx_root(f,initial_guess):
5     ...
6 x_star = approx_root(f0,.5)
7 print(x_star, x_star - math.exp(-x_star))
```

produces

```
1 0.5671433381054858 7.474599861279074e-08
```

The other two functions' roots are:

$$f_2(x) = x^4 - 4x - 7 \quad (18)$$

$$f_3(x) = x^5 - 3x^2 - 2 \quad (19)$$

Deliverables for Problem 6

- Complete the functions shown in equations 18 and 19 and implement the approx_root function.
- You can verify your outputs if they're very, very small numbers.
- For checking if the output is small, we can check if it's less than $\epsilon(\epsilon = 1e^{-7})$.