**#1 Pseudocode: 3.11 paint the wall**

* Start
* OBTAIN the measurements for the wall’s height and its width in feet
* Write wall\_Height value from the measurement OBTAINED for the wall’s height.
* Write wall\_Width value from the measurement OBTAINED for the wall’s width.
* CALCULATE the square feet for the area of the wall by multiplying the wall’s height with its width
* Write wall\_Area value from the calculation
* ASSUME one gallon of paint covers 350 square feet
* CALCULATE the amount of paint needed by dividing the wall’s square feet by 350 square feet
* Write paint\_Needed value from the calculation
* Round up the amount of paint needed to find the amount of can(s) needed
* Write can\_Needed value after rounding up
* After getting the value of can(s) needed, go to Home Depot’s paint section
* Once at the paint section, call wife and ask what paint color she wants the wall to be (Input)
* Find Home Depot’s set price for the requested color (Dictionary)
* Grab the amount of 1-gallon paint can(s) for the color requested to get
* Pay the amount of 1-gallon paint can(s) multiplied by the set price for the requested color
* Stop

**#1(A)(B):**

PRACTICES:

The best practices I used for this exercise were the following: Beautiful is better than ugly, simple is better than complex, sparse is better than dense, now is better than never,

INPUT:

1. 12
2. 15
3. Red

OUTPUT:

Enter wall height (feet):

Enter wall width (feet):

Wall area: 180.0 square feet

Paint needed: 0.5142857142857142 gallons

Cans needed: 1 can(s)

Choose a color to paint the wall:

Cost of purchasing red paint: $35

CODING:

1. import math
2. paintColors = {
3. 'red': 35,
4. 'blue': 25,
5. 'green': 23
6. }
7. #(1): Prompt user to input wall's width
8. # Calculate and output wall area
9. wallHeight = float(input('Enter wall height (feet): \n'))
10. wallWidth = float(input('Enter wall width (feet): \n'))
11. wall\_Area = wallHeight \* wallWidth
12. print('Wall area:',wall\_Area,'square feet')
13. # FIXME (2): Calculate and output the amount of paint in gallons needed to paint the wall
14. paint\_Needed = wall\_Area / 350
15. print('Paint needed:', paint\_Needed, 'gallons')
16. #(3): Calculate and output the number of 1 gallon cans needed to paint the wall, rounded up to nearest integer
17. cans\_Needed = math.ceil(paint\_Needed)
18. print('Cans needed: %d can(s)' % (cans\_Needed))
19. # FIXME (4): Calculate and output the total cost of paint can needed depending on color
20. userColor = input('\nChoose a color to paint the wall: \n')
21. price = paintColors[userColor]
22. totalPrice = int(price) \* cans\_Needed
23. print('Cost of purchasing %s paint: $%d' % (userColor, totalPrice))

Problem-solving approaches:

The problem-solving techniques I used to assist me with coding were going back throughout my notes to remember the use of certain expressions and wordings. Ex: **math.ceil** to raise the can needed to a non-decimal number. Another problem-solving technique I would use was a trial and error technique. I would input false information and run the program to see the results of the output. Even though this technique sounds like I go against the Zen of Python practice: “In the face of ambiguity, refuse the temptation to guess”, I do not believe it does because I am not guessing with random information instead I am testing a hypothesis of a certain coding I believe is right and the python error program will me a specific change to my coding. The tools I used was the run program tool and submit program to discover my errors in my coding for the lab activities.