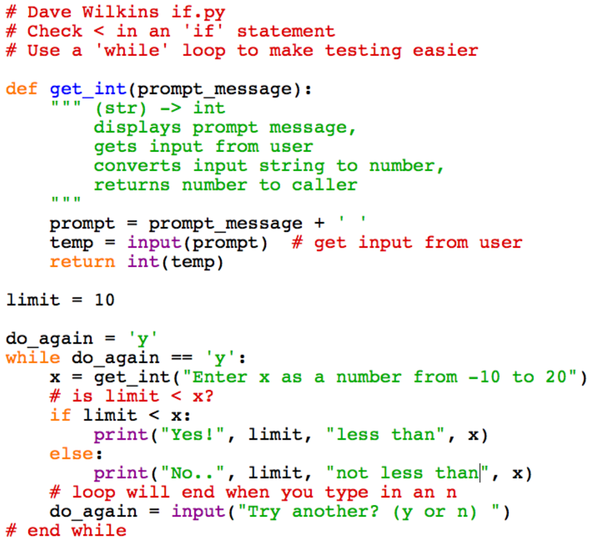
Lab 4 Choices using if, elif, else; bool True / False variables

Goal: **Get a lot of practice** *with Python* **if**: *statements, using* **if**, **elif** *and* **else**, *and using bool variables (with True / False values) to add clarity to more complicated ifs.*

Note: testing a program by running the module, then going back into IDLE to run the module again, and repeating that process several times gets tedious. Here's a solution that lets you run your code several times, until you decide to stop.

You will also want to define a get_float(prompt_message) function that returns a floating point (decimal) number to the caller, similar to the get_int(prompt_message) function shown here.



```
# Dave Wilkins if.py
# Check < in an 'if' statement
# Use a 'while' loop to make testing easier

def get_int(prompt_message):
    """ (str) -> int
        displays prompt message,
        gets input from user
        converts input string to number,
        returns number to caller
    """
    prompt = prompt_message + ' '
    temp = input(prompt)  # get input from user
    return int(temp)

limit = 10

do_again = 'y'
while do_again == 'y':
    x = get_int("Enter x as a number from -10 to 20")
    # is limit < x?
    if limit < x:
        print("Yes!", limit, "less than", x)
    else:
        print("No..", limit, "not less than", x)
    # loop will end when you type in an n
    do_again = input("Try another? (y or n) ")
# end while
```

Lab 4-1.py Implement the code above, and run tests for x set to 9, 10, 11.
Note what happens for these values - add a comment to your code about what happens.

Lab 4-2.py Copy the code from lab 4-1.py and add if tests for **less than or equal to**, and another for **equals** *(remember this is a test for equality, not an assignment statement)*, yet another for **does not equal.**

One input can now be tested for several conditions
less than
less than or equals
equals
not equal

Lab 4-3.py Define a get_float(prompt_message) after the get_int(prompt_message) function. It will look fairly similar, but it returns a float number, not an int.

Change your input to ask for dollar and cents price such as 4.37.
Run the same tests using floating point numbers.

Lab 4-4.py
Open your lab 4-3.py and Save As Lab 4-4.py.
Change your code to assign 'argon' to a variable.
Ask for character input that will be assigned to a second variable.

Is 'argon' < 'A' ?	Also equal? not equal?
Is 'argon' < 'a' ?	Ditto
Is 'argon' < 'Aster' ?	Ditto
Is 'argon' < 'Z' ?	Ditto

Lab 4-5.py Compare 'argon' to integers. Is 'argon' equal to 14? Not equal to 14? Less than 14?

Lab 4-6.py Putting items in the right categories (or "buckets") using if, elif, else

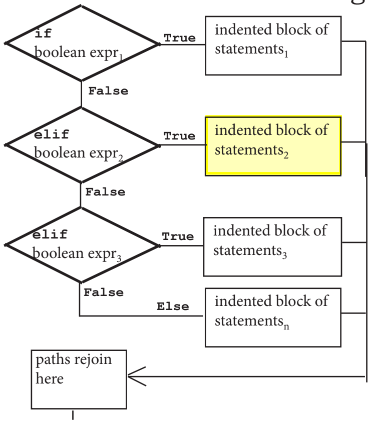
CIS 199 is an online course. Grades are based on project points:

points	Grade
50 or more	S ("Superstar")
40 or more	A
30 or more	B
25 or more	C
under 25	Z

Using the same kind of while loop, ask for total points, then print out the corresponding grade.
Use an if, elif, elif, else structure to assign letter grade to the variable grade.
Print both the points and the grade, somewhat like this.

```
37  B
44  A
 11 Z
```

Here is a flowchart showing a picture of the kinds of decisions and actions in an elif structure


if
boolean expr$_1$
True
indented block of statements$_1$
False
elif
boolean expr$_2$
True
indented block of statements$_2$
False
elif
boolean expr$_3$
True
indented block of statements$_3$
False
Else
indented block of statements$_n$
paths rejoin here

Notice that once an if or elif gets a True result, the indented block of code executes and then control transfers to the very end of the entire if-elif-elif-elif-else structure. In the example above, if the if was not True, but the first elif was true, the code shown in yellow will run, then we skip all the remaining elif and else statements.

Lab 4-7 Estimate risk of heart disease based on 2 factors: age in years, and "body mass index" (BMI) a rough (and sometimes misleading) measure of how skinny or overweight a person is.

Your program will prompt a user to enter a BMI (a number ranging from 0.15 to 0.35 or so).
Next, prompt for age in years.
Using age and BMI, calculate a risk category with if, elif logic like that on page 94.

Risks

age < 45, bmi < 0.22 Low
age < 45, bmi >= 0.22 Moderate
age >= 45, bmi < 0.22 Moderate
age >= 45, bmi >= 0.22 High

You can assign the result of a comparison (a True or False value) to a variable.

```
young = age < 45
slim = bmi < 0.22
```

Now you can use a series of statements such as this

```
if young and not slim:
    risk = "Moderate"
elif ...
```

Print age, bmi, and risk then ask whether to continue with the next person's risk.

Extra credit 10 points
The town of Sparta, Oregon will levy an income tax using the following table

Annual Income	Rate
Not Over	
15000	0.005 (0.5 percent)
17500	0.006
22400	0.010 (1 percent)
47300	0.014
75000	0.018
112450	0.022
277000	0.027
all higher	0.030

Ask for a user's annual income, us an if elif structure to find the rate. Print the income, rate and tax due (rate times income). Repeat until user says to quit.

Other extra credit
1 point each for other if, if else, and if, elif, else short programs.
For example, try using **and or not** with some of your
if statements.

Grading 40 points + XC

Please start 1st line of each program with
# by Jane Student *but using your own name...*

Lab 4-1.py 5 points
1 point input data
1 point while loop to process
3 points if else test(s)
Lab 4-2.py 5 points "
Lab 4-3.py 5 points "
Lab 4-4.py 5 points "
Lab 4-5.py 5 points "
Lab 4-6.py 7 points if elif elif else
1 point input data
1 point while loop to process
5 points if elif elif elif else test

Lab 4-7.py 8 points bool variables for heart risk
1 point input data
1 point while loop to process
1 point young bool variable correctly set up
1 point for slim variable correctly set up
1 point each for if, elif, elif, else tests