IF STATEMENT

1. Write a program that reads an integer from the user. Then your program should display a message indicating whether the integer is even or odd.
2. Write a program that implements the conversion from human years to dog years described in the previous paragraph. Ensure that your program works correctly for conversions of less than two human years and for conversions of two or more human years. Your program should display an appropriate error message if the user enters a negative number.
3. Create a program that reads a letter of the alphabet from the user. If the user enters `a`, `e`, `i`, `o` or `u` then your program should display a message indicating that the entered letter is a vowel. If the user enters `y` then your program should display a message indicating that sometimes y is a vowel, and sometimes y is a consonant. Otherwise your program should display a message indicating that the letter is a consonant.
4. Write a program that determines the name of a shape from its number of sides. Read the number of sides from the user and then report the appropriate name as part of a meaningful message. Your program should support shapes with anywhere from 3 up to (and including) 10 sides. If a number of sides outside of this range is entered then your program should display an appropriate error message.
5. The length of a month varies from 28 to 31 days. In this exercise you will create a program that reads the name of a month from the user as a string. Then your program should display the number of days in that month. Display "28 or 29 days" for February so that leap years are addressed.
6. The following table lists the sound level in decibels for several common noises.

Noise	Decibel level (dB)
Jackhammer	130
Gas lawnmower	106
Alarm clock	70
Quiet room	40

Write a program that reads a sound level in decibels from the user. If the user enters a decibel level that matches one of the noises in the table then your program should display a message containing only that noise. If the user

enters a number of decibels between the noises listed then your program should display a message indicating which noises the level is between. Ensure that your program also generates reasonable output for a value smaller than the quietest noise in the table, and for a value larger than the loudest noise in the table.

7. The following table lists an octave of music notes, beginning with middle C, along with their frequencies.

Note	Frequency (Hz)
C4	261.63
D4	293.66
E4	329.63
F4	349.23
G4	392.00
A4	440.00
B4	493.88

Begin by writing a program that reads the name of a note from the user and displays the note's frequency. Your program should support all of the notes listed previously.

Once you have your program working correctly for the notes listed previously you should add support for all of the notes from C0 to C8. While this could be done by adding many additional cases to your if statement, such a solution is cumbersome, inelegant and unacceptable for the purposes of this exercise. Instead, you should exploit the relationship between notes in adjacent octaves. In particular, the frequency of any note in octave n is half the frequency of the corresponding note in octave $n+1$n+1. By using this relationship, you should be able to add support for the additional notes without adding additional cases to your if statement.
Hint: To complete this exercise you will need to extract individual characters from the two-character note name so that you can work with the letter and the octave number separately. Once you have separated the parts, compute the frequency of the note in the fourth octave using the data in the table above. Then divide the frequency by 2^{4-x}24−x, where xx is the octave number entered by the user. This will halve or double the frequency the correct number of times.

8. In the previous question you converted from note name to frequency. In this question you will write a program that reverses that process. Begin by reading a frequency from the user. If the frequency is within one Hertz of a value listed in the table in the previous question then report the name of the note. Otherwise report that the frequency does not correspond to a known note. In this exercise you only need to consider the notes listed in the table. There is no need to consider notes from other octaves.

9. The wavelength of visible light ranges from 380 to 750 nanometers (nm). While the spectrum is continuous, it is often divided into 6 colors as shown below:

Color	Wavelength (nm)
Violet	380 to less than 450
Blue	450 to less than 495
Green	495 to less than 570
Yellow	570 to less than 590
Orange	590 to less than 620
Red	620 to 750

Write a program that reads a wavelength from the user and reports its color. Display an appropriate error message if the wavelength entered by the user is outside of the visible spectrum.

LIST Exercises

1. When analysing data collected as part of a science experiment it may be desirable to remove the most extreme values before performing other calculations. Write a function that takes a list of values and an non-negative integer, `n`, as its parameters. The function should create a new copy of the list with the `n` largest elements and the `n` smallest elements removed. Then it should return the new copy of the list as the function's only result. The order of the elements in the returned list does not have to match the order of the elements in the original list.

   Write a main program that demonstrates your function. Your function should read a list of numbers from the user and remove the two largest

and two smallest values from it. Display the list with the outliers removed, followed by the original list. Your program should generate an appropriate error message if the user enters less than 4 values.

2. In this exercise you will create a program that identifies all of the words in a string entered by the user. Begin by writing a function that takes a string of text as its only parameter. Your function should return a list of the words in the string with the punctuation marks at the edges of the words removed. The punctuation marks that you must remove include commas, periods, question marks, hyphens, apostrophes, exclamation points, colons, and semicolons. Do not remove punctuation marks that appear in the middle of a words, such as the apostrophes used to form a contraction. For example, if your function is provided with the string `"Examples of contractions include: don't, isn't, and wouldn't."` then your function should return the list `["Examples", "of","contractions","include","don't","isn't","and","wouldn't"]`.

   Write a main program that demonstrates your function. It should read a string from the user and display all of the words in the string with the punctuation marks removed. You will need to import your solution to this exercise when completing Exercise 158. As a result, you should ensure that your main program only runs when your file has not been imported into another program.