# CS 121 Practice Problems

Spring 2016, through March 7

## Simple Programs

1. **Clock arithmetic:** Suppose you keep time with a 24-hour clock. That is, hour 11 refers to 11am, hour 23 refers to 11pm, and hour 0 refers to midnight. If it is currently hour 9, and you set a timer to go off in 52 hours, then the clock will say hour 13 when the alarm goes off.

   Write a Python program that asks the user for the current hour and the number of hours on the timer. Your program should then print the hour on the clock when the alarm goes off.

2. **Wind chill:** If $w$ represents the wind speed in miles per hour, and $t$ represents the temperature in degrees Fahrenheit, then the wind chill is calculated according to the following formula:

   $$35.74 + 0.6215t - 35.75w^{0.16} + 0.4275tw^{0.16}$$

   Write a program that asks the user for the wind speed and temperature, and then prints the wind chill.

   When you run your program, it should look something like this:

   ```
   Enter the wind speed (in miles per hour): 20
   Enter the temperature (in degrees Fahrenheit): 5
   The wind chill is:  -15.435721635148337
   ```

   Here is another sample screenshot of the wind chill program:

   ```
   Enter the wind speed (in miles per hour): 10
   Enter the temperature (in degrees Fahrenheit): -5
   The wind chill is:  -22.131599314136327
   ```

3. **Compound interest:** If a principle $P$ is invested at interest rate $r$, compounded $n$ times per year, then after $t$ years the investment will have grown to amount $A$ given by the following formula:

   $$A = P\left(1 + \frac{r}{n}\right)^{nt}$$

   Write a program that asks the user for the principle, interest rate, the number of compoundings per year, and number of years. Your program should then output the amount to which the investment will grow.

## Practice with Python Modules

1. Print three random floating-point numbers between 0 and 1, and also print the sum of the three numbers.
2. Choose a random number between 20 and 200, and draw a hexagon with side lengths equal to that number of pixels.
3. The following fractions approximate pi:

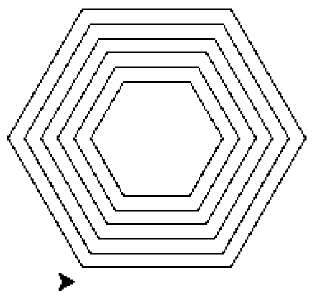
$$\frac{22}{7}, \frac{245}{78}, \frac{333}{106}, \frac{355}{113}, \frac{103993}{33102}.$$

   To how many decimal places is each approximation accurate? Compare each approximation to the value given by math.pi to find out.
4. Write a program that asks the user for the radius of a circle, and then prints the circumference and area of the circle. Use math.pi.

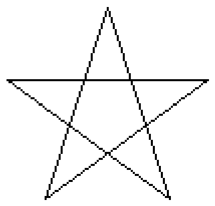
---

## Practice with Functions

1. Write a fruitful function `areaOfRectangle(height, width)` that returns the area of a rectangle with dimensions `height` and `width`.
2. Write a function `drawPolygon(anyTurtle, numSides, sideLength)` that makes a turtle draw a regular polygon with the speficied number of sides and side length. Use your function to draw four different polygons on the screen.
3. Write a function `drawHexagon(anyTurtle, sideLength)` that calls your `drawPolygon` function from the previous question to have a turtle draw a regular hexagon. Use your function to draw six nested hexagons on the screen, like this:



4. Write a function `drawStar(anyTurtle, sideLength)` that makes a turtle draw a five-pointed star, like the one shown below. Then use your function to draw five stars on the screen, in different positions.



## PRACTICE WITH SELECTION

1. Write a program that prompts the user to enter a decimal number. If the number is between 0 and 1 (inclusive), then the program prints the number as a percent. Otherwise, the program prints the message "You entered a number that is not between 0 and 1."
2. Write a program that prompts the user to enter 1 or 2. If the user enters 1, then the program draws a square. If the user enters 2, then the program draws a triangle.
3. Write a program that prompts the user to enter three numbers and prints the largest of the three numbers.
4. Write a program that asks the user to enter a day of the week, such as "Monday". If what the user enters is, in fact, the name of a weekday, then print "Thank you." However, if the user enters something that is not a day of the week, then print an error message.
5. Modify your previous program as follows. If the user enters a valid day name, then tell the user whether *today* is the day of the week that they entered. Furthermore, use the datetime module to get the current day of the week as follows:

   ```
   import datetime
   dayOfWeek = datetime.date.today().strftime("%A")
   ```

---

## PRACTICE WITH LOOPS

1. Write a program that will figure out how many terms in the sum $1 + 2 + 3 + \cdots$ are necessary for the sum to exceed one million.
2. Write a program that prompts the user for an integer between 1 and 10. However, if the user enters something that is *not* an integer between 1 and 10, your program should print an error message and prompt the user to try again. Your program should repeat this as many times as necessary, until the user enters an integer between 1 and 10.
3. Use a loop to investigate the following sum:

   $$1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \cdots$$

   What happens when you add up many terms of this sum? How many terms are necessary to obtain a sum greater than 1.99? How many terms are necessary to exceed 1.999999? Do you think the sum will ever exceed 2?
4. Write a program that asks the user for an integer greater than 1. Your program should then find and print the largest factor of that number. (Recall that a factor divides the given number with remainder zero. For example, 6 is a factor of 24, but 7 is not a factor of 24.)

## Practice with Loops: Image Processing

1. Write a program that converts an image to black and white. To do this, iterate over all pixels in the original image. For each pixel, if the sum of the red, green, and blue values is less than 383, then assign black (0, 0, 0) for to the corresponding pixel in the new image; otherwise assign white (255, 255, 255) for that pixel.
2. Write a program that resizes an image, creating a new image half as big (in length and width) as an original image. A simple way to do this is to read the colors of the pixels in every other row and column of the original image, and assign those colors to corresponding pixels in a new image.
3. A simple way to blur an image is to set the color values of each pixel to the average color values of the neighboring pixels. Write a program that does this.

---

## Practice with Strings

1. As we will see in a few weeks, web pages are built with HTML strings such as "`<b>Yay</b>`", which prints **Yay** in bold text. The letter *b*, appearing between angle brackets, is called a *tag*. Tags come in pairs, and the second tag in the pair must have a slash before the tag text.

   Write a function called `makeTags` that accepts a tag word and some text, and returns an HTML string with the text surrounded by properly-formatted tags. Your function declaration should be:

   ```
   def makeTags(tag, text):
   ```

   A call to `makeTags("div", "some text")` should return the string "`<div>some text</div>`".
2. Now write a function that extracts text from inside of HTML tags. Your function declaration should be:

   ```
   def extractText(HTMLstring):
   ```

   A call to `extractText("<i>some text</i>")` should return "`some text`".

   *Hint*: consider the `find` and `rfind` Python string methods.
3. Write a program that asks the user to type in some text, and then reports the number of characters and also the number of vowels that the user entered.
4. Write a function that removes all non-alphabetic characters from a string. That is, your function should accept a string of text, and then return a string containing the same text but with all non-alphabetic characters removed.