This assignment was originally developed by Casey Fiesler at University of Colorado Boulder for a non-coding "computing and society" class with students who largely had little prior computing experience. It is adapted from a similar exercise in an intro programming class to teach coding concepts; this assignment in contrast teaches algorithmic thinking without code. Please feel free to use and adapt this assignment however you like. You can also email casey.fiesler@colorado.edu with any questions.

For more assignments and resources: www.internetruleslab.com/responsible-computing

Assignment: College Admissions Algorithms

Learning objectives:
- Understand what algorithms are, how they might be used
- Understand how algorithms can reflect bias
- Create a simple algorithm
- Reflect on that algorithm's implications

As you know, the college admissions process involves a lot of types of data from prospective students to make decisions. With the number of applicants increasing, colleges may begin relying on algorithms to select which applications should receive more intensive human review. An algorithm could use quantitative data – such as GPA and SAT score – to provide initial recommendations. In fact, there is more data available than ever. Many colleges even track data about prospective student engagement – e.g., whether they open emails, visit the college website, engage on social media, etc. This creates a "demonstrated interest" value.

Based on a recent survey of college admissions officers, we know some of the weights that humans tend to give to these different types of data.

Your task will be to describe an algorithm that provides a recommendation for which prospective students are likely to be the best candidates for admission.

1: How should this algorithm work?

In plain language, write out how you think an algorithm based on student applications should work. e.g., "It should find the students with the highest grades but also take into account what high school they went to…"

2: How much weight should be given to each factor, what should the cut-off score be, and what does this algorithm look like?

To make things simpler for this algorithm, we have turned this prospective student data into a set of factors, all with a value on a **0 to 10 scale**, with 0 being the worst and 10 being the best.

- **Test Scores:** standardized tests such as SAT and ACT
- **Grades:** grade point average with more weight for advanced placement courses

- **Strength of Curriculum**: how difficult the curriculum at their high school is (or, how "good" of a high school they went to). Note that this factor is often correlated with socio-economic status (e.g., students whose families have a lot of money are more likely to go to very "good" schools).
- **Demonstrated Interest**: based on online engagement data (see description above)

An algorithm like this would actually be attaching weights to different factors. Let's create an algorithm that does that.

Decide on a weight for each of the four factors, so that the weights add up to 100. Basically, this means how much each of these factors should matter to the algorithm's decision. For example, if test scores are VERY important, give it a weight of 50%, or if demonstrated interest is not important, give it a weight of 10%. The total of the 4 weights should add up to 100.

Test scores: ____
Grades: ____
Strength of curriculum: ____
Demonstrated interest: ____

All of these factors are on a 0 to 10 scale with 0 being the worst and 10 being the best. If you averaged all of these together, the very highest overall score you could get would be 10!

Choose a number between 0 and 10 that you think should be the cut-off to further consider a student for admission:

Cut-off score: ____

Here is an example of the type of admissions algorithm you might have just created, if you decided that 5 should be the cut-off. Remember that to take 20% of something, you multiply it by 0.2. This algorithm has test scores as 20%, grades as 40%, strength as 30%, and interest as 10%.

```
Consider this student IF (SAT x 0.2 + GPA x 0.4 + Strength x 0.3 +
Interest x 0.1) > 5
```

Fill in the blanks for your own admissions algorithm, based on your answers above:

IF (SAT x ____ + GPA x ____ + Strength x ____ + Interest x ____) > _____

3: Apply your algorithm to three examples

Create THREE hypothetical candidates for admission, make up data for all of them (with each on a 0 to 10 scale), and then calculate whether or not they would be further considered for admission by your algorithm.

For example, let's say that Casey has a Test Score value of 5, a Grades value of 8, a Strength value of 10, and an Interest value of 2.

Using the example algorithm above, the calculation would look like this:

```
(5 x 0.2 + 8 x 0.4 + 10 x 0.3 + 2 x 0.1) = 7.4

7.4 is > than 5, so this student would be considered for
admission!
```

(Note that we are not testing your math skills, so please show your work as above - and if you have trouble with this part, please talk to your TA!)

Candidate 1:

Test Scores: ____
Grades: ____
Strength: ____
Interest: ____

Calculation:

Decision: ____ (Consider or Not Consider)

Candidate 2:

Test Scores: ____
Grades: ____
Strength: ____
Interest: ____

Calculation:

Decision: ____

Candidate 3:

Test Scores: ____
Grades: ____
Strength: ____
Interest: ____

Calculation:

Decision: ____

4: Reflect on the outcomes

In 100 to 200 words, reflect on the outcomes of this algorithm. What are some examples of ways this algorithm might give undesirable results for certain types of prospective students?

5: How should and shouldn't algorithms be used in college admissions? Why?

Given the realities of time and resources (i.e., only so many person hours can be involved in the admissions process), how *should* an algorithm like this be used in the admissions process (assuming an algorithm that is far more sophisticated than the one you just created!)? How *shouldn't* it be used? Explain your thinking, taking into account possible undesirable outcomes.