## Brainstorming `

* We’re assuming that these are software developers who already have some understanding of the social issues and care about them
* What ideas can we get from this audience for this curriculum? What do software developers know that it might be important for other people to learn? How do they approach problems that might be useful to model/explain?
  + Threat modeling
    - Think about how systems might be broken or used for purposes other than their original intent
    - What does this mean for algorithms people encounter in daily life? What might they be able to do other than what people think they’re doing, with the information they have?
    - What information do they need to do their jobs? Where might they be getting it?
  + They understand the value of datasets
    - How much your algorithmic outputs depend on your data inputs (e.g. your computer vision system can’t see black people if the training data was all white people)
    - The fact that data sets may be chosen for convenience rather than representative-ness or sociological utility
      * e.g. Google testing its social software on employees (predominantly wealthy, white or Asian, interacting in a work context) and thus being repeatedly blindsided by real-world uses of social software or dangers that their features expose users to (e.g. automatically friending domestic violence survivors with their abusers because they happen to have emailed each other)
      * Microsoft testing Kinect on employees (thus ending up with a computer vision system that didn’t do a great job on black people)
    - computers can be pretty good at telling you the way things *are*, within the limits of a particular data set, but that’s very different from the way things *should be*
      * e.g. google image search for “CEO” or autocomplete for “women are…” or “black girls are…” or any other group
  + They know potential technical solutions to the problems today
  + Recommendation algorithms are based on behavioral data (“people who bought X also bought Y”) and not on expert recommendations, the way things “should” be, etc.
  + Math
    - The things you can and can’t do with systems depend on the equations
    - You can supply inputs that have outputs, but those outputs may not be meaningful or intellectually honest depending on how the algorithm was constructed
    - Problem: lots of people don’t have the math background and/or are mathphobic - how do you translate this expert knowledge to a general audience?
  + Computers don’t see data the way people see data
    - You can easily convince a neural net that a picture of a toaster is a picture of a kitten by tweaking the right pixels - humans would never fall for this trick
    - The types of errors that algorithms make are not the types of errors humans make and can in fact be profoundly alien
    - That means that when the computer is wrong, it can be wrong in ways you would never even imagine
  + Computers are horrifyingly insecure
    - But also the quickest way to hack them is often the humans, not the machines
  + Personalization means that different people can see completely different internets
    - My parents’ internet is a visual nightmare of ads and attention disruptors; mine is not (thanks, adblock!)
    - Ad personalization may happen in ways that actually violate the law
      * Latanya Sweeney’s work on financial advertisement that ends up falling strongly along racial lines
      * Facebook letting you include/exclude groups in your ads in ways that violate laws against racism, ageism, religious discrimination, etc. -- Pro Publica has done good work on this
  + Algorithms are just things made by humans
    - They may be executed by machines, but they’re designed and implemented by humans
    - They can therefore be very efficient ways to multiply human blind spots and biases
* Software engineers may be good sources of concrete, real-world examples for illustrating these questions
* Software or software-adjacent people who’ve written on this topic and may thereby be good sources of concrete examples:
  + Safiya Noble
  + Virginia Eubanks
  + Cathy O’Neil
  + Latanya Sweeney
  + Nate Matias (e.g. <https://freedom-to-tinker.com/2018/07/03/teaching-the-craft-ethics-and-politics-of-field-experiments/> )
* software developers might be able to create interfaces which are pedagogically useful
* there’s some interesting work on folk models of how computational things work
  + this is often quite different from how they actually work
  + but that’s a useful basis for a discussion or other instructioonal exercise
  + e.g. <http://www.rickwash.com/papers/rwash-homesec-soups10-final.pdf> , <https://dl.acm.org/citation.cfm?id=2998316>

## Logic Model Template

**Logic Model for Algorithmic Awareness**

***Audience: Software Developers***

**Overall Project Goal: To engage software developers in creating resources and “train the trainer” opportunities for Algorithmic Awareness**

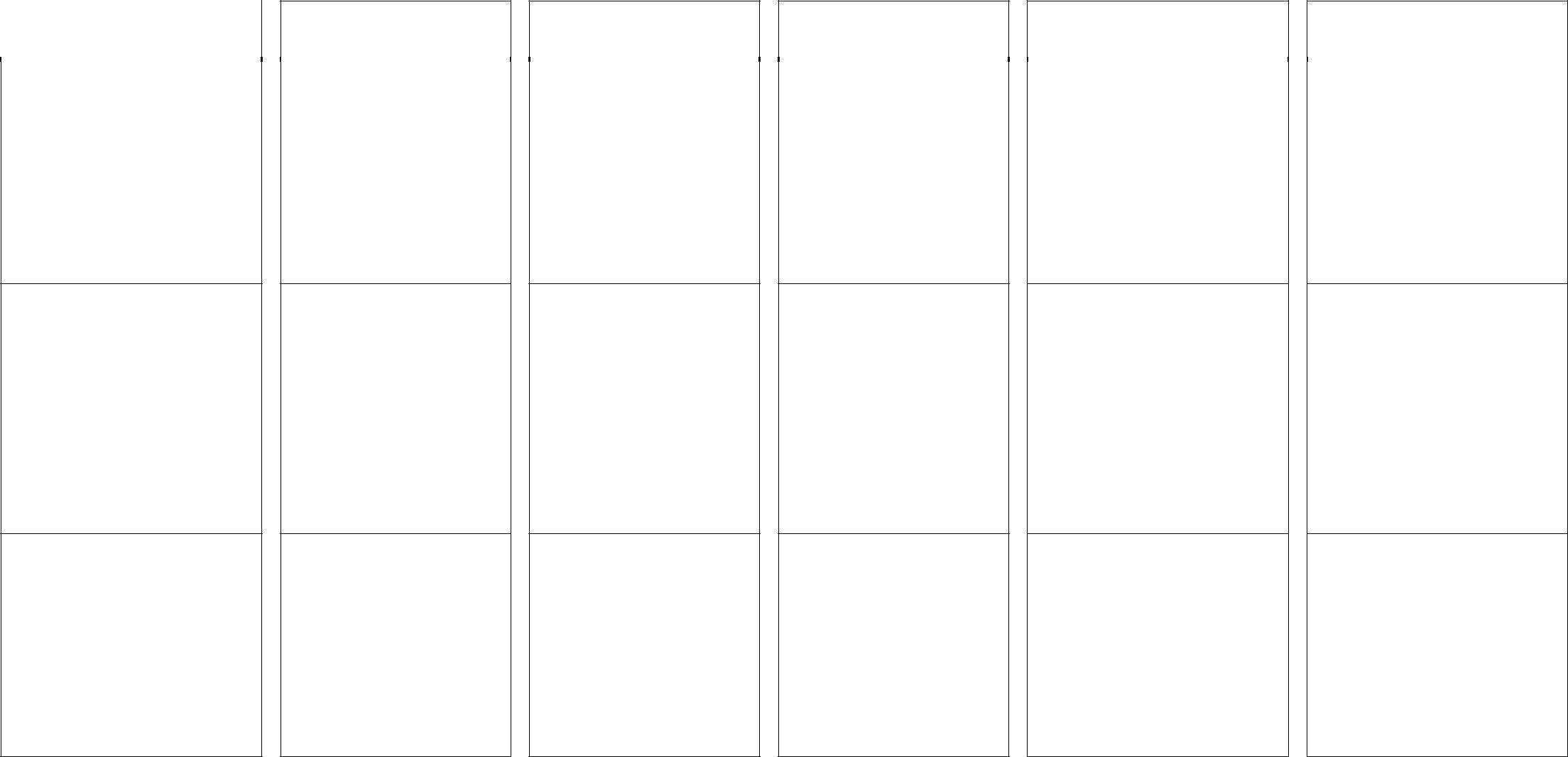
|  |  |  |  |
| --- | --- | --- | --- |
| **Objectives** (what your audience will be able to do/say/feel, etc. after the project) | **Resources**  (what you have or need to do your project) | **Activities**  (what you do to make project happen)  software dev does // project does | **Outputs**  (what you produce or deliver as part of your project) |
| Audience will show they know more about the project by... |  | Volunteering their time // Have an avenue for developers to volunteer |  |
| Audience will demonstrate that they care about their teaching/learning objectives by... |  | Consulting, providing examples, helping others on the project use github // Setup meetings / calls for input from developers |  |
| Audience will state their intent to engage in these stewardship actions... |  | Maintaining the more technical elements of the resources // Add a governance structure for maintainability |  |

**Logic Model Template**

***Audience:***



**Overall Goal:**



|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Objectives** |  | | **Resources** |  | **Activities** |  | **Outputs** |  | **Short-term Outcomes** |  | **Mid- to Long-term** |
|  |  |  |  |  |  |  |  |  |  |  | **Outcomes/Impacts** |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | • |  |  | • |  | • |  | • |  | • |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| • | • | • | • | • |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| • | • | • | • | • |