# **Project 1:** Developers as Decision Makers

*A modified version* [*of the assignment created*](https://ethicalcs.github.io/#decision-makers) *by Evan Peck at Bucknell University*



*In this lab, we will create algorithms that determine housing for SFSU students.*

When we say the word ‘algorithm’, we tend to ascribe agency to the computer. It is *deciding* things for us. But the reality is that there is no magic. There is software developers like you and me who design and create sets of rules that the computer carries out for us.

**These algorithms are all around us, and they are *constantly* making decisions.**

The decisions we make in code impact the lives of real people. For example, the [**Silicon Valley Triage Tool**](https://economicrt.org/publication/silicon-valley-triage-tool/) is an *algorithm* that identifies homeless people for whom giving them housing would cost the public less than keeping them homeless. So even as we learn the simple structures of code, we need to think about *how we can make* ***good*** *decisions.*

We are going to explore this idea in a more familiar context to you - **university housing allocation**. At SFSU, on-campus housing is in high demand but very limited. This has resulted in the university relying on a first-come first-serve policy: <https://housing.sfsu.edu/apply>. You might not think of it as one, but this method is **an algorithm.**

In this project, you will have the opportunity to design your *own* algorithm. We’re also going to begin dabbling with a [human-centered design process](http://www.designkit.org/human-centered-design) to make sure that **the decisions we make are never untethered from the people we impact.**

### In this project, you’ll practice...

* translating English rulesets into code,
* soliciting text input from people,
* **\*Applying conditionals (if, else if, else) to make decisions with a program\*,**
* using an accumulation variable to keep track of information in a program, and
* integrating basic human-centered design processes into your programs.

## First: Is the first-come first-serve fair? Can we do better?

[R1] You must start by thinking if the existing way of things doing is fair. Is the first-come first-serve approach to housing allocation fair? Notice how you must think about what “fairness” involves.

First come first serve housing is not entirely fair. Sure, students who submit the application may deserve more consideration, but this disadvantages students who may be experiencing great adversity. Factors like internet connection consistency, unsupportive family, documentation/citizenship status may affect one's ability to complete the housing application, and these are likely the people who need the housing more to begin with. For housing to be considered need-based you need to consider the needs of those it is serving and making the resources first-come first-serve would be ignoring those needs.

[R2] Who are the actors that may be at a greater disadvantage with the current system?

I think the main actors that are disadvantaged by the first-come-first-serve algorithm are people that are in greater need of housing. People who typically have higher quality of life will likely have no problem completing the application before others, but they are also likely less in need of housing. So, the first-come-first-serve algorithm would likely be unproportionally benefiting people who need the housing less, and putting people who need the housing more at a disadvantage.

[R3] Do you think there is a better way to allocate housing such that it is fairer? What ethical framework(s) guide your evaluation?

I think that a more fair way of allocating housing is actually considering what factors may cause a need for on-campus housing. Students who are international cannot commute to the campus daily and may likely struggle with finding a place to live on their own. Therefore distance from school is a high-risk factor that should be considered when offering need-based housing. Age is also a good factor to look at in combination with factors like need for financial aid or family income. Younger students are much more likely to struggle with finding housing if they were to not be offered on-campus housing. And it makes much more sense to push a 26 year old, for example, to find their own place off campus as opposed to asking a new 18 year old to find their own place off campus. Financial aid and family income is good in combination with the age factor as well because high financial aid usage/need may indicate a greater need regardless of age, and high family income may indicate lack of need in younger students who may actually be able to find their own place due to their family having the financial resources.

## The Decision-Maker: Who gets to choose their housing first?

**Your job is to build an algorithm that helps determine the order in which students will get to select their housing.** To simplify things, we’re going to use a *point system.*

* Students are awarded several points based on a variety of factors.
* Students with the most points get first choice at housing.

**This real approach** is used by many universities. For example, consider the following *real* *point system* used by another college in the United States:

* *Current Freshman:* 1 point
* *Current Sophomore:* 2 points
* *Current Junior:* 3 points
* *Current Senior:*4 points
* *23+ Years of Age:* 1 point
* *Full-Time, Off-Campus Program Credit (e.g., student teaching):* 1 point
* *Academic Probation:* -1 point
* *Possible Academic Suspension:* -2 points
* *On Disciplinary Probation at Any Point during the Academic Year:* -3 points

So, a junior (+3 points) who is 23 years old (+1 point) would have priority over a senior (+4 points) who is on academic probation (-1 point).

**Overall goal:** Create a program that assigns points to students to prioritize them in housing selection.

## [R4] Before you Code, Assess the Needs of Your Users

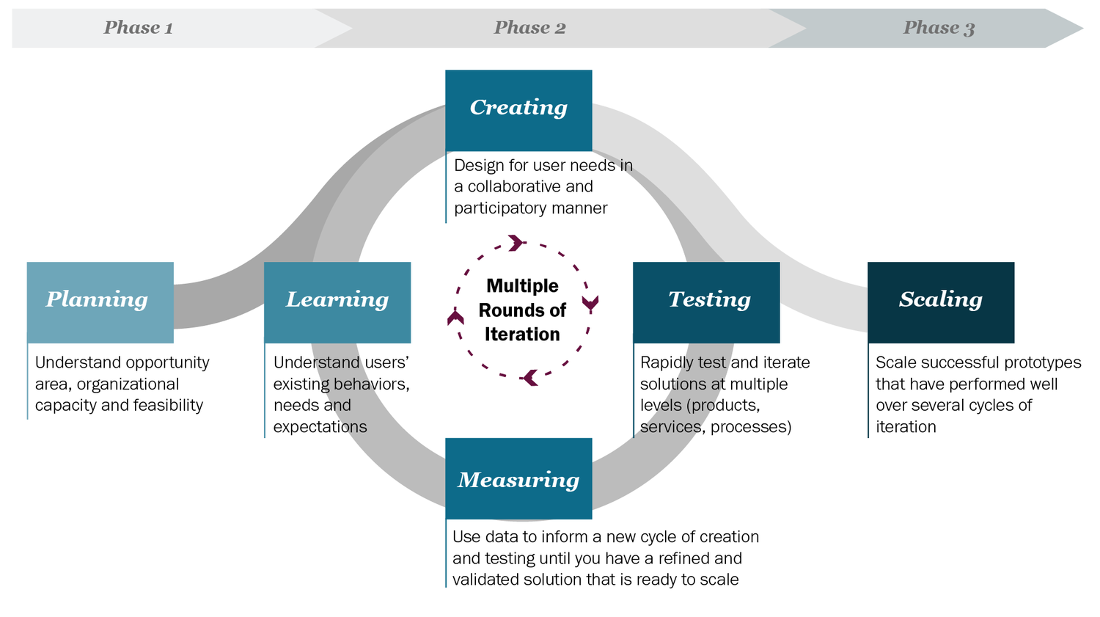
While the list above was *one* college’s take, there are **many** more potential aspects to consider if you want to create a **fair** algorithm that considers the diverse needs of students.

**You should not create a program that serves people without talking to people.** Talk to other students in your class. Ask them about their needs. What other unique factors may be important in deciding who should choose housing first?

**Group Work:** Form a group of 3—4 students. Discuss the various factors to consider when deciding on housing allocation. Fill in the bullet-point list below with the factors that came up in the conversation. Give each one a numeric priority (-1 to 5 points).

* Factor 1: Distance student lives from school 5 (Most important)
* Factor 2: Age of student 4
* Factor 3: Year of student 3
* Factor 4: Academic standing 2
* Factor 5: Extra curriculars on campus -1 (Least important)
* Factor 6: Previous housing record 1
* ….

Notice that this is the learning phase of the **human-centered design process** shown in the following figure.



Talking to users, prototyping ideas, and then testing them with users, is part of what is known as the **human-centered design process**… a key process for developing useful and usable programs.

## Assessing the Needs of Our Users

## So far, you have discussed the factors amongst yourself. It is not enough; other people may think differently about what is fair.

## Using the list that was prepared in your group, interview other people. Try to interview people outside the CSC 101 class. These can be family, friends, or classmates from a different class. Ask what factors are important to them. Show the factors that you had identified from group work and ask them about their priority.

### **Individual Work:**

## [R5] Results of the Interviews and Discussions: Factors that Matter.

I interviewed 3 SFSU students about their factors (full score can be earned when interviewing 3 or more different people).

**MODIFY/ADD TO THE LIST BELOW WITH OBSERVATIONS FROM YOUR INTERVIEWS.** You can add or delete factors. Provide a number with the factors to indicate which factors that you think are the most important. For example, if you think class year is the most important, place a “1” by that factor. If you think two or more factors are equally important, place the same number by those factors.You can use our factors from the group work to start with.

* Distance student lives from school 1
* Age of student 3
* Year of student 3
* Academic standing 4
* Extra curriculars on campus 5
* Previous housing record 4
* Income/financial stability of the student 2

## [R6] Design and Plan Your Point-Assigning Algorithm

Now it is time to translate our student needs into a concrete algorithm. Be careful and limit yourself to the most important factors. *You have a limited amount of time in this class!*

Our overall algorithm will be the following:

1. Ask students questions (like *What class year are you?*).
2. Assign points based on their answers (like *4 points for seniors*).
3. Accumulate their total points across all answers (like *You have 23 housing points*).

**Your goal:** Create a bullet-point list that describes the factors you are considering, and how you are mapping those factors to point values (positive or negative). Please do your best to correctly fill out the list. When coding starts, this will serve as an outline for writing your prompts and how you will handle responses.

## Writing Our Algorithm in English

**(EDIT OR REPLACE THE FOLLOWING LIST TO REFLECT *YOUR* HOUSING PRIORITIES.)**

For our algorithm, we are going to prioritize distance the student lives from school, financial situation of the student, and then age.

Distance student lives from school. The further students are considered much more eligible.

* In county: +0 point
* Out of county: +3 points
* Out of state: +6 points
* Out of country: +9 points

Income/financial stability of the student. Students who are more financially able are less eligible for housing.

* No financial aid requested or used: +0 points
* Financial aid requested but max not used: +2 points
* Financial aid requested and max used: +4 points

Age of student. Students that are the age where college attendance is expected they are more eligible for housing. If student is above that age they are more likely to be able to support themselves and therefore are less eligible.

* Between 17-24: +5 points
* Else: +0 points

Year of student combined with status of expected college age. Students in their early years at the college are more eligible for housing because they are likely to be less established in the city the college is located in.

* First year: +6 points
* Second year: +4 points
* Third year: +2 points
* Fourth year: +0 points

Academic disciplinary standing. Students with better academic/disciplinary standing are more eligible for housing.

* Good standing: +3 points
* Poor standing: -5 points

Extra curriculars on campus. Students engaged with extra curriculars on-campus are more eligible for on-campus housing.

* Engaged in extra curriculars: +2 points
* Not engaged in extra curriculars: +0 points

Previous housing record

* Good housing record: +3 points
* Poor housing record: -3 points

## [R7] Before you Code, Make Sure it Works: User Testing

**How will you know if your program serves people well?** At a minimum, you should have tests considering different groups of people who may be affected by your program.

* Were the results what you expect?
* Did you discover any cases which you haven’t accounted for previously?

**Your goal:** Write **at least** 5 hypothetical *test* cases for your program. You may need to modify the given tests if they do not mention any of your factors. Five test cases must be listed in order to earn a full score for R7.

### Test Cases

**EDIT THE FOLLOWING LIST TO REFLECT *YOUR* PROPOSED POINT VALUES.** You are required to add at least 2 test cases of your own.Make sure to modify the points and/or factors in the already given tests so that they match your factors.

* A 25-year-old senior; On academic probation; From the county; Fin. Aid requested but not all used; Engaged in on campus extra curriculars; Clean previous housing record; should output 2 points.
* A 22-year-old junior who is student teaching; good academic standing; no fin. Aid requested; clean previous housing record; and lives in the county; should output 15 points.
* A 20-year-old sophomore on disciplinary probation; from the county; clean previous housing record; fin. aid requested and max used; no extra curriculars; should output 11 points.
* A 19 year old freshman international student has clean disciplinary/academic record and proof of clean housing record from home country; Student has requested and used the max amount of financial aid available; no extra curriculars planned; should output 30 points.
* An 18 year old freshman from out of state has poor academic/disciplinary standing, proof of clean housing record, and has requested financial aid. Student plays softball. Should output 21 points.

## [R8] Write Code that Automates Your Decision-Making Process

**Now it’s time to translate your algorithm into code.**

**Your goal:** Implement *the algorithm you designed* in IntelliJ.

* **Step 1:** Follow the instructions posted on Canvas to create a new project for this assignment.
* **Step 2:** Implement your algorithm inside the main method.
* **Step 3:** Is it correct? You should check your code with the test cases you outlined above. You should run your program using the test cases as an outline for inputs. Your program, if implemented correctly and according to your algorithm, should produce the expected points value stated in each test case.
* **Step 4:** Submit a zip file with the Java file and this completed document.

During your creation, keep a couple of things in mind:

* Use comments to describe what was happening in the program.
* Choose variable names that clearly describe the data that they hold.
* Use spacing to group similar code.

## [R9] Is Your Implementation Fair?

Your Code Works… but is it fair? Now Reflect on the Tradeoffs and Conflicts

**You should never deploy real code without checking your assumptions.**

Your test cases tested your *technical* assumptions, but not *your social* assumptions.

1. Find classmates either inside or outside of the lab.
2. Run your code with them.
3. Get feedback on what worked and what didn’t.

### **Individual Work: Add your reflections to this document below.**

1. Which students are most likely to benefit from your algorithm?
   1. Students that either live far from campus or who are the expected college age (18-24 counting one gap year and students who were always a year older than their peers) benefit greatly from this algorithm. Students who are engaged on campus, and are a good community member and classmate also benefit greatly.
2. Which students are likely to be forgotten by your algorithm?
   1. Students who either are not good community members/classmates, or those demonstrate less need possibly because they live in the same county as the school or are not on financial aid.

*If you are interested…*

* **Optional reading:** [*What Happens When An Algorithm Cuts Your Health Care*](https://www.theverge.com/2018/3/21/17144260/healthcare-medicaid-algorithm-arkansas-cerebral-palsy) *By Colin Lecher*
* [*We created poverty. Algorithms won’t make that go away.*](https://www.theguardian.com/commentisfree/2018/may/13/we-created-poverty-algorithms-wont-make-that-go-away) *By Virginia Eubanks*