Fair and Stable Matchings

A Computational Thinking Module

Paul Kehle
kehle@hws.edu

Tami Carpenter

Draft Version
February 28, 2015

Computational Thinking

Computational thinking is a way of looking at and thinking about the world in terms of how information and data can be defined, generated, collected, related, analyzed, represented, and communicated using computer and other information technologies. It is a problem-solving approach informed by an awareness of computer technology's capabilities.

VCTAL

The Value of Computational Thinking Across Levels (VCTAL) project is a curriculum development project carried out collaboratively by Rutgers University's Center for Discrete Mathematics and Computer Science (DIMACS) and the Consortium for Mathematics and Its Applications (COMAP) with assistance from Hobart and William Smith Colleges, and Colorado State University. VCTAL is funded by the National Science Foundation.

Over four years, VCTAL will develop a collection of one-week modules that introduce students to computational thinking in a wide variety of contexts and disciplines. These modules will be field-tested by high school teachers, revised, reviewed, published, and made available in a variety of formats. The modules will be suited both to stand-alone use as an enhancement to an existing class, and to use as a complete curriculum for a half-year course in computational thinking.

Overarching Goal

The project's overall goal is to provide curricular resources to help high school teachers develop in their students an appreciation of how computational approaches can be of use in virtually any setting or discipline. Acquiring this appreciation does not mean that the student must continue with computer science courses. Although some students might be motivated to pursue courses in computer science or information science, in fact, our goal will be achieved whenever a student understands that computational approaches to a particular topic or problem probably exist and that collaborating with a computer scientist might be wise.

Module Overview

This module introduces students to matching problems in general and then focuses on a particular type of matching problem that involves people's mutual preferences. Students learn an algorithm for finding stable matchings and then discuss how to decide which stable matching is fairest. Finally, students transfer what they've learned to different matching-problem case studies. The module concludes with a mini-symposium at which the students present their case studies.

Prerequisites

No computer programming experience is required or involved. Students should understand and be able to calculate means, medians, and modes for a set of data.

Materials & Resources

Copies of handouts and printed materials for activities.

Suggested Uses

Although this module can be used in many subjects and in grades 9 through 12, it is perhaps best suited to 10th or 11th grade mathematics, business, or social-studies courses.

Note on Contemporary Relevance

The 2012 Nobel Prize in economics was awarded for work done on stable matching problems. The article at the link below is an accessible summary for popular audiences.

http://www.nobelprize.org/nobel_prizes/economics/laureates/2012/popular-economicsciences2012.pdf

References

C. Cheng, Understanding the Generalized Median Stable Matchings, *Algorithmica 58:1* (2010) pp. 34-51.

C. Cheng and A. Lin, Stable Roommates Matchings, Mirror Posets, Median Graphs, and the Local/Global Median Phenomenon in Stable Matchings, *SIAM Journal on Discrete Mathematics 25:1* (2011) pp. 72-94.

C. Cheng, E. McDermid, I. Suzuki, A Unified Approach to Finding Good Stable Matchings in the Hospitals/Residents Setting. *Theoretical Computer Science 400:1-3* (2008) pp. 84-99.

D. Gale and L. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, *69*, 1962.

D. Gusfield and R. Irving. *The Stable Marriage Problem: Structure and Algorithms*. The MIT Press, 1989.

Module Structure: Units, Suggested Timeline, Materials, and Objectives

The suggested times (days) below refer to the number of 45-minute class periods. These times are intended as rough guides and will need to be adapted to the length and format of your classes and to the levels of your students. The time needed to teach particular units will also depend on how you choose to use the activities in any given unit as some can be used equally effectively as homework or as in-class work.

Unit/Time	*Topic*	*Materials*
1 1 day	• Matching Problems	• Handout of Matching Figures • Handout on Lists and Matrices • Homework: Finding a Good Match
2 1 days	• Stable Matchings	• Big Sibling/Little Sibling Case Study • Homework: Algorithm Design
3 2 days	• Student Algorithms • Gale Shapley Algorithm	• Gale Shapley Algorithm Handout • Homework: G-S Practice
4 2 days	• Measures of Fairness	• Fairest of All Case Study Handout

5
out of class HW + 1 day
	• Assessment: Applying what students have learned.	• Final Case Study Handouts
		  ○ Matching Residents and Hospitals
		  ○ Matching Married Residents and Hospitals
		  ○ Matching Roommates

Unit 1 Objective: Students will be able to describe a variety of different situations where various kinds of matching problems arise.

Unit 2 Objective: Students will be able to explain what an instability is and what a stable matching is.

Unit 3 Objective: Students will be able to use the Gale-Shapley algorithm and explain what it can reveal about a stable-matching problem.

Unit 4 Objective: Students will be able to explain and apply various measures of fairness to stable-matching problems that have more than one solution, and they will be able to argue for which matching is best.

Unit 5 Objective: Students will be able to adapt their skills to new problems.

Glossary

algorithm	An explicit repeatable process or set of rules for solving a problem.
computational thinking	A way of looking at and thinking about the world in terms of how information and data can be defined, generated, collected, related, analyzed, represented, and shared using computer and other information technologies.
default	The way something is done in the absence of other specifications being provided.
instance	An instance is a particular example of a type or category of problems.
matching	Typically, a matching is a set of paired assignments between two distinct groups of individuals or things; however, in some contexts the matches are made within one group and sometimes more than two items might be involved in the assignments making up the matching.
parameter	A value that can be adjusted to suit a particular purpose in a program or process. In most web browsers, the user can decide after how many days the web history of the browser will be cleared. This value, can be changed depending on the needs of the user; the value is a parameter associated with how the web browser functions.
partial order	An transitive ordering of a set that does not require all pairs of elements to be ordered; there can be "ties" in which some pairs are not orderable according to the rule used to order the set. A partial ordering is abbreviated poset, for partially ordered set.
transitive	A relationship is transitive if when a relation holds between A and B and between B and C, then the relation holds between A and C; for example, "less than" is a transitive relationship: if A is less than B, and if B is less than C, then A is less than C.

Unit 1: Make Me a Match

Objective: Students will be able to describe different kinds of matching problems and explain some of the considerations that must be taken into account when formulating and solving them.

Teacher Notes: Class discussion of Make Me a Match!

Begin this module and the first unit with a discussion about making matches.
Pose the question: What are examples of things that need to be matched up?
As students give examples, write them on the board for later discussion. If students have trouble, guide them toward some of some of the examples below or share the examples directly to assure that the list includes a variety of situations in which matching is used.

1. Pairing up socks in the laundry
2. Matching people in a dating service or to attend the prom together
3. Creating doubles partners for a tennis team – mixed doubles; men's doubles; women's doubles
4. Creating the initial brackets in a tournament
5. Matching transplant organs to recipients
6. Pairing big brothers/big sisters with little brothers/little sisters
7. Pairing up people for the 3-legged race on Field Day
8. Matching up college roommates
9. Assigning tasks or jobs to people
10. Assigning people to rooms in a hotel
11. Assigning teachers to classes to teach

Once the class has a good list of different situations in which we need to match things, have the students begin to think about some of the features of particular matching situations. Use some of the items on the above list to spur discussion of two questions:

- *Are you matching items from a single group or are you matching items that come from two different sets?* A pile of socks to be paired up is an example in which you need to match items drawn from a single group. (Note this is not true of shoes which have distinct left and right.) Matching transplant organs to recipients is an example in which you have two distinct sets of items: people and organs. Going through the list above, a (heterosexual) dating service, mixed doubles tennis, big brother/big sister programs, hotel room assignments, task assignments, and assigning classes to teachers are all examples in which you are matching items from different sets. The two sets can be: men and women (in mixed doubles and dating); older and younger people (in big brother and big sister programs); people and organs; people and hotel rooms; or people and tasks/jobs. Other examples where the items to be matched are all of the same type are in assigning roommates, men's or women's tennis doubles, 3-legged race partners, or pairing teams in a tournament.

- *What criteria can you use to determine whether a potential match is a good match?* In some cases this is easy: socks either match or they don't. In other cases, there are some well-established traditions. For instance, the pairings for a tournament usually start by matching the highest ranked player with the lowest, the second highest with the second lowest, and so on. One way to view this is as rewarding past performance of the highly ranked teams. Another way to view it is that, if the seeding is done properly and the favorites advance, the pairings in the next round will again match the highest ranked player with the lowest remaining player, the second highest with the second lowest, and so on. This continues until the final round, which would pair the two most highly ranked teams/players. The idea behind this is to stack the deck for the final game to be as exciting and meaningful as possible or to be able to see the favorites play as often as possible.

Now consider the case where you have 3 jobs and 3 people, and you need to assign people to do the jobs. What are some things that you might want to consider? What criteria might you use to decide whether one assignment of people to jobs is better than another? Here are some ideas that students might suggest:

- It may be the case that all people cannot perform all jobs, so you want to ***match people with jobs they are able to do***. If your people include a chef, a pilot, a doctor, a florist, and a professor and your tasks are to bake a cake, fly a plane, remove an appendix, create a floral centerpiece, and teach a class the best choice is obvious.
- Even when all of the people can do all of the jobs, some people may be able to do some jobs better/faster than others, so you may want to ***assign people to the jobs they are most efficient in completing***. Have the students look at the picture in Figure 1. On the left are three people: Alice; Bob; and Granny. On the right are three tasks: sawing firewood; putting up a fence; and fixing a leaky faucet. On the line connecting a person to a job is the number of hours it will take the person to do the job. Now show them the two possible assignments in Figure 2 and ask which is better. Some may note that in the green assignment on the left the total time worked is 6 hours, which is much less than the 9 total hours required by the red assignment on the right. But, suppose that they all drove to work in the same car (only Granny can drive) so they have to stay until the last task is completed. In this case, the green assignment will require them to stay for 4 hours, while the red assignment will keep them only three hours. As a final example show them the purple assignment in Figure 3. Is the purple assignment better than the green one? It probably is because it requires the same total hours of work (6 hours) in less total elapsed time. Is this assignment better than the red one? It probably is because it has the same elapsed time (3 hours) but requires only six total hours of combined work.

Does any student mention that the red one may be *fairer* because everyone does the *same* amount of work? If not, raise this question. Refer back to Figure 2 and compare the two matches and note that there is no one who has to work longer in the purple match than they did in the red match, so no one is penalized if you switch from red to purple.

- Some people might *enjoy* doing certain jobs more than others, so it might be better to assign people to jobs they enjoy. (But do we have the data needed to include "enjoyment" in our matching criteria?)

As a slight variation on this last point, consider a case in which there are 5 people who each interview for 5 jobs. When the interviews are complete, each person ranks each potential employer, with 1 being most preferred and 5 being least preferred (but all jobs being more desirable than their current situation.) The employers likewise rank the 5 candidates they interviewed. What makes a good match here? This question will be the starting point for the following demonstration.

The following page is the Student handout of the figures mentioned above.

Figures for Make Me a Match Discussion

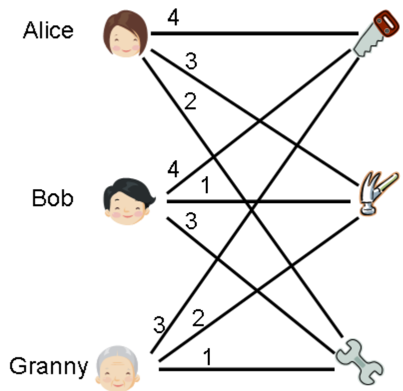


Figure 1

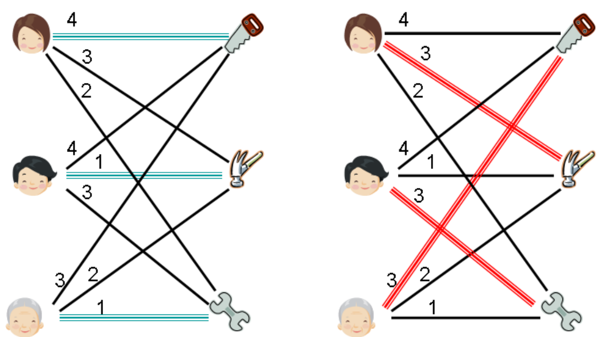


Figure 2

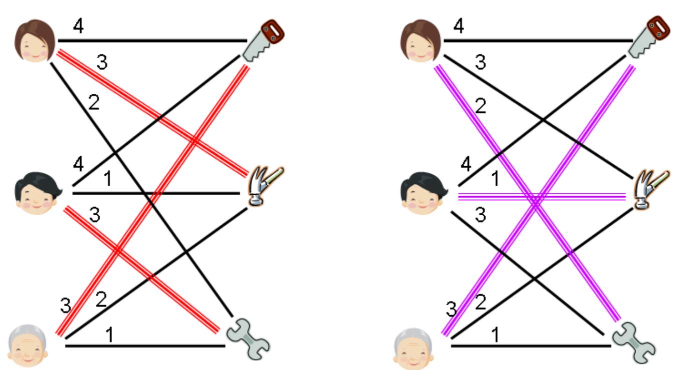


Figure 3

Teacher notes: Demonstration: Representing Matching Problems with Matrices

Distribute the handout **Representing Preferences: Lists and Matrices**.

We are about to begin a detailed study of one type of matching problem: that of matching one group with another group in a one-to-one manner. The basic assumption is that everyone wants to be matched and that everyone has preferences regarding whom they are matched with. Consider a Big Sibling organization that matches Big Siblings with Little Siblings. Suppose for the winter program, there are five big siblings and five little siblings to be matched. The people all read brief descriptions of the other group members' interests and personality traits. Then they rank the members of the other group in terms of whom they would most like to be matched with. Suppose Big Sibling A is most interested in working with Little Sibling w, followed by Little Sibling y, then x, then z, and finally v.

It is important to note that the ranking is *relative* and not *absolute*. In fact, Big Sibling A might be equally happy being assigned to *any* Little Sibling, but we required him or her to come up with a ranking without ties. After asking all the big and little siblings to rank each other, we would have ten lists of preferences (p. 1 of the handout). Make sure students understand how to read the lists, by asking questions such as: "which Little Sibling is D's third choice?" [answer: z]; "which Big Sibling is most often ranked first by the Little Siblings?" [answer: E]; "how have the Little Siblings ranked Big Sibling A?" [answer: 1, 2, 2, 3, 5].

Next, start students on the activity of organizing the information in the preference lists into a preference matrix (p. 2 of handout). Point out that matrices provide another useful way of organizing preference lists.

Once students have completed the preference matrices, make sure they know how to read them because preference matrices will be used throughout the rest of this module. Check their understanding by asking questions such as: "who is Big

Sibling's B's last choice?" [answer: y]; "which Big Sibling appears least popular among the Little Siblings?" [answer: B is ranked last (5^{th}) by two Lil Sibs, is ranked second to last (4^{th}) by two others, and is ranked 3^{rd} by the other Lil Sib; no one has ranked Big Sibling B higher than 3^{rd}].

In the matrices below a 1 indicates a person's first preference and a 5 indicates the person's last preference. For example, Big Siblings C's first choice is Little Sibling x and C's second pick is y. Similarly, Little Sibling v's first choice is Big Sibling E and v's last choice is B.

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Help students understand how to read preference matrices before assigning the homework **Finding a Good Matching.**

Handout Name:____________________________

Representing Preferences: Lists and Matrices p. 1 of 2

The Big Sibling organization has collected the preferences both of the students wanting to volunteer as a Big Sibling and of the Little Siblings, those younger students wanting to be matched with a Big Sibling.

Below are the two sets of lists in which everyone's first choice is listed first. For example, in reading these lists we see that both Big Siblings C and E have as their first choice Little Sibling x.

Big Sib Preference Lists

	1st	2nd	3rd	4th	5th
A:	w	y	x	z	v
B:	w	x	z	v	y
C:	x	y	w	v	z
D:	w	x	z	v	y
E:	x	y	w	v	z

Lil Sib Preference Lists

	1st	2nd	3rd	4th	5th
v:	E	A	D	C	B
w:	C	E	A	D	B
x:	A	C	E	B	D
y:	D	A	E	B	C
z:	E	C	B	D	A

What else do you notice about this way of representing the preferences?

Handout Name:______________________________

Representing Preferences: Lists and Matrices p. 2 of 2

Matrices (or tables of numbers) are another way of representing all the information that is in the preference lists on page 1. In this case, the rows and columns correspond to the people we are trying to match. Each entry in the body of the matrix will indicate how one person has ranked another person. As with the preference lists, we will need two preference matrices: one for the Big Sibs' rankings of the Lil Sibs, and one for the Lil Sibs' rankings of the Big Sibs. Fill in the matrices using the preferences given on page 1.

Big Sib Preference Matrix

	v	w	x	y	z
A					
B					
C					
D					
E					

Lil Sib Preference Matrix

	A	B	C	D	E
v					
w					
x					
y					
z					

What do you notice about this way of representing the preferences?

Teacher notes: Introducing **Finding a Good Matching** Homework

This homework assignment introduces students to the Big Sibling Little Sibling case study. It asks them to evaluate potential matchings and to create matchings of their own that they think would be optimal. Impress upon them the need to examine the matchings in detail and to use their imaginations in evaluating each matching from several perspectives, including those of the individuals being matched.

If students raise the topic it might be worth discussing how matching problems of this sort change if the rankings are public (i.e., everyone knows everyone else's rankings) or private (i.e., no one, except the match maker, sees all the rankings). The private or public aspect of preferences can create various interesting problems; however, we do not study this aspect in this module.

The homework provides students with the opportunity to develop their own notion of stability even though they might not call it that initially. Unit 2 formalizes the concepts of stability and stable matchings using the same preference matrices used in this homework.

Homework: Finding a Good Matching Name:______________________________

Introducing the Big Sibling/Little Sibling Case Study

You have been hired as a computational consultant by the local Big Sibs organization to help them figure out the best matches between the little siblings and the big siblings. The organization collected the preferences of each little sibling and each big sibling and organized the data into preference matrices. Uppercase letters refer to the big siblings and lowercase letters refer to the little siblings.

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

The director of the organization has given you two tasks: 1) to evaluate a few possible matchings that they think might be good, and 2) to see if you can find any matchings that would be better than theirs.

1. Evaluate these potential matchings and explain their strengths and weaknesses.

I { {A, x}, {B, z}, {C, v}, {D, y}, {E, w} }

II { {E, v}, {A, w}, {B, x}, {D, y}, {C, z} }

III { {C, x}, {A, w}, {E, y}, {B, z}, {D, v} }

2. Find another potential matching that you think the organization should consider and explain its advantages and disadvantages.

Unit 2: Defining Stability in Matching Problems

Objective: Students will be able to explain *instability* and *stable matching*.

Teacher Notes: HW Discussion of the Big Sibling Little Sibling Case Study

The homework in Unit 1 introduced students to the Big Sibling case study. The focus in this unit is on understanding *stability* in the context of matching problems and on beginning to develop a method for finding stable matchings. Ask the students to share their assessments of the strengths and weaknesses of the three proposed matchings in problem 1 of the homework. Through discussion, the goal is to arrive at an understanding of a type of *instability* that is undesirable in a good matchings.

If the students don't focus on the preferences that each sibling received, suggest that lists such as the ones below be used to evaluate the matchings. Next to each sibling is the preference number (ranking) of the person that sibling is assigned to. Point out the various ways in which each list might be ordered, by Big Sibs, by Lil Sibs, or by overall satisfaction level of the matched pairs.

I				II				III			
3	A	x	1	4	E	v	1	1	C	x	2
3	B	z	3	1	A	w	3	1	A	w	3
4	C	v	4	2	B	x	4	2	E	y	3
5	D	y	1	5	D	y	1	3	B	z	3
3	E	w	2	5	C	z	2	4	D	v	3

In matching I, if students don't raise the issue themselves, lead them to consider how *stable* the matching is; suggest that they consider how the individuals who received their lower preferences feel.

Finally, you can pose a more pointed question: are there two people who are not matched with each other who would each prefer the other to their current partner? And if needed, you can explicitly ask students to consider individuals D and v, or to consider individuals C and w. Both of these pairs are unstable. The next class activity focuses on testing the stability of matchings I, II, and III.

If time permits, you can explore whether students would evaluate the matchings differently if all of the preferences were public. Another optional topic for discussion is whether the evaluation of the matchings would change if the context changed, for example, from that of Big and Little Siblings to jobs and job applicants (in this context the boss would rank the applicants in terms of their suitability for the jobs—since jobs cannot express preferences!).

Teacher Notes: Class Activity: ***Testing for Stability***

Hand out the three activity sheets and help students develop a good understanding of what an *instability* is. The presence of even just one pair of people who are not currently matched but who *each* prefer the other to their current partners causes the entire matching to be *unstable*. Students sometimes forget about the mutuality requirement. Students often assume a matching is unstable if just one person receives his or her lowest preference; however, if no one else prefers that unfortunate person to his or her assigned partner, then that unfortunate person is not part of an instability. Unrequited love does not create an instability! It takes two.

To show that a matching is unstable, only *one* instability needs to be found.

To prove that a matching is stable, *every* possible pair of people who might create an instability must be checked; an exhaustive search must be performed.

The essence of an instability is two people who *each* would be better off if assigned to each other than to their current partners. Reassigning these two people to each other might not turn the matching into a stable one as there could be other instances of such mutual preferences.

On p. 1 of the activity handout, attention is focused on D and v. However, you can also point out that C and w create an unstable pair: C prefers w to v *and* w prefers C to E. The preferences below reveal this instability (i.e., 3 < 4 and 1 < 2).

Matching I

3	A	x	1
3	B	z	3
4	***C***	v	4
5	D	y	1
3	E	***w***	***2***

Mutually Preferred Match

3	***C***	***w***	***1***

Next, on p. 2, have students work in pairs to test the stability of Matching II. Below are the conclusions they should arrive at.

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Matching II

4	E	v	1
1	A	w	3
2	B	x	4
5	D	y	1
5	C	z	2

Matching II is unstable because of any one of these four instabilities:

1 w & C 3   2 w & E 3   2 x & C 1   3 x & E 1

For example, the pair 1 w & C 3 is better for w *and* C because for w, 1st is better than 3rd, and for C, 3rd is better than 5th. After students have found one or more of the instabilities above, discuss aspects of checking efficiently for instabilities. Have the students draw boxes around those people who received their top choices. These people cannot be part of an instability because they prefer no one to their current partners; so they can be ignored.

This means that B, C, D, E, w, x, and z are the only ones who might be part of an instability. Because there are three little siblings, compared with four big siblings, we might save time by focusing only on the little siblings.

Checking z is easy, because z only prefers E (ranked 1st) over C (2nd); however, E does not prefer z (ranked 5th) to v (4th), so z is not a part of any instability.

Checking w means checking how C & E feel about w since they are the only two options ranked higher than 3rd (w's ranking of its current partner). Because both C and E prefer w to their current partners, either one creates an instability with w.

Checking x means checking only how C and E feel about x; we can ignore A because although x ranks A 1st, A ranks x 3rd, lower than A's assigned partner of w. Both C and E would prefer to be with x than with their current partners z and v respectively. These two pairs, x & C, and x & E create instabilities.

Finally, note that even though D is assigned to his or her last (5th) choice, no one prefers D to his or her current partner and so no one forms an instability with D.

When students are ready, have them move on to p. 3 and determine if Matching III is stable. It turns out that Matching III is stable, but to prove this fact, all of the possible instabilities must be checked for mutually preferred alternatives. Suggest that students begin by boxing off all those people receiving their first choices.

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Matching III

1	C	x	2
1	A	w	3
2	E	y	3
3	B	z	3
4	D	v	3

After students conclude that the matching is indeed stable, insist on a patient exhaustive checking of the possibilities for instabilities. All of the following six pairs must be checked to see if any of the little siblings also prefer the big siblings to their current partners.

E & x B & w B & x D & w D & x D & z

Because x does not prefer E (3rd) or B (4th) or D (5th) to its current match of C (2nd), x is not part of any instability. The same is true of w and A (3rd) in relation to B (5th) and D (4th), and of z and B (3rd) in relation to D (4th). This exhaustive elimination of possible instabilities means the matching is stable!

Conclude the class activity on stability by reinforcing the definition of stability as it is applied to matchings and by distinguishing stability from happiness or fairness. A matching can be stable even if some people do not get their top preferences. Putting the two simple examples below on the board and discussing them can reinforce the distinction between stability and other aspects of stable matching problems.

Example 1 Preference Matrices

	w	**x**	**y**	**z**
A	1	2	3	4
B	4	1	2	3
C	3	4	1	2
D	2	3	4	1

	A	**B**	**C**	**D**
w	4	1	2	3
x	3	4	1	2
y	2	3	4	1
z	1	2	3	4

For the above matrices, in which people's mutual preferences are inversely related, each of the following two matchings is in fact a stable matching despite the fact that one group or the other receives their lowest preferences. Other, less extreme, stable matchings are possible; however the point is that stability, happiness, and fairness are not always coincident.

1	A	w	4
1	B	x	4
1	C	y	4
1	D	z	4

4	A	z	1
4	B	w	1
4	C	x	1
4	D	y	1

Example 2 Preference Matrices

	w	**x**	**y**	**z**
A	1	2	3	4
B	4	1	2	3
C	3	4	1	2
D	2	3	1	4

	A	**B**	**C**	**D**
w	1	4	2	3
x	3	1	4	2
y	2	3	1	4
z	1	2	3	4

For the above matrices, the matching below is stable and in fact it is the only stable matching. No one likes D or z enough to "rescue" them from their pairing.

1	A	w	1
1	B	x	1
1	C	y	1
4	D	z	4

Class Activity: **Testing for Stability** Name:______________________

Defining *Stable Matching*

A matching is *stable* if no two people *each* prefer the other to their current partners. A matching is *unstable* if at least one pair of people have *each* ranked the other higher than their current partners. Such an unstable pair is called an instability.

To prove that a matching is stable requires checking every person who did not get his or her first preference. Do any of the people whom a person ranked higher than his or her assigned partner also prefer that person to his or her assigned partner?

Sometimes you can quickly find an instability by focusing on the people who are assigned low preferences. Consider big sibling D and little sibling v in Matching I from the homework. D is assigned to y but prefers v; and v prefers D over C. Because of their mutual preference for each other over their current partners, Matching I is unstable. For D, 4 is better than 5, and for v, 3 is better than 4. Other instabilities might exist; however the presence of just one instability makes a matching unstable.

Matching I

3	A	x	1
3	B	z	3
4	C	***v***	***4***
5	***D***	y	1
3	E	w	2

Mutually Preferred Match

4	***D***	***v***	***3***

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Testing for Stability Name:______________________

With a partner, determine if Matching II from the homework is stable or unstable.

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Matching II

4	E	v	1
1	A	w	3
2	B	x	4
5	D	y	1
5	C	z	2

Testing for Stability Name:________________________

With a partner, determine if Matching III from the homework is stable or unstable.

Big Sibling Preferences

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

Little Sibling Preferences

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Matching III

1	C	x	2
1	A	w	3
2	E	y	3
3	B	z	3
4	D	v	3

Teacher Notes: Assigning Homework: **A Method for Finding Stable Matchings**

Point out to students that when we check for stability what we are really doing is seeing if each person's unassigned higher preferences would rather be with him or her than with *their* current partners.

This insight is the basis of the Gale-Shapley proposal algorithm presented in the next unit. In that algorithm, each member of one group is given a chance to partner with his or her top preferences, in order, until they are each accepted by someone in the other group. Each person in the other group only gets to reject someone if a better option comes along. By working through the preferences in order, there can be no instabilities in the end.

Without mentioning the idea of proposals or that the Gale-Shapley algorithm exists, suggest that students think about what might happen if all the big and little siblings were put in a room and left to sort out their own best matching; and then suggest that they think of a way of organizing that process. If you want to give overwhelmed students a hint, consider mentioning that perhaps some form of "speed dating" might be useful. It can also be helpful to ask students to state how they would personally go about finding a match if they were participating in this process.

Designing and writing an algorithm is a challenging task. Engaging in the work is more important than succeeding with the articulation of a perfect algorithm. Encouraging students to work in pairs and giving them some time in class to get started will be useful. Encourage them to come up with anything that seems reasonable even if it is not perfect.

Name:______________________________

Homework: A Method for Finding Stable Matchings

1. Design an algorithm (a step-by-step procedure or list of instructions) for finding a stable matching. Your algorithm must be stated completely, concisely, clearly, and precisely enough for someone else to use successfully. Your algorithm must work with any two pairs of preference matrices that someone might have.

2. After you have written down your algorithm apply it to the set of preference matrices in the Big Sibling Little Sibling case study to demonstrate how it works and to find what you think is a stable matching for this set of preference matrices.

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Unit 3: Stable Matching Algorithms

Objective: Students will design their own stable matching algorithms and test them, and then students will be able to explain and use the Gale-Shapley algorithm to find stable matchings.

Teacher Notes: ***Activity: Comparing and Testing of Student Algorithms***

The last homework or activity had the students devise an algorithm for finding a stable matching when given any pair of preference matrices. Describing algorithms in enough detail to enable someone other than the author to use the algorithm is very challenging. This articulation challenge compounds the difficulty of simply creating an algorithm that will work.

Unless students are highly engaged in perfecting their algorithms, it is sufficient to point out that the development of good algorithms is hard work but also very rewarding intellectually and sometimes economically as well.

Finding a good way to solve a type of problem is like solving a mystery that no one else has figured out, and many organizations pay people a lot of money for developing new algorithms. In fact there are some problems for which no one yet knows whether a good algorithm can even be created. Finding the prime factorization of large numbers is one such open problem with great significance for cryptography that is used in national security and in online banking transactions.

Students' answers will vary, so they should pair off to read and troubleshoot each other's algorithms. As a class you can discuss those that hold the most promise.

When the class is ready, move on to the next class activity that presents the Gale-Shapley algorithm for finding stable matchings. As you help the students understand the algorithm point out any similarities between it and the algorithms the students devised.

Teacher Notes: ***Class Activity on the Gale-Shapley Algorithm***

Use the handout *The Gale-Shapley Algorithm* to explain the algorithm. First, review what the preference matrices represent by pointing out that Big Siblings B and D have identical preferences, as do Big Siblings C and E. Note that v is the least-preferred Little Sibling while w and x are highly preferred. Also call attention to the fact that the Little Siblings' first preferences are largely distinct—only v and z have ranked E first.

The Gale-Shapley algorithm avoids a brute-force exhaustive checking of all possible matchings and follows what might play out if *n* BigSibs and *n* LilSibs were to pursue each other according to their preferences. For reasons that will become clear, the Gale-Shapley algorithm is also known as the Proposal algorithm: the BigSibs, in turn, propose to the LilSibs who accept or reject the proposals according to their own preferences. It is important to emphasize that everyone would prefer to be matched (even with their last preference) than not to be matched. More precisely, here are the steps of the Gale-Shapley (GS) algorithm stated for when the BigSibs propose.

Statement of the Gale-Shapley Algorithm

1. One at a time and in any order, the BigSibs each propose to his or her highest ranked LilSib who has not rejected the BigSib.
2. After each proposal, the LilSib either must tentatively accept the proposal (if he or she is unengaged) or must choose the higher ranked of the two BigSibs (if he or she is already engaged).
3. Any BigSib who is rejected by a LilSib proposes to the next LilSib on his or her list before another BigSib makes a proposal.
4. Repeat steps 2 & 3 until there are no LilSibs entertaining competing proposals.
5. Repeat steps 1 through 4 until all the BigSibs and LilSibs are engaged.

The resulting set of matches will be a stable matching because each BigSib has attempted to partner in order with his most preferred choices and because each LilSib has accepted the proposal from the BigSib he or she most prefers of those that have proposed. There can be no instability because no LilSib has received a proposal from a BigSib that he or she might prefer to his or her current match, and because each BigSib has been rejected by the LilSibs he might have preferred to his or her current match. This reasoning is a proof of the correctness of the algorithm.

Note that the order in which the BigSibs propose makes no difference because eventually everyone works through his or her preference list in order. Students will understand the algorithm better after they use it on homework problems, checking their work with each other and with the solutions. First, lead them through the example below.

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Example 1. Preference matrices for a Big/Little Sibling matching problem in the handout, *The Gale-Shapley Algorithm*.

Let's first use the algorithm with the BigSibs making the proposals and let's work in alphabetical order since the order of proposals doesn't matter. Apply step 1 (S1) which results in the first proposal of A to w. It will make checking student work much easier if we keep track of the proposals in the order they occur. There is potential for confusion, so we will put the proposal number in a box; these boxed numbers are not to be confused with either the algorithm step (always designated with an S) or a BigSib's preference (always read in the matrix). To indicate that A and w are matched, the student's handout should now read:

A [1] w

This reflects completion of S2 since w must accept A's proposal.

Because there are no LilSibs with multiple proposals, we apply S1 again and this time it results in B proposing to w. The student's chart should now appear as follows:

A	1	w				
B	2	w				

Now we must take care with S2; because w has competing proposals, we look at w's preference list and find that w prefers A (ranked 3) to B (ranked 5). So w rejects B.

Step S3: B proposes to B's next most highly ranked LilSib, which is x. So the proposal chart should now look like:

A	1	w				
B	2	w	3	x		

Note that the boxed 3 means that B's proposal to x is the third one the algorithm has instructed us to make; the fact that it occurred during step S3 is coincidental.

Step S1: C proposes to x. This step results in competing proposals for x which must be resolved.

A	1	w				
B	2	w	3	x		
C	4	x				

Step S2: Because x prefers C (ranked 2) to B (ranked 4), x rejects B requiring step S3.
Step S3: B proposes to B's next most highly ranked LilSib, which in this case is z. The boxed 5 means that B's proposal to z is the fifth one we've made so far.

A	1	w				
B	2	w	3	x	5	z
C	4	x				

At this point, we have a tentative and partial matching of {{A, w}, {B, z}, {C, x}}.

Step S1: BigSib D proposes to its top choice, w (this is the 6th proposal overall).

A	1	w				
B	2	w	3	x	5	z
C	4	x				
D	6	w				

Step S2: LilSib w must choose between BigSibs A and D; w prefers A (ranked 3) to D (ranked 4) and so rejects D's proposal.

Step S3: D proposes to x, making proposal number 7.

A	1	w				
B	2	w	3	x	5	z
C	4	x				
D	6	w	7	x		

Step S2: Now LilSib x must choose between C and D and chooses C.

Step S3: D proposes to z.

Step S2: z rejects D preferring to remain with B.

Step S3: D proposes to v who must accept because v is not currently engaged.

A	1	w						
B	2	w	3	x	5	z		
C	4	x						
D	6	w	7	x	8	z	9	v

At this point, we have a tentative and partial matching of {{A, w}, {B, z}, {C, x}, {D,v}}.

Step S1: E proposes to x.

Step S2: x rejects E, preferring C to E.

Step S3: E proposes to y; done! The result is {{A, w}, {B, z}, {C, x}, {D,v}, {E,y}}.

A	1	w						
B	2	w	3	x	5	z		
C	4	x						
D	6	w	7	x	8	z	9	v
E	10	x	11	y				

On page 2 of the handout *The Gale-Shapley Algorithm*, use the Gale-Shapley algorithm again but this time with the LilSibs doing the proposing. Because the LilSibs' preferences conflict less often than the BigSibs' preferences, only seven proposals are needed. The first four proposals resulting from repeated uses of Steps S1 and S2 result in no competing proposals to the same BigSib.

v	1	E
w	2	C
x	3	A
y	4	D

However when LilSib z proposes to E, BigSib E chooses to remain paired with v (ranked 4) and rejects z (ranked 5) and so Step S3 is needed.

z	5	E

Step S3: LilSib z proposes to C.

z	5	E	6	C

Step S2: BigSib C rejects z (ranked 5) and remains paired with w (ranked 3).
Step S3: LilSib z proposes to BigSib B and we are done because all Big and Little Siblings are matched with no conflicts.

z	5	E	6	C	7	B

The stable matching we have found is: {{A, x}, {B, z}, {C, w}, {D,y}, {E,v}}.

Note! The stable matching we found by having the LilSibs do the proposing is *not* the same as the stable matching found when the BigSibs do the proposing!
The first stable matching we found was: {{A, w}, {B, z}, {C, x}, {D,v}, {E,y}}.

The class discussion **Now What?!** examines the possibility of finding more than one stable matching for a set of preference matrices.

Name:____________________________________

Handout: The Gale-Shapley Algorithm p. 1 of 2

1. One at a time and in any order, the BigSibs each propose to his or her highest ranked LilSib who has not rejected the BigSib.
2. After each proposal, the LilSib must either tentatively accept the proposal (if he or she is unengaged) or must choose the higher ranked BigSib (if he or she is already engaged).
3. Any BigSib who is rejected by a LilSib proposes to the next LilSib on his or her list before another BigSib makes a proposal.
4. Repeat steps 2 & 3 until there are no LilSibs entertaining competing proposals.
5. Repeat steps 1 through 4 until all the BigSibs and LilSibs are engaged.

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Big Sibling Proposals

A

B

C

D

E

Name:_______________________________

Handout: The Gale-Shapley Algorithm p. 2 of 2

1. One at a time and in any order, the LilSibs each propose to his or her highest ranked BigSib who has not rejected the LilSib.
2. After each proposal, the BigSib must either tentatively accept the proposal (if he or she is unengaged) or must choose the higher ranked LilSib (if he or she is already engaged).
3. Any LilSib who is rejected by a BigSib proposes to the next BigSib on his or her list before another LilSib makes a proposal.
4. Repeat steps 2 & 3 until there are no BigSibs entertaining competing proposals.
5. Repeat steps 1 through 4 until all the LilSibs and BigSibs are engaged.

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Little Sibling Proposals

v

w

x

y

z

Name:___**WORKED DETAILS**_________

Handout: The Gale-Shapley Algorithm p. 1 of 2

1. One at a time and in any order, the BigSibs each propose to his or her highest ranked LilSib who has not rejected the BigSib.
2. After each proposal, the LilSib must either tentatively accept the proposal (if he or she is unengaged) or must choose the higher ranked BigSib (if he or she is already engaged).
3. Any BigSib who is rejected by a LilSib proposes to the next LilSib on his or her list before another BigSib makes a proposal.
4. Repeat steps 2 & 3 until there are no LilSibs entertaining competing proposals.
5. Repeat steps 1 through 4 until all the BigSibs and LilSibs are engaged.

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Big Sibling Proposals

A [1] w

B [2] w [3] x [5] z

C [4] x

D [6] w [7] x [8] z [9] v

E [10] x [11] y

Stable Matching: {{A, w}, {B, z}, {C, x}, {D, v}, {E, y}}

Name:___WORKED DETAILS__________

Handout: The Gale-Shapley Algorithm p. 2 of 2

1. One at a time and in any order, the LilSibs each propose to his or her highest ranked BigSib who has not rejected the LilSib.
2. After each proposal, the BigSib must either tentatively accept the proposal (if he or she is unengaged) or must choose the higher ranked LilSib (if he or she is already engaged).
3. Any LilSib who is rejected by a BigSib proposes to the next BigSib on his or her list before another LilSib makes a proposal.
4. Repeat steps 2 & 3 until there are no BigSibs entertaining competing proposals.
5. Repeat steps 1 through 4 until all the LilSibs and BigSibs are engaged.

	v	w	x	y	z
A	5	1	3	2	4
B	4	1	2	5	3
C	4	3	1	2	5
D	4	1	2	5	3
E	4	3	1	2	5

	A	B	C	D	E
v	2	5	4	3	1
w	3	5	1	4	2
x	1	4	2	5	3
y	2	4	5	1	3
z	5	3	2	4	1

Little Sibling Proposals

v 1 E

w 2 C

x 3 A

y 4 D

z 5 E 6 C 7 B

Stable Matching: {{A, x}, {B, z}, {C, w}, {D, y}, {E, v}}.

Teacher Notes: *Class Discussion:* ***Now What?!***

Using the prompts below, lead a discussion of the properties of the two stable matchings found with the Gale-Shapley algorithm for the Big Sib Little Sib case study. After students share their observations, conjectures, and questions, emphasize the points following each prompt.

Prompt: Compare the two stable matchings. Is one group, the LilSibs or the BigSibs, happier with one of the matchings? Suggest that the students record which preference each person wound up with in each stable matching. Below is one way of organizing this information. When the BigSibs proposed, A and w were matched, and each received his or her 1st and 3rd preferences respectively; similarly B and z were matched and they each obtained their 3rd preferences.

BigSibs Proposing				LilSibs Proposing			
1	A	w	3	3	A	x	1
3	B	z	3	3	B	z	3
1	C	x	2	3	C	w	1
4	D	v	3	5	D	y	1
2	E	y	3	4	E	v	1

BigSib Average: 2.2 BigSib Average: 3.6

LilSib Average: 2.8 LilSib Average: 1.4

Key Ideas: The stable matching that results when one group does the proposing is the best possible outcome for that group. In this example, we get a BigSib *optimal* matching of {{A, w}, {B, z}, {C, x}, {D,v}, {E,y}} and we get a LilSib *optimal* matching of {{A, x}, {B, z}, {C, w}, {D,y}, {E,v}}. We can compare the average rankings received by each group as one way to assess fairness, the lower the average, the better; an average of 1 would mean that everyone received his or her 1st preference. Note, there are many other ways to measure or assess fairness; such strategies are the topic of Unit 4.

Prompt: For some matching problems, the Gale-Shapley algorithm will find two different stable matchings. Might there be more than two stable matchings for a set of preference matrices? How many different stable matchings do you think exist for the preference matrices in the handout? How many did you find for this problem in Unit 1?

Key Ideas: Yes, this instance of the problem (i.e., this pair of preference matrices) has another stable matching; it is: {{A, y}, {B, z}, {C, x}, {D, v}, {E, w}}. In general, stable-matching problems can have one, two, three, four, or many different stable matchings. The Big Sib average is 2.6 and the Lil Sib average is 2.4. These averages are nice compromises between the averages of the other two stable matchings.

It is somewhat surprising that *every* pair of preference matrices will have *at least one* stable matching. No matter how conflicting or contrary individual preferences are, there will always be at least one stable matching. As an extension, you can challenge skeptical students to design pairs of preference matrices that they think will have no stable matchings; if they use the Gale-Shapley algorithm they will *always* find at least one stable matching.

The Gale-Shapley algorithm will always find at least one stable matching and can find at most two stable matchings. If the two matchings found (from each of the two groups doing the proposing) are identical, then there is *only one* stable matching possible and it is optimal for both groups.

If the two matchings found (from each of the two groups doing the proposing) are different, then we can conclude only that *at least two* stable matchings exist. To find out if other stable matchings exist, other time-intensive computational methods are needed to search for all possible stable matchings. A computer is obviously needed when large numbers of things are being matched!

Teacher Notes: Assigning Homework: **The Gale-Shapley Algorithm**

This homework gives students practice with the G-S algorithm and checks their understanding of what the algorithm can reveal about a stable-matching problem.

Problem 3 asks the students to create a stable-matching problem by specifying the two preference matrices. They are encouraged to select preference matrices with some interesting property. As long as the preference matrices are legitimate rankings, they can be anything at all. To be a legitimate preference matrix, each *row* must contain each of the numbers 1 through 6 once and only once; this requirement follows from each person ranking all of the available partners from first to last with no ties permitted.

For students who need help thinking of something "interesting" you can suggest such situations as:
what if two or more people had the exact same preferences?
what if everyone's first choice ranked that person last in return (opposites attract)?
what if the two matrices are identical?

A couple of situations might be gently discouraged depending on the student:
Some students want every one's first choice to rank that person first in return; but a little thought should lead students to conclude that this would simply lead to a "perfect" stable matching in which everyone gets his or her first preference; not very interesting.

Some students will want to create "random" matrices; but it's not clear what would be learned from the stable matching(s) found. Exploring random matrices only really makes sense if you are able to explore a very large number of random matrices in an attempt to draw conclusions about what can be expected on average; exploring just one random example doesn't tell us much in isolation.

Name:______________________________

Homework: The Gale-Shapley Algorithm p. 1 of 2

1. Use the Gale-Shapley algorithm to find stable matching(s) for the preference matrices below. Label your proposals in the order they occur, as we did in class.

	s	t	u	v	w	x	y	z
A	5	8	7	1	4	6	2	3
B	3	4	6	2	5	7	8	1
C	4	1	3	5	8	2	6	7
D	7	6	8	3	2	4	1	5
E	3	5	8	6	7	2	1	4
F	8	4	1	6	5	3	7	2
G	3	6	8	2	4	5	7	1
H	2	1	7	3	4	5	6	8

	A	B	C	D	E	F	G	H
s	8	5	6	7	2	3	1	4
t	7	1	6	5	4	8	3	2
u	3	6	4	2	7	8	1	5
v	6	7	3	8	1	4	5	2
w	1	8	6	7	3	5	4	2
x	6	3	5	2	1	4	7	8
y	3	4	1	5	2	7	8	6
z	2	6	8	3	7	5	4	1

2. How many stable matchings do you think exist for the matrices above? Why?

Name:________________________________

Homework: The Gale-Shapley Algorithm p. 2 of 2

3. Create a pair of preference matrices with some interesting property or distinctive feature and then use the Gale-Shapley algorithm to find a stable matching (or two if a second one exists). Explain what is special about your matrices and explain what the Gale-Shapley algorithm revealed about them.

	u	v	w	x	y	z
A						
B						
C						
D						
E						
F						

	A	B	C	D	E	F
u						
v						
w						
x						
y						
z						

4. Explain what the Gale-Shapley algorithm can tell us about a stable-matching problem.

Name:__SOLUTIONS___________

Homework: The Gale-Shapley Algorithm p. 1 of 2

1. Use the Gale-Shapley algorithm to find stable matching(s) for the preference matrices below. Label your proposals in the order they occur, as we did in class.

	s	t	u	v	w	x	y	z
A	5	8	7	1	4	6	2	3
B	3	4	6	2	5	7	8	1
C	4	1	3	5	8	2	6	7
D	7	6	8	3	2	4	1	5
E	3	5	8	6	7	2	1	4
F	8	4	1	6	5	3	7	2
G	3	6	8	2	4	5	7	1
H	2	1	7	3	4	5	6	8

	A	B	C	D	E	F	G	H
s	8	5	6	7	2	3	1	4
t	7	1	6	5	4	8	3	2
u	3	6	4	2	7	8	1	5
v	6	7	3	8	1	4	5	2
w	1	8	6	7	3	5	4	2
x	6	3	5	2	1	4	7	8
y	3	4	1	5	2	7	8	6
z	2	6	8	3	7	5	4	1

A 1 v

B 2 z 9 v 10 s

C 3 t 12 x

D 4 y 6 w

E 5 y

F 7 u

G 8 z

H 11 t

s 1 G

t 2 B

u 3 G 4 D

v 5 E 8 H

w 6 A 12 H 13 E 14 G 15 F

x 7 E

y 9 C

z 10 H 11 A

2. How many stable matchings do you think exist for the matrices above? Why?

Answers will vary. One reasonable conjecture is that because all of the pairs are different and because several individuals' higher preferences had ranked them lower, there could be many more stable matchings.

[Students have no way of knowing this unless some *very* motivated students checked a huge number of possibilities, but there are 16 stable matchings in total!]

Name:__SOLUTIONS__________

Homework: The Gale-Shapley Algorithm p. 2 of 2

3. Create a pair of preference matrices with some interesting property or distinctive feature and then use the Gale-Shapley algorithm to find a stable matching (or two if a second one exists). Explain what is special about your matrices and explain what the Gale-Shapley algorithm revealed about them.

	u	v	w	x	y	z
A	1	2	3	4	5	6
B	1	2	3	4	5	6
C	1	2	3	4	5	6
D	1	2	3	4	5	6
E	1	2	3	4	5	6
F	1	2	3	4	5	6

	A	B	C	D	E	F
u	1	2	3	4	5	6
v	1	2	3	4	5	6
w	1	2	3	4	5	6
x	1	2	3	4	5	6
y	1	2	3	4	5	6
z	1	2	3	4	5	6

Answers will vary. The above pair is common: everyone's choices are the same but a stable matching is still possible!

A 1 u
B 2 u 3 v
C 4 u 5 v 6 w
D 7 u 8 v 9 w 10 x
E 11 u 12 v 13 w 14 x 15 y
F 16 u 17 v 18 w 19 x 20 y 21 z

u 1 A
v 2 A 3 B
w 4 A 5 B 6 C
x 7 A 8 B 9 C 10 D
y 11 A 12 B 13 C 14 D 15 E
z 16 A 17 B 18 C 19 D 20 E 21 F

4. Explain what the Gale-Shapley algorithm can tell us about a stable-matching problem. The G-S algorithm will always find one or two stable matchings for any problem. When both proposal processes yield the same stable matching, it is the only one possible. If the two proposal processes yield two different stable matchings, we can only conclude that there are at least two stable matchings; there might be more than two but the G-S algorithm can't determine that.

Teacher Notes: Checking **Homework: The Gale-Shapley Algorithm**

Have students check each other's homework. Make sure everyone knows how to use the Gale-Shapley algorithm and understands what it can and cannot reveal about stable-matching problems.

Students should carefully check each other's problem 3. In fact, the best approach is for one student to cover up another student's answer use the G-S algorithm; his or her result should match the work of the student who created the matrices. This can provide a further check on whether students understand how to use the algorithm.

Note that the preference matrices in problem 1 are the basis for all the work that is done in Unit 4. The actual answer to problem 2, about how many stable matchings might exist for the set of preference matrices in problem 1, can set the stage for Unit 4. Students might be shocked to learn that there are 16 different stable matchings for those preference matrices.

Unit 4: Measures of Fairness

Teacher Notes: Introducing the "Fairest of All" Case Study

Many stable-matching problems have many stable matchings, any one of which could be selected as the solution to the problem. Which stable matching is selected as a solution depends on criteria other than stability. Different people, with different values or priorities, might select different stable matchings as being the preferred solution. In this case study, students must come up with a measure of fairness—a way of determining which of many different stable matchings is the fairest; and they must defend their way of assessing fairness by appealing both to principles of fairness that they define and to computations that assign numerical measures of fairness to the different stable matchings.

If you haven't already, introduce the Hospital and Resident stable-matching context. Each year, the National Residency Match Program (NRMP) matches hospitals with medical students for their residencies, one of the final steps toward becoming a doctor. The assignment of residents to hospitals is based on the residents' rankings of the hospitals and on the hospitals' rankings of the residents. It is one of the early stable-matching problems that played a role in the development of the Gale-Shapley (G-S) algorithm.

In this case study, a highly simplified NRMP problem is used as a context. Later in this module students will learn more about added complications present in more authentic NRMP problems. For example, the matchings are not one-to-one because many residents can be assigned to the same hospital, and the presence of married residents who wish to be assigned to the same geographic area are two common complicating factors. But the case study is of a simple one-to-one stable-matching problem just as with the Big Siblings Little Siblings example. The change in context is meant to help students start thinking about other factors that could influence a stable-matching problem. For example, would you as a patient rather the hospitals or the residents be happier with their assignment?

When more than two stable matchings exist, the G-S algorithm will find only two. The two that the G-S algorithm will find are extreme solutions: one solution will be optimal for one set of people being matched (and least optimal for the other set of people) and the other solution will be optimal for the other set of people (and least optimal for the other set of people). If other stable matchings exist, they will lie between the two extremes and are very likely fairer overall than the extremes.

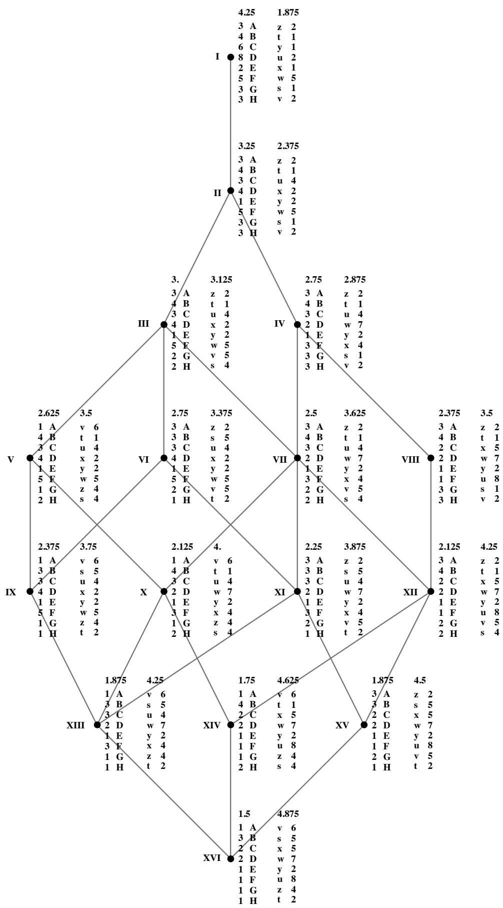
In this case, optimality is based on the simple average of a group's preferences in a given matching. The lower a group's average preference is, the better the matching is for that group. An average of 1 means that everyone in the group received his or her first choice. If there are six residents being matched to six hospitals, then an average of 6 means that everyone in that group received his or her last choice.

One of the factors that students must confront in this case study is the tension between what is good for the individuals of a group and what is good for the group as a whole. Consider matchings XI and XIV in the case study. The hospitals' average is 2.25 in XI and it is 1.75 in XIV. For the group, XIV is better than XI; however, note that hospitals B and H are assigned lower preferences in XIV than in XI. Hospitals B and H are less happy with matching XIV than they are with matching XI, but all the other hospitals are as happy or happier with XIV than they are with XI. Is it fair that B is reduced to its 4th choice and H loses its 1st choice so that four other hospitals can receive higher choices? There are many ways to approach this question.

Matching XI				Matching XIV			
2.25		3.875		1.75		4.625	
3	A	z	2	1	A	v	6
3	B	s	5	4	B	t	1
3	C	u	4	2	C	x	5
2	D	w	7	2	D	w	7
1	E	y	2	1	E	y	2
3	F	x	4	1	F	u	8
2	G	v	5	1	G	z	4
1	H	t	2	2	H	s	4

Poset: Partially Ordered Set

The diagram on p. 3 of the case study is of a partial order applied to a set, a *poset*. In the case of stable matchings the ordering principle is "at least as good as".



Consider the hospitals: a line connects two matchings if every hospital receives at least as good a partner in one matching as it does in the other matching. Matchings XI and XIII have a line between them because each hospital has the same or a better partner in XIII as it does in XI, (i.e., 1 ≤ 3, 3 ≤ 3, 3 ≤ 3, 2 ≤ 2, 1 ≤ 1, 3 ≤ 3, 1 ≤ 2, 1 ≤ 1). No line connects matchings XI and XIV because even though it is true that 1 ≤ 3 for hospital A, it is *false* that 4 ≤ 3 for hospital B; not every hospital fares as well or better in XIV as it does in XI, and vice versa.

Posets are transitive, so redundant lines are omitted. No line connects VIII to XIV even though XIV is a better matching than VIII. The line is omitted because 1) XIV is better than XII, and 2) XII is better than VIII, and therefore XIV must be better than VIII. Care must be taken not to assume that any matching is always better than one appearing higher in the diagram. For example, XV is not better than V even though XV appears two levels below V.

Finally, note the symmetry in the poset with respect to hospitals and residents. A line that connects two matchings always means the ordering holds for the hospitals and for the residents, but in the opposite direction. Following the line from VIII to XII we know that not only is every hospital at least as well off in XII as in VIII, but we also know that every resident is at least as well off in VIII as in XII. Following lines *down* the diagram takes us from worse to better matchings for the hospitals and from better to worse matchings for the residents. Similarly, following lines *up* the diagram takes us from better to worse matchings for the hospitals and from worse to better matchings for the residents.

This symmetry is a consequence of the stability requirement we imposed on the matchings. If it were possible to move from one matching to another matching and have both members of a pair (i.e., the hospital *and* the resident) receive higher preferences, then the first matching would have been unstable and should not have been in the diagram. Whenever we move from one stable matching to another stable matching, someone must be better off and someone must be worse off. In comparing one stable matching to another it can be the case that all four of the following statements are simultaneously true: i) a hospital receives a better partner, ii) a hospital receives a worse partner, iii) a resident receives a better partner, and iv) a resident receives a worse partner. An example is found in matchings III and IV. Those two matchings are not connected by a line. Lines only connect matchings in which “the same or better” applies to everyone in one group and “the same or worse” applies to everyone in the other group.

Posets can be subtly fascinating and sometimes counterintuitive structures.

Name:______________________________

Case Study: Which Stable Matching is Fairest of All?

Hospitals and medical students are keenly interested in stable matchings. The National Resident Match Program (NRMP) matches hospitals with medical students for their residencies. In 1998 the NRMP modified their matching algorithm because of an issue of fairness. Typically, there are many possible stable matchings and determining a "best" one means deciding whether the hospitals' preferences or the medical students' preferences should be given priority when comparing the options. In this case study we are going to assume that best means fairest. We want to treat the hospitals and the students equally. Deciding what is fairest involves working with large and complex mathematical structures. In 1962, David Gale and Lloyd Shapley initiated the formal study of stable matchings and some of their early questions remain unanswered. How will you define fairness?

You work for a Computational Consulting Company (come up with a better name than C3) that has been asked to determine which stable matching is fairest for a particularly complex problem facing a small country with eight hospitals and eight medical students. One student must be placed at each hospital and the matching must be a stable one.

Your task is to create a *measure of fairness* that can be used to compare many stable matchings and select the one that you believe is fairest. You will need to define your way of measuring fairness clearly enough so that another person could apply your method and get the same results. Different consulting companies will likely come up with different measures depending on the consultants' beliefs about what is fair.

The presidential cabinet of the country (led by your teacher) will decide which company's approach is best and will award a lucrative contract to the winning proposal. Be prepared to explain and defend your method of measuring fairness.

The data follow on the next two pages.

Name:___________________________________

Case Study: Fairest of All—Preference Matrices

Below are the preference matrices for the hospitals (A through H) on the left, and the residents (s through z) on the right.

Hospitals' Rankings of Residents

	s	t	u	v	w	x	y	z
A	5	8	7	1	4	6	2	3
B	3	4	6	2	5	7	8	1
C	4	1	3	5	8	2	6	7
D	7	6	8	3	2	4	1	5
E	3	5	8	6	7	2	1	4
F	8	4	1	6	5	3	7	2
G	3	6	8	2	4	5	7	1
H	2	1	7	3	4	5	6	8

Residents' Rankings of Hospitals

	A	B	C	D	E	F	G	H
s	8	5	6	7	2	3	1	4
t	7	1	6	5	4	8	3	2
u	3	6	4	2	7	8	1	5
v	6	7	3	8	1	4	5	2
w	1	8	6	7	3	5	4	2
x	6	3	5	2	1	4	7	8
y	3	4	1	5	2	7	8	6
z	2	6	8	3	7	5	4	1

The Gale-Shapley algorithm produces these two stable matchings:

Hospitals Propose

```
1.5    4.875

1 A    v 6
3 B    s 5
2 C    x 5
2 D    w 7
1 E    y 2
1 F    u 8
1 G    z 4
1 H    t 2
```

Residents Propose

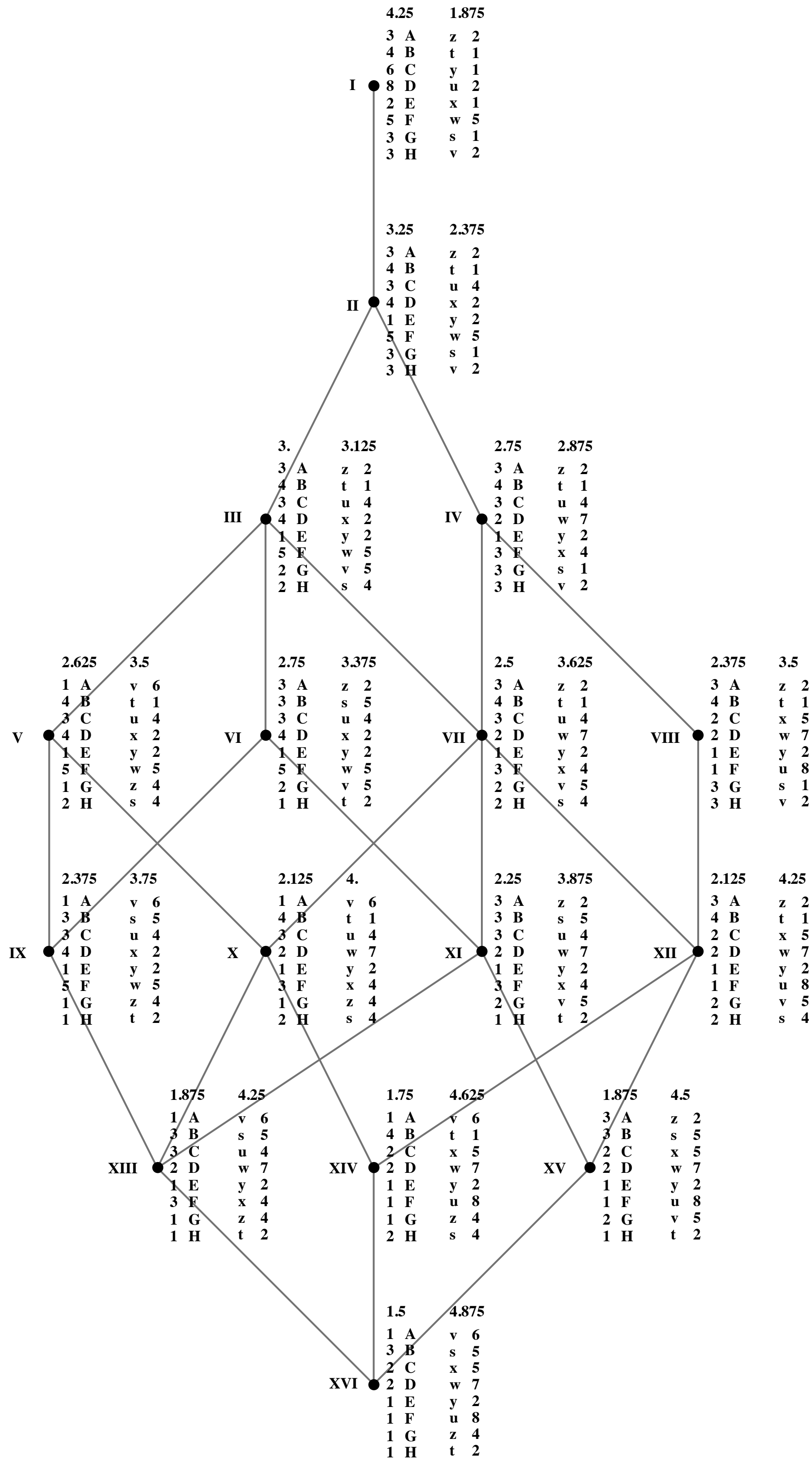
```
4.25   1.875

3 A    z 2
4 B    t 1
6 C    y 1
8 D    u 2
2 E    x 1
5 F    w 5
3 G    s 1
3 H    v 2
```

The best possible result for the hospitals has an average preference of 1.5 and the best possible result for the residents has an average of 1.875. But there are *fourteen* more stable matchings to consider! All sixteen stable matchings for the two preference matrices above are on the following page. Your task is to devise a way to determine which one is fairest overall. How would you measure fairness?

The stable matchings on the next page have been *partially ordered*. A line connects two matchings if every hospital (or resident) has at least as good a partner in one matching as it does in the other matching.

Case Study: Fairest of All—Stable Matchings Name:____________________________________

p. 3 of 4



Case Study: Fairest of All—Your Report Name:________________________________

On separate sheets of paper, your consulting company must respond to the following prompts.

1. Explain how to calculate your measure of fairness. Provide instructions that anyone could use on any set of stable matchings, and be sure to specify how any ties should be broken. Be complete, concise, clear, and precise.

2. Provide a worked example of how to use your measure of fairness to calculate the fairness value of one matching.

3. Provide the fairness values of all the matchings, I through XVI, and sort them from fairest to least fair.

4. Which stable matching do you think is fairest? Briefly explain why.

5. Be prepared to explain why your choice is fairest and why it is better than some of the other options.

Teacher Notes: **Discussion of Fairest of All Case Study Solutions**

When the Consulting Companies (small groups) have finished their work, ask each one to present their proposal for their fairness measure. Explain to the class that there are many different but equally acceptable interpretations of fairness. This is a problem with more than one solution and which solution a person favors will depend on that person's priorities. Before debating the various interpretations, take time to make sure the different measures of fairness are understood and that the principles or priorities that led to each particular measure are clear.

For example, a group might argue that, in addition to other criteria, the number of individuals receiving their least-preferred partners should be minimized. Or a group might argue that maximizing the number of individuals receiving their first choices is a priority. These two approaches, maximizing the good or minimizing the bad, will not always yield the same result. The two approaches alongside many others are all based on different principles that might only sometimes lead to the same conclusion.

After you and the presidential cabinet (perhaps colleagues or students from another class) have accepted one of the group's recommendations, you can steer discussion of the pros and cons of different measures of fairness to consider issues that might not have arisen in the group presentations and debate. Several issues are described below concluding with a very recent research finding proven by Christine Cheng in 2010. The topics of stable matchings and fairness are contemporary ones requiring new insights that could lead to widespread applications.

<u>Sample Student Responses</u> (may other legitimate ones are possible)
One common fairness criterion students propose focuses on the difference between the hospitals' average preference and the residents' average preference. Many claim that the matching with the smallest difference is the fairest one. In this particular case this criterion leads to two possible matchings, III and IV, with equally small

differences of .125. In this case the tie-breaking argument might be that matching IV is "obviously" better than matching III because both groups have lower average preferences: 2.75 < 3.0 for the hospitals and 2.875 < 3.125 for the residents. But do we really want to assign resident ***w*** her 7th choice?

In other cases, this particular minimum-difference criterion leads to even more problematic situations: it is possible for a matching to have a larger between-average difference than another matching and yet each group's average is equal to or lower than they are in the matching with the smaller between-average difference. This result strikes most people as counterintuitive.

Some students focus more on the individual hospital-resident pairings than on the group averages and argue that the fairest matching is the one that minimizes the sum of absolute values of the differences between pairs' assigned preferences. By this criterion, matching VI is fairest because it has the smallest sum of absolute differences: 11 (i.e., 1+2+1+2+1+0+3+1 respectively for A & z, B & s, C & u, etc.).

As students begin to compare and contrast different measures of fairness, distinctions between *local* issues of fairness and *global* issues of fairness emerge. For example, in matching VI no one receives lower than a 5^{th} preference whereas in matching III resident ***w*** receives her 7th preference. But in matching III, apart from the one 7^{th} preference, the next worst preferences assigned are three 4's while in matching VI there are four 5's. Is it fair to reduce rather sharply one person's level of satisfaction for the global good? This line of thinking leads students to propose minimizing the worst-case outcomes for the largest number of the individuals with the lowest preferences. This approach can also lead to problematic conclusions.

Contemporary Research Findings

One interesting fact about hierarchies of stable matchings, proven by Christine Cheng, enables us to accommodate simultaneously concerns about the local level of satisfaction and the global level of satisfaction. Consider the list of preferences received by all the individuals in all the stable matchings. For example, hospital D receives these sixteen preferences: {2,2,2,2,2,2,2,***2***,***2***,2,4,4,4,4,4,8}. The median of this list is 2. (In fact, given the even number of elements, the list has both a lower median of 2 and an upper median of 2.) Continue this process and calculate the median preferences of all the hospitals and all the residents. Might there be one stable matching in which all the hospitals and all the residents receive their median preference? Yes! In 2010, Cheng proved that such a median always exists, no matter what the preference matrices are like. This fact alone might be a good fairness criterion to use at the local (individual) level. Every hospital or resident receives its median preference of all the preferences that appear in the set of stable matchings. In our example, this simultaneous local-median matching is matching VII.

But it gets even better. Cheng also proved that this local-median matching is also always a *global*-median matching. Within the hierarchy of stable matchings, it will lie halfway between the two extreme matchings (in this case, I and XVI). In the case where the extremes are separated by an odd number of steps it will lie at the floor or ceiling of the halfway point. In our case, matching VII lies three steps, via lines in the hierarchy, from matching XVI and it lies three steps from matching I. Of the four matchings that lie at this halfway point (V, VI, VII, and VIII) matching VII is the only one in which all the preferences are median preferences.

So which matching is fairest? We've seen plausible arguments for III, IV, VI, and VII but other candidates exist! A close look at matching VIII reveals both the hospitals' and the residents' group averages are below those of matching VII and matching VIII is the matching that evenly distributes the largest number of 1st preferences: two for the hospitals and two for the residents.

As mentioned earlier, rather than minimizing the worst outcomes, some students argue for maximizing the best outcomes. But residents ***w*** and ***u*** are likely to protest as they are reduced to their 7th and 8th choices. There is room for further debate.

Unit 5: Assessment Case Studies: Variations on a Theme

Teacher Notes

This module concludes with three final case studies that all involve stable-matching problems at their core; however, each case study introduces a new complication to the simple stable-matching problems studied in Units 1 through 4. The goal of this final case study is to both assess student understanding of stable-matching problems and to assess their ability to transfer what they have learned to similar but slightly different problems. Motivate the students for this challenge by pointing out that rarely does one tool or one model work in every situation—good problem solvers learn to adapt what they've learned in one setting for use in a different setting. Seeing similarities and seeing the differences is the beginning of adapting what one knows to a novel situation. All learning is built out of the interaction between the familiar and the strange.

You can carry over the Consulting Company role-playing motif from Unit 4 using the same teams or forming new ones.

Students should be allowed some choice in which case study they will work on, or you can assign them randomly. Ideally, each case study will be worked by at least three different teams so that different problem-solving approaches can be tried and compared.

Although students could research these problems on the Web and possibly find algorithms or information about solution strateges, the goal is to have them develop their own methods and solutions.

Below are brief statements of case studies and their notable differences from the stable-matching problems seen previously in this module. The case studies follow.

Final Case Studies Synopses

In all case studies, the task is to develop a method for finding the fairest matching possible and to use the method to identify an optimal matching.

Case Study A: Hospitals Accepting Multiple Residents

In this case study, two preference matrices (one for the hospitals and one for the residents) are again used just as in the previous units, and the same requirement of stability applies. However, in this more realistic case study, each hospital can accept more than one resident; now, there is an additional constraint on the problem: each hospital must receive *at least one* resident but *no more than* their *resident capacity*. This case study is the least challenging in its difference from the simple stable-matching problem already studied.

Case Study B: Hospitals Accepting Multiple Married Residents

This case study adds another complication to the hospital/resident stable-matching problems found in Unit 3 and in Case Study A. In this case study not only can hospitals accept more than one resident but an additional constraint is that some married residents must be assigned to the same hospital. The matter is further complicated by the fact that the spouses might not rank the hospitals identically; these two new aspects of this case study are authentic aspects of realistic NRMP problems. This case study is more challenging than Case Study A.

Case Study C: Roommate Matching

This case study is significantly different from the other stable-matching problems in this module. In this roommate-matching problem, only one preference matrix is needed because the pairings are formed within one group of people. Everyone ranks everyone else. Although some students might think this makes the problem much easier and some will think it makes the problem much harder, in fact it is simply different and therefore easier for some and harder for others.

Case Study A: Hospitals Accepting Multiple Residents (2 Problems)

As medical students complete medical school they enter a residency during which they practice medicine in a hospital under the guidance of doctors. Typically, each medical student must complete one residency at one hospital; however, some hospitals can accept several residents. For example, a large city hospital might have a dozen or more different residents, while a small rural hospital might only accept one resident. This type of matching is a *many to one* matching because more than one item can be matched with another item.

Similar to previous stable-matching contexts, the members of both groups (the residents and the hospitals) each rank all the members of the other group. So there are two preference matrices in which the 1's indicate the top choices, 2's the second choices, etc. But now, the number of hospitals is not equal to the number of residents and each hospital now has a *capacity* that it would like to fill. The capacity is the number of residents it wants. How can we go about finding an optimal matching now?

Problem 1: 9 Residents & 5 Hospitals

Hospital Capacities: **v**: 1 **w**: 2 **x**: 2 **y**: 2 **z**: 3

Residents' Rankings of Hospitals

	v	**w**	**x**	**y**	**z**
A	3	1	2	5	4
B	4	1	2	5	3
C	3	2	1	4	5
D	1	2	3	5	4
E	1	4	3	5	2
F	2	3	1	4	5
G	4	3	1	5	2
H	1	3	4	5	2
I	1	2	3	4	5

Hospitals' Rankings of Residents

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
v	2	7	8	5	6	1	9	3	4
w	6	7	9	4	5	2	8	1	3
x	6	8	9	4	3	1	7	2	5
y	4	3	9	5	6	8	7	1	2
z	8	2	9	1	4	3	6	5	7

Problem 2: 12 Residents & 4 Hospitals

Hospital Capacities: **w**: 3 **x**: 4 **y**: 3 **z**: 2

Residents' Rankings of Hospitals

	w	**x**	**y**	**z**
A	1	2	3	4
B	2	1	3	4
C	4	2	3	1
D	1	3	2	4
E	2	1	3	4
F	2	1	3	4
G	1	2	4	3
H	4	2	3	1
I	1	3	2	4
J	4	2	3	1
K	1	2	3	4
L	2	1	4	3

Hospitals' Rankings of Residents

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**
w	12	9	1	11	8	7	10	2	5	6	4	3
x	11	10	3	12	5	7	8	1	6	9	4	2
y	11	10	1	12	6	5	8	2	9	7	4	3
z	12	10	3	11	6	5	9	1	7	8	4	2

Case Study B: Hospitals Accepting Multiple Residents & Spouses p.1 of 2

As medical students complete medical school they enter a residency during which they practice medicine in a hospital under the guidance of doctors. Typically, each medical student must complete one residency at one hospital; however, some hospitals can accept several residents. For example, a large city hospital might have a dozen or more different residents, while a small rural hospital might only accept one resident. This type of matching is a *many to one* matching because more than one item can be matched with another item.

An additional complication is that many married couples want to be placed at the same hospital. Sometimes each spouse in a couple ranks the hospitals identically; however, sometimes each spouse ranks the hospitals independently agreeing to go wherever they are assigned, even if the placement is slightly better for one spouse than the other. This complication might require you to redefine stability.

Similar to previous stable-matching contexts, the members of both groups (the residents and the hospitals) each rank all the members of the other group. So there are two preference matrices in which the 1's indicate the top choices, 2's the second choices, etc. Now, the number of hospitals is not equal to the number of residents and each hospital now has a *capacity* that it would like to fill. The capacity is the number of residents it wants. Furthermore, some of the residents must be assigned to the same hospital. How can we go about finding an optimal matching now?

Case Study B p. 2 of 2

12 Residents, 3 Married Residents, and 4 Hospitals

Hospital Capacities

W: 3 **X**: 4 **Y**: 3 **Z**: 2

The symbols ***** **#** and **^** indicate married residents (B & E, D & F, and J & L). Each member of a couple must be placed in the same hospital.

Residents' Rankings of Hospitals

	w	x	y	z
A	1	2	3	4
B*	2	1	3	4
C	4	2	3	1
D#	1	3	2	4
E*	2	1	3	4
F#	2	1	3	4
G	1	2	4	3
H	4	2	3	1
I	1	3	2	4
J^	4	2	3	1
K	1	2	3	4
L^	2	1	4	3

Hospitals' Rankings of Residents

	A	B	C	D	E	F	G	H	I	J	K	L
w	12	9	1	11	8	7	10	2	5	6	4	3
x	11	10	3	12	5	7	8	1	6	9	4	2
y	11	10	1	12	6	5	8	2	9	7	4	3
z	12	10	3	11	6	5	9	1	7	8	4	2

Case Study C: The Roommate Matching Problem

Not all matching-problems involve matching two separate groups of items. Consider a summer sports camp that houses the athletes in double rooms. Prior to arriving at the camp, the athletes view profiles of all the other athletes and provide their preferences. How should the camp directors assign roommates for an optimal matching?

Note that unlike previous case studies in this module, in this context there is just one preference matrix in which everyone ranks everyone else.

For example, reading across the third row we see that C's first choice is D, C's second choice is L, C's third choice is A, etc. And reading down the fourth column, we see that athlete D has been ranked first by A, B, C, and K.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**
A		2	4	1	3	6	5	8	7	9	10	11
B	4		2	1	3	7	10	9	5	8	11	6
C	3	5		1	4	6	9	7	8	10	11	2
D	1	2	3		6	8	10	9	4	7	11	5
E	3	4	11	2		1	8	10	5	7	9	6
F	7	4	3	5	8		1	2	6	10	11	9
G	7	11	4	2	1	8		3	9	5	10	6
H	9	11	10	8	1	6	7		3	2	5	4
I	5	3	4	2	1	9	6	10		7	8	11
J	9	5	6	10	2	11	8	4	7		3	1
K	9	8	4	1	2	3	10	11	5	6		7
L	4	5	6	7	8	11	10	1	9	3	2	

Final Case Study Report

Consulting Company: ______________________________

Names: ______________________________

Case Study: ______________________________

Due Date: ______________

Format for Written Report and Oral Summary

1. Explain your definition of optimality and fairness for this type of problem.

2. Explain your method for finding an optimally fair matching.

3. Give the matching you recommend be used and explain its pros and cons.

In addition to providing written responses to the above prompts, prepare to give a 5-minute oral summary. The summary should convince the organization that your approach and solution are the best.

Teacher Notes: **Guiding Work on Final Case Studies, Reports, & Presentations**

The final case studies will challenge different students in different ways. Encourage them not to worry about doing what is "right" but instead to focus on doing anything that makes sense to them. Remind them that are often different equally legitimate approaches to solving problems and, as they saw in Unit 4, there can be multiple correct answers that depend on what a group's priorities are.

Also encourage groups working on the same case study not to simply copy each other. More will be learned about the case studies if different groups approach the problems independently. A small prize can represent a contract to be awarded to the company with the best solution and it might motivate the groups.

If a group becomes very discouraged a few hints (below) could help them persevere; however, the most rewarding engagement with the case studies will come from students relying upon their own creative problem-solving skills.

<u>Hints</u>

Case Study A can be approached in a way that makes it much more like the first Hospital/Resident Case Study in Unit 4. Because some hospitals can accept more than one resident it is as if they are functioning as multiple hospitals with identical rankings of the residents. The hospitals' preference matrix can simply be expanded adding duplicate rows as needed. See expanded matrix below. The "duplicate" hospitals play similar roles in the Gale-Shapley proposal algorithm; however, they do not compete with themselves for residents. So if hospital z1 is already engaged to a resident, hospital z2 skips that resident and proposes to the next one on the list. There is still the complexity in Problem 1 of Case Study A of figuring out how to decide which resident position does not get filled, since there are more residency positions than there are residents.

Hospitals' Expanded Preference Matrix Case Study A

	A	B	C	D	E	F	G	H	I
v	2	7	8	5	6	1	9	3	4
w1	6	7	9	4	5	2	8	1	3
w2	6	7	9	4	5	2	8	1	3
x1	6	8	9	4	3	1	7	2	5
x2	6	8	9	4	3	1	7	2	5
y1	4	3	9	5	6	8	7	1	2
y2	4	3	9	5	6	8	7	1	2
z1	8	2	9	1	4	3	6	5	7
z2	8	2	9	1	4	3	6	5	7
z3	8	2	9	1	4	3	6	5	7

In Case Study B, the capacity of hospitals to have more than one resident can be approached with an expanded hospital preference matrix as in Case Study A. And working with the married residents can be a bit easier if they think about each spouse proposing and accepting proposals independently but remembering that any change in one spouse's assignment immediately affects the other spouse. Still, handling the married residents when their preferences don't coincide requires the students to create a method for resolving dilemmas faced by a couple. In one sense it adds another level to the fairness analysis. In Unit 4 students were introduced to the notions of local- and global-median solutions. Similar thinking can be applied to the options a married couple might face, resulting in a micro-level assessment of what is optimal. The statement of the problem suggests a solution: that a couple will go wherever either of them receives the highest preference.

Case Study C will seem easy to some students until they realize that stability might not be possible. Reassure them that this context is different and might require different definitions of stability and fairness than were used in other case studies.