Clock Buddies

Debra K. Borkovitz, Wheelock College

The Problem:

This is one version of the problem we have been studying in class: Construct a "clock buddy" schedule for twelve people with the following conditions:

1) Buddies meet on the hour, with the first meeting at 1:00 (then 2:00, 3:00, etc.)
2) At each time, every person meets with exactly one other person (his or her "clock buddy" for that time period).
3) Two people can only be clock buddies during one time slot, no more.

The ultimate goal is to construct a schedule in which every possible pair of clock buddies meet, but a first goal is to construct a schedule for six times, on the hour, from 1:00 to 6:00, for each person.

Another goal is to find an algorithm or algorithms for constructing schedules with different numbers of people and time slots (an algorithm is a step-by-step method that will work to solve a variety of similar problems).

Some Student Thinking About the Problem:

Below is the thinking of some (fictional) students about this problem, based on algorithms developed by my students in previous years, as well as some exercises based on their thinking. Here are some questions to think about as you read each student's work:

1) Do I understand the algorithm?
2) How effective is the student's representation?
3) Will the algorithm extend to other problems?
4) Can I improve the student's algorithm?

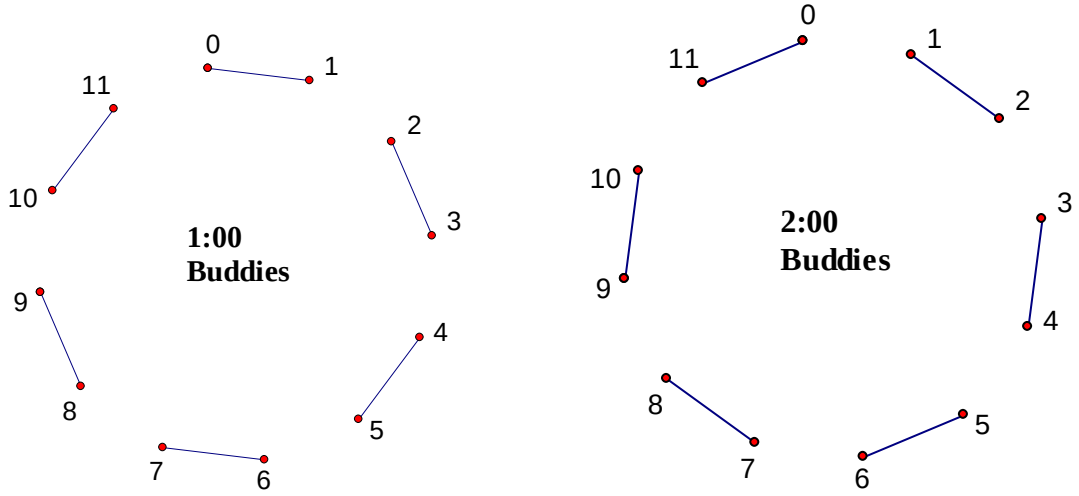
Corianne: The Circle Algorithm

In our class we first tried to do the problem randomly, just walking around and asking people if they had a slot open, and that didn't work, so we decided to start over. All twelve people stood in a circle. For 1:00 we made appointments with the person next to us on one side, and for 2:00 we made appointments with the person on the other side. Then for 3:00 we made appointments with the person who was one over on one side, and for 4:00 with the person one over on the other side. We did the same thing again for 5:00 and 6:00 with the people two over on each side, so we solved the original problem, but

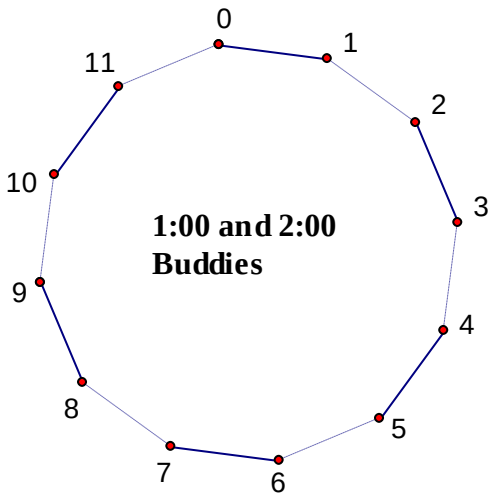
then when we tried to schedule 7:00 this way, it got really confusing and didn't seem to work – two people both thought they should meet with me, and when I picked one, the other didn't have anyone to meet with.

Stop and see if you can reconstruct Corianne's method – either with a group of people or by representing it on paper.

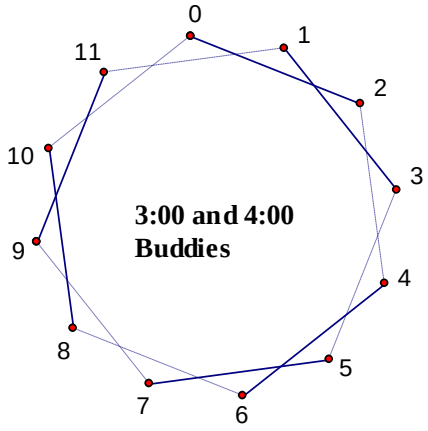
The teacher suggested we try to represent what we were doing on paper, and also that we name our algorithm; we decided to call it "The Circle Algorithm." Originally, we wrote down everyone's names, but that was too cumbersome, so we decided to use numbers. The teacher also suggested that we start with 0 instead of with 1; she said it might be easier to see patterns this way. Here are the appointments for 1:00 and 2:00; at 1:00, 0 and 1 meet, 2 and 3 meet, etc.



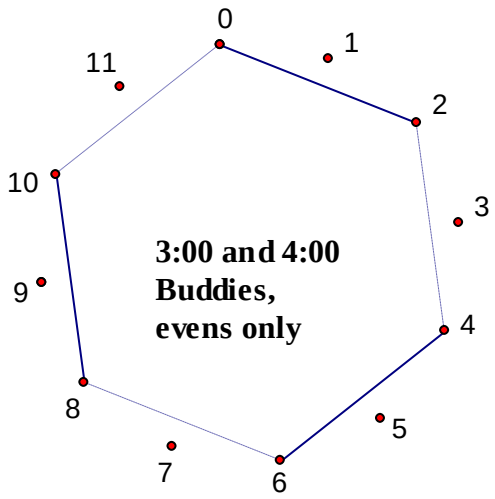
We noticed that looking at them together, the 1:00 and 2:00 clock buddies form a ring, where the 1:00 appointments are solid lines and the 2:00 appointments are dashed lines:

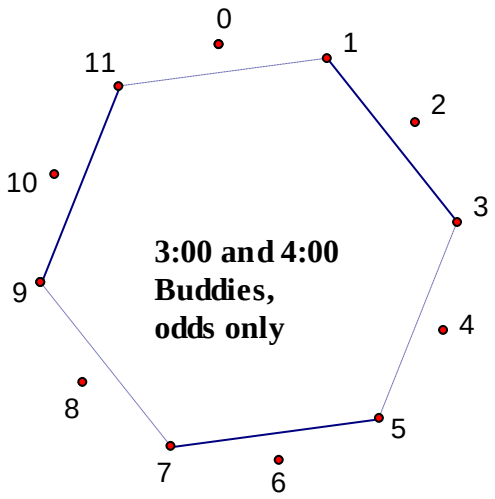


Then we made the 3:00 and 4:00 appointments together, with the 3:00 appointments as solid lines and the 4:00 ones as dashed lines. They form two hexagons:

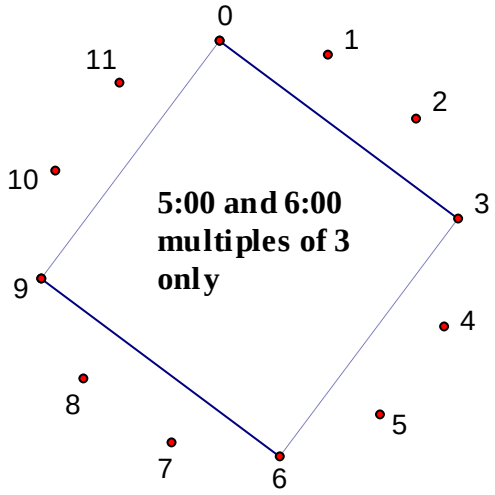


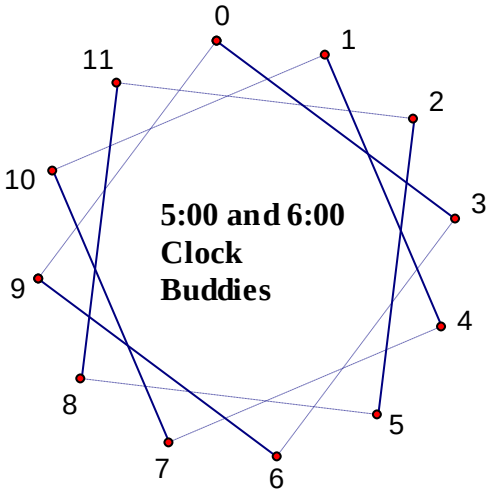
It's a little easier to see the appointments when the odds and evens are separated:



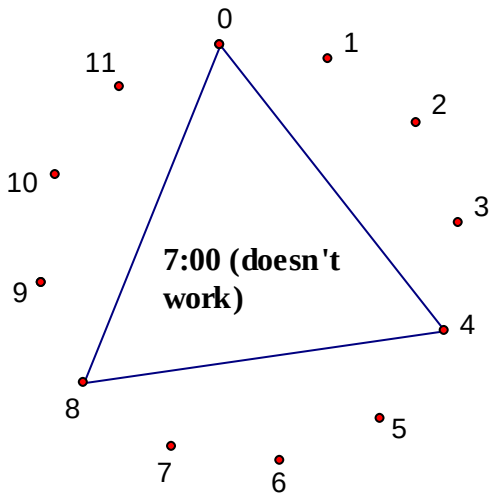


The 5:00 and 6:00 appointments make three squares. We noticed that one of these squares connected multiples of three. The first diagram shows one square, and the second shows all three squares together:

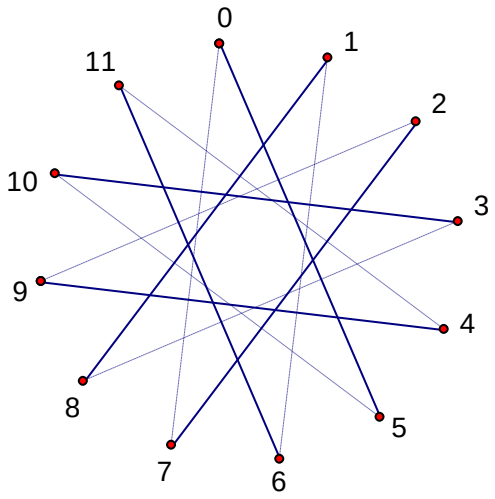




Now is where we had the problem. When we tried to make clock buddies for 7:00, this time skipping three people, we ended up with triangles. Here's one of them:



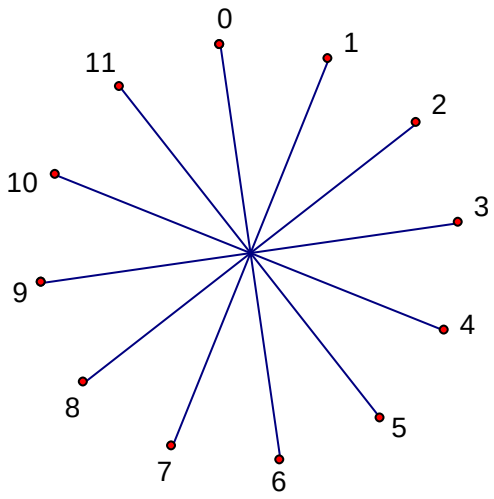
If person 0 is a clock buddy with person 4, then person 8 has no buddy. This is where we got stuck when we were working in a circle of people, but on paper we decided to try skipping 4 people, since skipping 3 people didn't work. We connected 0 to 5 and keep skipping 5 to see what happens. We got a design like a spirograph, and it works for scheduling clock buddies:



7:00 and 8:00 Clock Buddies

We just alternated solid and dashed lines. So the solid lines represent the 7:00 appointments, which are 0-5, 3-10, 1-8, 6-11, 4-9, and 2-7. The 8:00 appointments are 0-7, 2-9, 4-11, 1-6, 3-8, and 5-10. We know that none of these appointments are duplicates because everyone always meets with someone who is five people away from them, and in the previous times, people met with people who were closer in the circle.

Moving on to 9:00, we skip 5 people, and whether we go to the right or to the left, each person gets the same clock buddy.



9:00 Clock Buddies

This is all we can do, because if we skip 6 people, we'd connect 0 to 7, which would be a repeat of the clock buddies for 8:00 that we already did, and we'd go backwards. Also, for example, the only people who haven't had person 0 as a clock buddy are persons 4 and 8, and they make a triangle, similar to the one we tried for 7:00. There's no way to schedule them.

Exercises:

1) Try the Circle Algorithm with some other numbers of people, including small numbers such as 4, 6, and 8. What patterns do you notice? Try to construct an argument that will convince a skeptical person that your patterns will continue for scheduling larger groups.

2) How many clock buddies could we schedule for our class of 22 using the Circle Algorithm? Can you modify the algorithm to schedule more clock buddies?

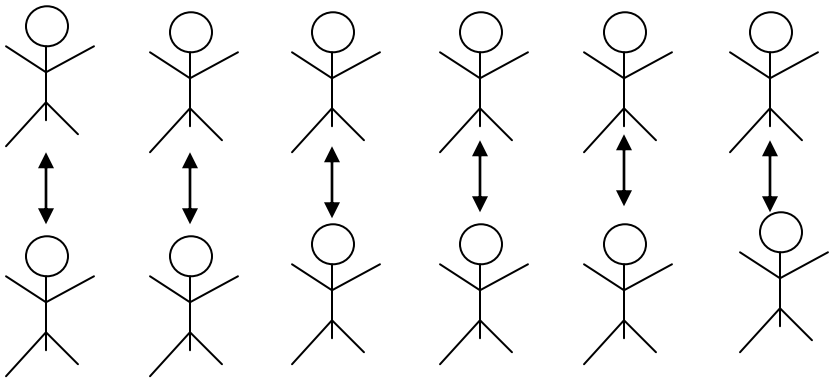
3) When will The Circle Algorithm produce a complete schedule of clock buddies (i.e. every person is a clock buddy with everyone else)? Again, work on constructing a convincing argument that your answer is correct and complete (or explain why you're not sure that it is).

4) Find a method for quickly predicting the number of clock buddies that The Circle Algorithm will produce for any given (even) number of people. Again, work on constructing a convincing argument that your answer is correct and complete (or explain why you're not sure that it is).

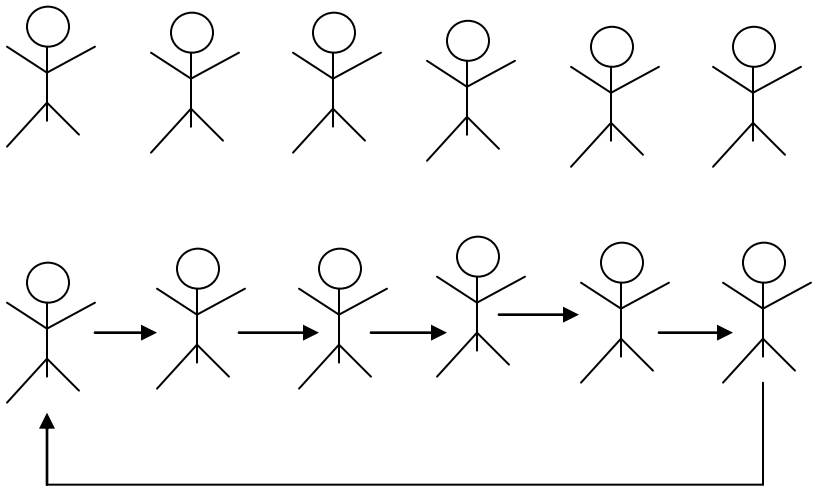
5) What is the mathematical name for polygon formed in the 1:00 and 2:00 appointments (Corrine calls it a "ring.")

Monique's Method:

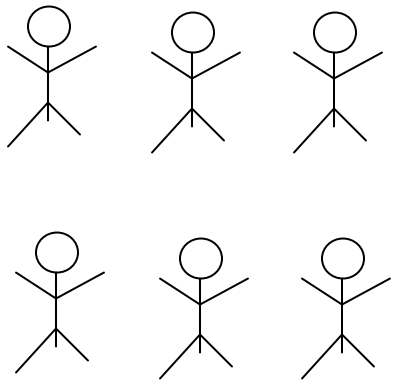
We decided to divide the class into two lines like this:



We first scheduled the 1:00 appointment with the person right across from us. Then, we had the bottom line rotate to the right, while the top line stayed still. The person in the rightmost place moved to the left:



We called this the "Two Lines Algorithm," and we scheduled clock buddies from 1:00 to 6:00, just by having one line keep shifting over one, while the other stayed still. We solved the original problem pretty easily. To continue, we realized that everyone had met with all the people in the other line. So, then we split each line in half and did the same thing, this time with lines of three:



We were able to schedule three more clock buddies this way, but then we couldn't get any more, because then there were three people left in each line, and they could only schedule clock buddies with the people in their own line, but we couldn't split a line of three into two equal groups.

Exercises:

1) Try the Two Lines Algorithm with some other numbers of people, including small numbers such as 4, 6, and 8. What patterns do you notice? Try to construct an argument that will convince a skeptical person that your patterns will continue for scheduling larger groups.

2) How many clock buddies could we schedule for our class of 22 using the Two Lines Algorithm? Can you modify the algorithm to schedule more clock buddies?

3) When will The Two Lines Algorithm produce a complete schedule of clock buddies (i.e. every person is a clock buddy with everyone else)? Again, work on constructing a convincing argument that your answer is correct and complete (or explain why you're not sure that it is).

4) Find a method for quickly predicting the number of clock buddies that The Two Lines Algorithm will produce for any given (even) number of people. Again, work on constructing a convincing argument that your answer is correct and complete (or explain why you're not sure that it is).

5) Both The Circle and The Two Lines algorithms scheduled nine clock buddies for each person, starting with twelve people. Do these two methods <u>always</u> produce the same number of clock buddies for each person, when we start with the same number of people? Why or why not? (if you started with this algorithm, you can wait to answer this question until you've also looked at the Circle Algorithm).

Zaskya's method:

In our class we had eleven students and the teacher. The teacher said we could make appointments with her, but she wouldn't really participate. We labeled the students A through K and called the teacher T. What we did was go around and fill up the schedules one at a time, so that people were clock buddies with the others in alphabetical order, but when they got to themselves, they met with the teacher instead. This method worked! Everyone got a full schedule.

Person A filled in her schedule like this:

Time	Meet
1:00	B
2:00	C
3:00	D
4:00	E
5:00	F
6:00	G
7:00	H
8:00	I
9:00	J
10:00	K
11:00	T

Then it was person B's turn. He was already meeting with A at 1:00. Since B comes after A, he would be a clock buddy with himself at 2:00, but instead, he was a clock buddy with the teacher then. Then at 3:00 he was clock buddies with C, at 4:00 with D, etc. His schedule looked like this:

Time	Meet
1:00	A
2:00	T
3:00	C
4:00	D
5:00	E
6:00	F
7:00	G
8:00	H
9:00	I
10:00	J
11:00	K

Person C started with her schedule filled in like this:

Time	Meet
1:00	
2:00	A
3:00	B

At 4:00, she would meet with herself, so she meets with the teacher instead, and then at 5:00 she meets with person D. At 11:00, she meets with person J, and then she wraps around to meet with K at 1:00. Her schedule looks like this:

Time	Meet
1:00	K
2:00	A
3:00	B
4:00	T
5:00	D
6:00	E
7:00	F
8:00	G
9:00	H
10:00	I
11:00	J

Person D's schedule looks like this:

Time	Meet
1:00	J
2:00	K
3:00	A
4:00	B
5:00	C
6:00	T
7:00	E
8:00	F
9:00	G
10:00	H
11:00	I

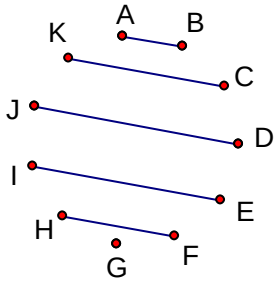
We called our algorithm the "Skip Algorithm" and demonstrated that it worked by acting out the appointments: at 1:00, the 1:00 people all got together, etc. The teacher challenged us to find different ways to show the schedules all together. We came up with a few different representations.

The first was the chart below, which shows what times each pair are clock buddies. For example, if you look at row C and column E (or column C and row E), you can see that C and E meet together at 6:00. If you check G and G (or any other two of the same letters), you see a 0, which means people don't meet with themselves. The teacher said that this chart is called a *Latin Square*, which means that every row and column contains each number once (since each person is clock buddies with every other person once). It is also a *symmetric* Latin Square, which means that the first row is the same as the first column, the second row is the same as the second column, etc. It needs to be

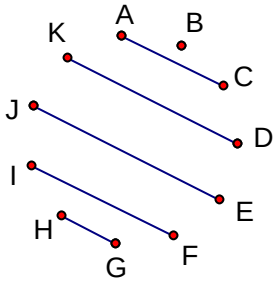
symmetric because when A is a clock buddy with B, of course B is also a clock buddy with A.

	A	B	C	D	E	F	G	H	I	J	K	T
A	0	1	2	3	4	5	6	7	8	9	10	11
B	1	0	3	4	5	6	7	8	9	10	11	2
C	2	3	0	5	6	7	8	9	10	11	1	4
D	3	4	5	0	7	8	9	10	11	1	2	6
E	4	5	6	7	0	9	10	11	1	2	3	8
F	5	6	7	8	9	0	11	1	2	3	4	10
G	6	7	8	9	10	11	0	2	3	4	5	1
H	7	8	9	10	11	1	2	0	4	5	6	3
I	8	9	10	11	1	2	3	4	0	6	7	5
J	9	10	11	1	2	3	4	5	6	0	8	7
K	10	11	12	2	3	4	5	6	7	8	0	9
T	11	2	4	6	8	10	1	3	5	7	9	0

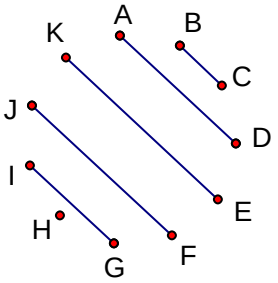
The next representation we came up with was visual and gave us a whole different way of looking at the problem. We drew lines connecting clock buddies at each time; the person who was left out met with the teacher. There were lots of patterns.



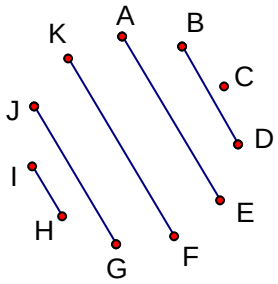
1:00 Clock Buddies



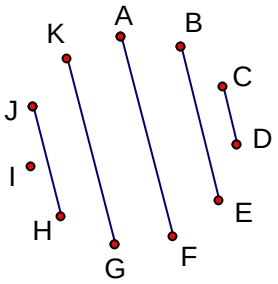
2:00 Clock Buddies



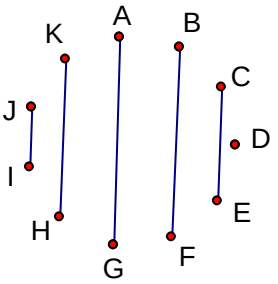
3:00 Clock Buddies



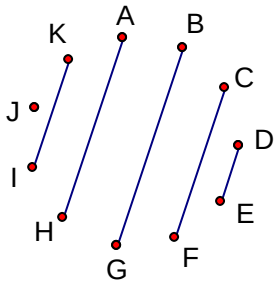
4:00 Clock Buddies



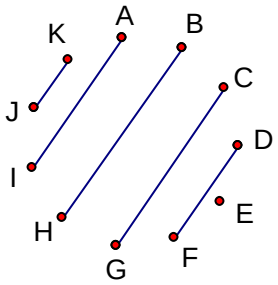
5:00 Clock Buddies



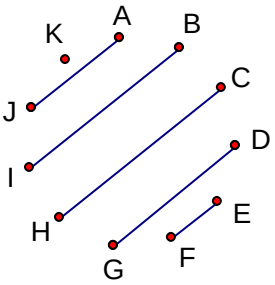
6:00 Clock Buddies



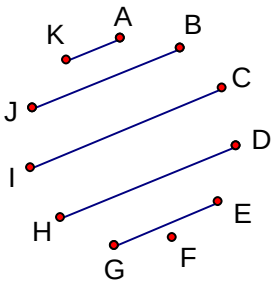
7:00 Clock Buddies



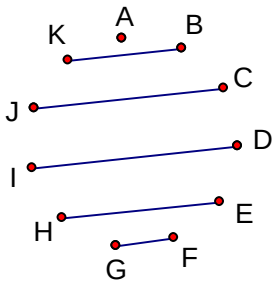
8:00 Clock Buddies



9:00 Clock Buddies



10:00 Clock Buddies



11:00 Clock Buddies

For our last representation, we just listed times and clock buddies for each time, such as 1:00: A-B, C-K, D-J, E-I, F-H, and G-T. We also tried using numbers to represent the students, rather than letters, to see if we could find any patterns. We made A=1, B=2, up to K=11, and we left the teacher as T. Here is the schedule:

1:00	1-2,	3-11,	4-10,	5-9,	6-8,	7-T
2:00	1-3,	4-11,	5-10,	6-9,	7-8,	2-T
3:00	1-4,	2-3,	5-11,	6-10,	7-9,	8-T
4:00	1-5,	2-4,	6-11,	7-10,	8-9,	3-T
5:00	1-6,	2-5,	3-4,	7-11,	8-10,	9-T
6:00	1-7,	2-6,	3-5,	8-11,	7-10,	4-T
7:00	1-8,	2-7,	3-6,	4-5,	9-11,	10-T
8:00	1-9,	2-8,	3-7,	4-6,	10-11,	5-T
9:00	1-10,	2-9,	3-8,	4-7,	5-6,	11-T
10:00	1-11,	2-10,	3-9,	4-8,	5-7,	6-T
11:00	2-11,	3-10,	4-9,	5-8,	6-7,	1-T

We also found many patterns in these numbers.

Exercises

1) Try the Skip Algorithm with some other numbers of people, including small numbers such as 4, 6, and 8. Form all the representations given above (and more if you can think of them). Try starting with different representations. What are the strengths and limitations of each representation? Which do you find easiest to use to construct a schedule? Do others in your study group agree or disagree with your choice?

2) What patterns do you notice in each of your representations above (and in the ones given in the text and in any new ones you create)?

3) How many clock buddies could we schedule for our class of 22 using the Skip Algorithm?

4) When will The Skip Algorithm produce a complete schedule of clock buddies (i.e. every person is a clock buddy with everyone else)? Again, work on constructing a convincing argument that your answer is correct and complete (or explain why you're not sure that it is).

5) After you have studied all three algorithms, compare the advantages and disadvantages of them all. Which do you like most? Why? Which is simplest to explain? Which is simplest to use? Which is most powerful?