UNIVERSITY OF
WATERLOO

Faculty of Mathematics
Waterloo, Ontario N2L 3G1

Centre for Education in
Mathematics and Computing

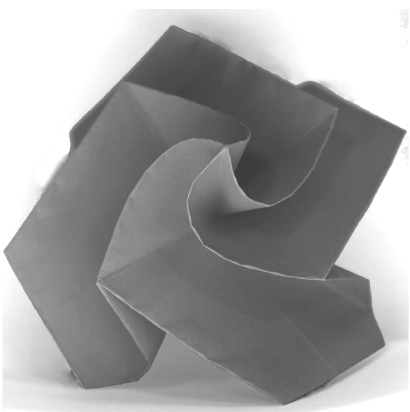
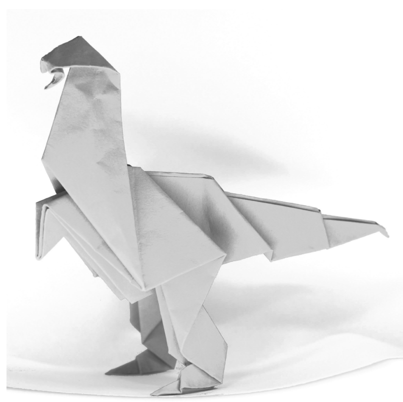
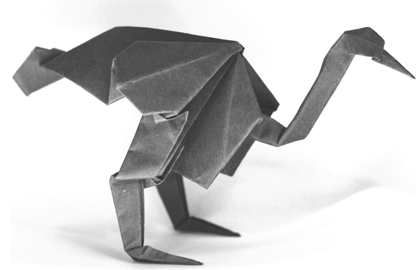
Grade 6 Math Circles

October 6/7, 2015

Algorithms

Origami

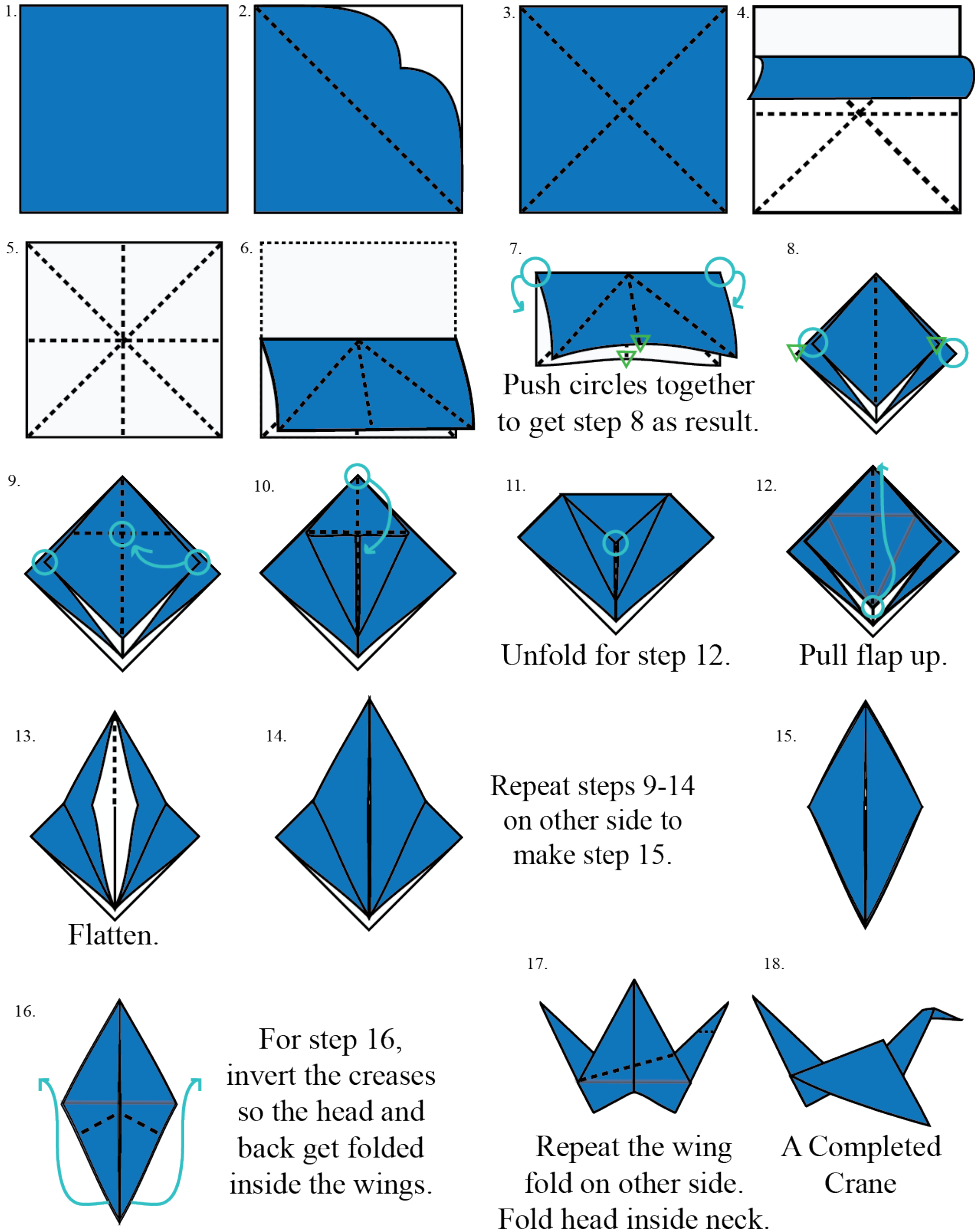
Origami is the art of paper folding. There are many forms of origami which allow for different rules to be applied. Kiragami is a form of origami that also allows for cutting of paper. Traditionally, Origami is very strict and does not allow cutting or gluing. Even so, there are many amazing creations that can be folded from a single piece of paper. Here are three examples, a rose, a dinosaur, and an ostrich.

They are all different objects but underlying their construction is the same shape! The method to obtain this basic shape, or base, is the same for each structure. There is an algorithm, or a collection of rules to be followed in some order, for folding this base. Let us call it the bird base. Try following these steps on your own to construct it. The completed bird base is step 15 and the remaining steps construct an origami crane.

If you get stuck, watch this video: `https://youtu.be/vS9VJcEqpnE`.

How to Fold an Origami Crane



What are Algorithms for?

Algorithms are everywhere. Whether you know it or not, you use algorithms on a daily basis. Do you follow the same routine every morning to get ready for school? Then you are following an algorithm. Your parents follow algorithms to cook. The way we keep track of time can be explained in an algorithm. Algorithms take many forms, anything from a list of pictures or symbols to language or mathematics can be used to make algorithms. Anything that can describe a procedure to solve a problem. However, in mathematics they take an extremely precise form. Whereas, in life, one might forget something in their morning routine, or add a pinch of salt to a cooking recipe, in mathematics an algorithm is followed very strictly.

Think about how to add two-digit numbers together. When we add two numbers, usually we place one above the other and add each of their columns together. Recall that these columns are called place values and have names, like the ones column, tens column, hundreds, and so on. So to add two-digit numbers we add the ones column and then the tens column. This is the sort of structure I want to draw your attention to, we can use it to create an algorithm.

Now adding two digit numbers is easy! Just sum the numbers in the ones column and then sum the numbers in the tens column. Perfect, these are the steps that form our algorithm and now we should test them on a few examples:

$$
\begin{array}{lrrr||lrrr||lrrr}
\text{(a)} & & & 9 & \text{(b)} & & 1 & 0 & \text{(c)} & & 1 & 8 \\
 & & + & 9 & & + & 1 & 8 & & + & 2 & 7 \\
\hline
 & & 1 & 8 & & & 2 & 8 & & & 3 & 15
\end{array}
$$

Wait. $18 + 27$ is not 315. But I followed each step of our algorithm perfectly: I added the numbers in each of the columns together. How can our algorithm produce a wrong answer?

I forgot something and now our algorithm has an error. Sometimes algorithms will produce errors and that is why we should always test a few examples, even if we believe our algorithms are perfect — it could be that only a single example breaks it! How do we correct our algorithm? We should write it out again in a list:

1. Add the digits in the ones column.
2. If their sum is greater than 9,

   then we need to carry the tens digit from the sum.
3. Add the digits in the tens column and the amount we carried.

Step two in our algorithm is what we will call a *comparison step*. This is because it compares two items, in this case the sum of the ones column and a 9. We could make step two shorter: Carry the tens column digit of the sum of the ones column. Why? Well think about if our ones column summed to 8. Then the tens digit in 8 is nothing, or in other words zero. Doing this removes the comparison part of the step.

Exercise: The word generalize means to make something more widespread. Our algorithm only adds two digit numbers with a ones and tens column. Can you generalize our algorithm to something that works for more than just two digit numbers? Try three digits? What about any number of digits?

Here is an algorithm for adding any two numbers:

- For each place value:
- Add the entire column and write the ones digit in the same column below a line. Carry the remaining numbers of the column's sum.
- Repeat for the next place value, for each new place value always add the carry over amount.

Exercise (for home!): Think about how to subtract (with borrowing). Write a set of steps so someone who does not subtract by borrowing can learn. For a challenge, try doing the same for multiplication and division. Solutions may vary.

Sorting

Algorithms solve problems and computer scientists have a lot of problems. We live in an age of information; billions of people own computers and for these computers to be fast they must be able to handle a lot of data. Data is anything that can be collected and measured. Lists of words are data, such as a dictionary; and so are lists of numbers, such as ages from a survey. It is sometimes difficult to use data if is not sorted. How would we find words in a dictionary if it were not for the alphabetical ordering? Therefore, it is useful for computers to be able to sort data and sort *a lot* of data. This problem has lead to the invention of many sorting algorithms but we will explore only two today. The first is a very natural way to sort items that you may already use. (For the second, see question 5 of the Exercises.)

Insertion Sort

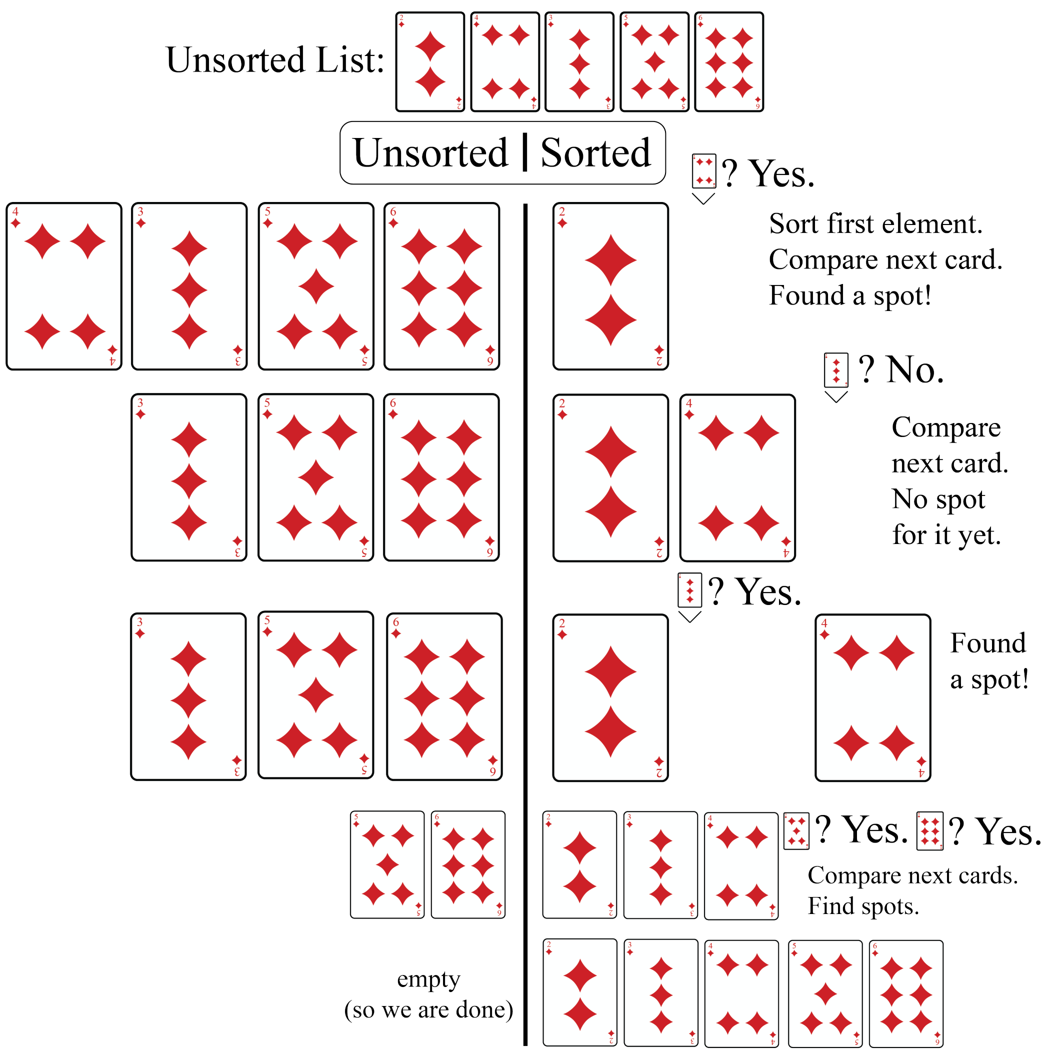
Insertion sort builds a sorted list from an unsorted list. It pulls an element, one at a time, from the unsorted list and then compares it to the sorted elements until a spot is found.

It is worthwhile to point out that insertion sort works for any list that has a nice way to order itself. Alphabetical order is one type of ordering that is nice. Any ordering that can order all of the elements in a list with any other element in the list will be nice. For example, one ordering that is not as useful is by smell. If I had a banana, apple, and orange to sort in order of smell, how would I do it? I could order them by which I think smells the best but someone else might disagree with me. Also, what happens when I do not know what a fruit smells like? Where would you place dragon fruit in this list? Numbers are much easier to order and we can compare them by asking which number is lesser (or greater) than another.

We will only sort whole numbers for now so that we do not have to worry about weird orderings. Also, our algorithm will always sort numbers from least to greatest. So when we compare them, we only need to ask one question: Is this number greater than that one? Here is an algorithm for insertion sort:

1. Begin with two lists: Unsorted (full to begin) and Sorted (empty to begin). Sort the first number of the unsorted list (easy: a list of only one number is already sorted).
2. Pull another number from the Unsorted list and compare it to all of your sorted numbers from greatest to least (moving backwards in your list).
3. At every sorted number ask the question: Is this unsorted number larger than the sorted one? If the answer is yes, put the greater number to the right of the lesser one. If you come to the start of your sorted list then the number is the smallest so far so put it at the front.
4. Continue this process until you run out of unsorted numbers.

Consider this simple example with playing cards. While trying to read it, review the steps in the algorithm and make sure it follows them.



The last two steps for sorting 5 and 6 are straight forward since we will say 'yes' immediately; they are identical to the first step. We began with a list that was almost sorted but if you begin with a messier list, you can expect the algorithm to repeat more times.

This example uses the unsorted list $(9, 4, 3, 6, 7)$. Each column has the current unsorted list, the step we do, and the result on our sorted list. We slowly build up our sorted list and shrink our unsorted list. Read the table from left to right.

Unsorted	Orders	Sorted
9,4,3,6,7	Sort the first element.	9
4,3,6,7	Is 4 greater than 9? No. Move on.	9
4,3,6,7	No more elements to compare 4 to. Insert 4 at start.	4,9
3,6,7	Is 3 greater than 9? No. Move on.	4,9
3,6,7	Is 3 greater than 4? No. Move on.	4,9
3,6,7	No more elements to compare 3 to. Insert 3 at start	3,4,9
6,7	Is 6 greater than 9? No. Move on.	3,4,9
6,7	Is 6 greater than 4? Yes. Insert 6 on right.	3,4,6,9
7	Is 7 greater than 9? No. Move on.	3,4,6,9
7	Is 7 greater than 6? Yes. Insert 7 on right.	3,4,6,7,9
empty	Unsorted list is empty. We are now done.	3,4,6,7,9

Games

The Number Factory

Algorithms can have input and output. In other words, we give them *something* they can use and they give us *something* we want. They do not need input or output but for the next exercise they do. Below are columns, one is input and the other is output. You have to write a set of instructions to turn each input into the output on the same row. These instructions are an algorithm. Try figuring out the algorithms for the first three questions using only addition, subtraction, multiplication, and division:

(a)	Input	Output	(b)	Input	Output	(c)	Input	Output
	1	6		2	10		7	21
	2	11		50	154		15	21
	3	16		101	307		21	21
	4	21		31	97		28	21
	9	46		1	7		34	21

(a) Calculate input multiplied by 5 and then add 1.

(b) Calculate input multiplied by 3 and then add 4.

(c) Either (results may vary):

- Calculate input divided by input and then add 20.
- Calculate input divided by input and then multiply by 21.
- (Calculate nothing and simply) Output 21.

We do not need to use only math in our algorithms. Try the next three exercises. Some use math and others do not.

(d)	Input	Output	(e*)	Input	Output	(f)	Input	Output	(g)	In	Out
	20	0		50	2, 5		Tim	Ujn		7	G
	100	8		12	2, 3		Ann	Boo		20	T
	180	16		5	5		Spy	Tqz		N	14
	340	32		6	2, 3		Jim	Kjn		25	Y
	60	4		4	2		Add	Bee		3	C

(d) Calculate input divided by 10 and then minus 2.

(e*) Either:

- Divide input by 2,3, and 5 and list only those that go nicely into input. If none do, put null.
- Compute the prime factors of input. If none, put null.

(f) Input a word and for each letter output the letter after it in the alphabet. I.e. ABC = BCD.

(g) For each number under 26, find the corresponding letter in the alphabet. I.e. 1=A, 2=B, and so on.

Exercise: Come up with your own algorithms and then list only their inputs and outputs. Turn to the person sitting next to you and ask them if can find the pattern.

Input	Output		Input	Output		Input	Output		Input	Output

Four is the Cosmic Number

Try determining the pattern in these number sequences:

11, 6, 3, 5, 4, 4, 4, 4, ...

31, 9, 4, 4, 4, 4, ...

25, 11, 6, 3, 5, 4, 4, ...

This pattern is often spoken out loud instead of written down. You should read the above sequences out loud as:

11 is 6, 6 is 3, 3 is 5, 5 is 4, 4 is 4, and four is the cosmic number.

31 is 9, 9 is 4, 4 is 4, and four is the cosmic number.

25 is 11, 11 is 6, 6 is 3, 3 is 5, 5 is 4, 4 is 4, and four is the cosmic number.

Hints

The pattern can be explained algorithmically. Start with an input number and come up with a pattern that works no matter what number you give it as input.

Another example is 12: 12 is 6, 6 is 3, 3 is 5, 5 is 4, and 4 is the cosmic number.

This type of algorithm is called recursive. You can take your previous output and make it input in the next step.

It helps to spell the words for the numbers.

Determine the pattern yet? See the solutions for an answer.

When you write the words for the numbers, count the number of letters in each word. For example, twelve has six letters, six has three letters, three has five letters, five has four letters, and four has four letters so it is the cosmic number since it will only ever be four after this point.

Exercises

1. The Fibonacci Sequence is another famous number sequence. Numbers from the sequence and the sequence itself are often found in nature used, we think, for specific reasons. Perhaps this is because the sequence can be generated algorithmically. Here are the first six numbers: $1, 1, 2, 3, 5, 8, \ldots$

   (a) Determine the next three numbers in the sequence.

   $13, 21, 34$

   (b) What is the pattern of this sequence? Explain.

   Start with $1, 1$ and always add the previous two numbers to get the next number. $1 + 1 = 2$, $1 + 2 = 3$, $2 + 3 = 5$, and so on.

   (c) Can you write an algorithm with precise steps to generate this sequence?

   Begin with two numbers. Add those two numbers to get the next number. Always add the most recent two numbers in the sequence to get the next number.

   (d) Would your algorithm work if I wanted to start with 1 and 3 instead of 1 and 1?

   See solution to previous question.

2. Answer the questions about the following algorithm:

   - Input: Two Numbers.
   - Calculate: Add them. Divide by 2. Output result.

   (a) Test the algorithm out on these numbers: $(4, 2)$, $(3, 7)$, $(10, 12)$, and $(2, 1)$. Answers in order: 3. 5. 11. 1.5

   (b) What does the algorithm do? In what situations would this algorithm be useful? The algorithm calulates both the mean between two numbers or the midpoint between two numbers.

(c) (*Challenge) Alter the algorithm so it performs the same job but can take in any number of numbers. That is, it should take these inputs and give these outputs:

In: $(2, 3, 4) \to$ Out: 3	In: $(4, 6, 8, 10) \to$ Out: 7	In: $(4, 5, 9) \to$ Out: 6

Add all numbers inputted together and divide the sum by the number of numbers inputted. So, if 3 numbers are input: 2,3,4. We add $2 + 3 + 4 = 9$. Dividing 9 by 3 gives the required result.

3. Sort the following list using the insertion sort algorithm: $(5, 2, 4, 1, 9)$. Show your steps. You may find making cards with the numbers written on them helpful.

4. Insertion sort is pretty quick for a lot of unsorted lists. However, it can take much longer on some lists. If a case takes the most amount of time for an algorithm then it is called the worst case. For insertion sort, its worst case is a completely reversed list, such as $(9, 8, 7, 6, 5, 4, 3, 2, 1)$. Use insertion sort to sort this list — it is good practice!

5. (Working this problem alone may be tough! Find a group, teacher, or YouTube video to help.) Other sorting algorithms do exist and one of them is called **Merge Sort**. Merge sort works by combining already sorted lists. Here is an algorithm for combining sorted lists:

   (a) Compare the first elements of two lists by asking: Which element is lesser?
   (b) Put the lesser element into a new list that will contain only sorted elements.
   (c) Compare the next first elements; find the lesser element again.
   (d) Place the lesser element at the back of your new sorted element list.
   (e) Repeat until one list disappears; place all of the remaining elements at the back of your new sorted list.

   Exercise: Follow the steps of this algorithm on these two sorted lists to combine them: $(1, 6, 8)$ and $(2, 4, 7)$.

   But how do we obtain two sorted lists to begin merge sort? Recall that a list of one element is always sorted. What we will do is split our unsorted list up into groups of one, and they must be sorted since they are only one element. Then we can merge each together and create a domino effect until only one list is left.

Exercise: Follow along on a separate piece of paper with the following example:

Merging	Orders	Resulting List(s)
	Begin with this unsorted list.	$(4, 3, 6, 7, 1)$
	Split the list into lists of length one.	$(4), (3), (6), (7), (1)$
$(4), (3)$ — $(6), (7)$	4 less than 3? No. — 6 less than 7? Yes.	$(3, 4), (6, 7), (1)$
$(3, 4), (6, 7)$	Is 3 less than 6? Yes.	(3) — $(4), (6, 7)(1)$
$(4), (6, 7)$	Is 4 less than 6? Yes.	$(3, 4)$ — $(6, 7), (1)$
$(6, 7)$	Put whole list on end.	$(3, 4, 6, 7)$,(1)
$(3, 4, 6, 7)$,(1)	Is 3 less than 1? No.	(1) — $(3, 4, 6, 7)$
$(3, 4, 6, 7)$	Put whole list on end.	$(1, 3, 4, 6, 7)$

6. Repeat questions three and four by using the merge sort algorithm.

7. Samantha invents a sequence of numbers by starting with 1 and then she adds 1 to itself to get the next number, 2. She then adds 2 to itself to get the next number 4, then again $4 + 4$ gives the next number 8, and so on. $(1, 2, 4, 8, 16, \ldots)$

   (a) Samantha is trying to program a computer to find the 27th number in the sequence but she has run into a problem. She doesn't know how to tell the computer to add the number she's currently on to itself. What other operation could see use instead? She could multiply her previous number by 2.

   (b) After 16, find the next three numbers in the sequence. $32, 64, 128$

   (c) * Is there a quicker rule for finding every number in this sequence? Say, if we wanted to find the 5th number, could I put 5 into an algorithm and it would spit out the correct number for the 5th place in this sequence? To find the n-th number **after one** in the sequence, you can multiply 2 by itself n times; or, in other words, compute 2^n.

(d) ** Find the 27th number in the sequence without calculating the 26th, 25th, 24th, or 23rd numbers in the sequence. $2^{27-1} = 67108864$ (minus one because of the **after one** condition).

8. Pascal's Triangle is a number pyramid that grows by continuing to add rows below. Here are three iterations:

1	1	1	1
1 1	1 1	1 1	1 1
1 2 1	1 2 1	1 2 1	1 2 1
	1 3 3 1	1 3 3 1	1 3 3 1
		1 4 6 4 1	1 4 6 4 1
			_ _ _ _ _ _

   (a) Find the next iteration. In the six blanks write the numbers $1, 5, 10, 10, 5, 1$. The pattern adds the two numbers directly above each position, with 1 on the outsides, to compute each row.

   (b) Write down a list of steps that someone could follow to build the next triangle in this pattern.

   - Copy and paste the previous iteration of Pascal's Triangle.
   - 1 goes on the outsides of the row. For the remaining numbers, add the two numbers directly above each position.

   (c) * Write down a list of steps that someone could follow to build any iteration of the triangle from the previous iteration. See previous question's solution.

9. Determine the algorithms in these number factory problems and then complete the next two lines. If you are stuck on (a) or (e), there are hints elsewhere in the lesson.

(a) Input	Output	(b) Input	Output	(c) Input	Output	(d) Input	Output
3	5	3	8	55	10	0	4
20	6	5	12	96	15	1	9
400	11	7	16	57	12	2	14
6000	11	8	18	28	10	3	19
0	4	9	20	59	14	4	24
9		10		60		5	
	10		26		2		34

(e) Input	Output	(f) Input	Output	(g) Input	Output	(h*) Input	Output
11,8	19	100	2	2	5	6,3	3
20,9	29	555	0	4	9	12,7	1
19,9	118	202	1	6	13	40,10	10
27,4	211	811	2	8	17	60,36	12
155,5	1510	918	3	12	25	12,8	4
29,3		600		100		35,25	
	116		3		29		9

(a) Count how many letters are in the English spelling of the input number.

(b) Calculate input multiplied by 2 and then add 2.

(c) Calculate the sum of the digits of the input number. I.e. 55 gives $5 + 5 = 10$.

(d) Calculate input multiplied by 5 and then add 4.

(e) Calculate the naïve sum of the incorrect adding algorithm from the beginning of the lesson.

(f) Count how many closed loops there are in the numbers. 100 has two zeroes. Zeroes contain one closed loop each. So $100 = 2$. $188 = 4$, because of the 8's. And so on.

(g) Calculate input multiplied by 2 and then add 1.

(h) Determine the greatest common divisor of both input numbers. I.e. 3 is the largest number that divides both 3 and 6.

10. Determine an algorithm to generate this sequence of numbers:

$$1, 1, 1, 2, 3, 4, 6, 9, 13, 19, \ldots$$

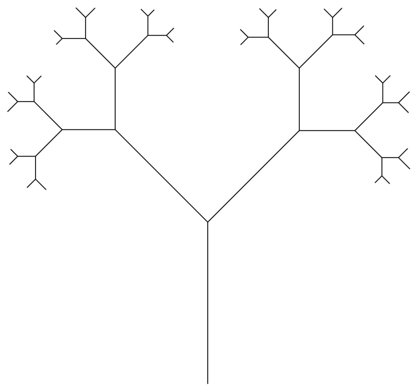
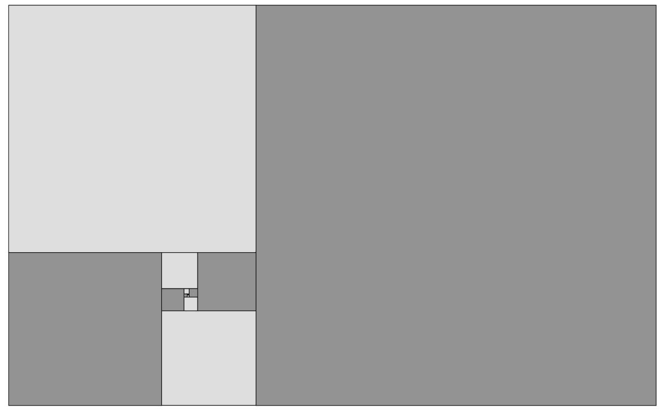
Begin with three numbers and add every other number. So if we start with $1, 2, 3$ we add 1 and 3 first to get 4. Then we add 2 and 4 to get 6. 3 and 6 to get 9, and so on.

11. Here is another puzzle similar to *Four is the Cosmic Number*. Try to determine the algorithm:

- Sixty is Zero, Zero is One, One is One, and One is the Only Number.
- Fifty-One is One, One is One, and One is the Only Number.
- Two-Thousand Three is Two, Two is One, One is One, and One is the Only Number.

Count the number of o's in each of the spellings of numbers. Repeat until one is achieved.

12. A *fractal* is a pattern which is self-similar. This means that if you looked at a small portion of it, it would look like the whole fractal, or vice versa. Here are a couple examples. Can you identify algorithms for each example?

The first fractal begins with a line and repeats the same procedure: At the endpoint, draw two smaller line segments at 90 degrees from each other and 135 degrees from the original line. You did not need to specify angle measurements in your solution. The second fractal is constructed from the four outside squares and is copy and pasted inside the holes forever. We will see what this copy and paste really means next week.

13. (Take this problem home!) Find a model you are interested in making on this website `http://www.origami-instructions.com/` and try folding it. Try some simpler models before the more advanced ones. Remember, Origami is an art and takes practice. If you are interested in learning more about Origami, have a look at Robert Lang's website `http://www.langorigami.com/`. Lang is an origami artist, physicist, and mathematician. After you fold a few different models, ask yourself: do any of the models have steps in common? Do these steps produce similar features in the final creation? Try unfolding your models and looking at the pattern of creases on the sheet of paper you began with; what does each part of the paper become in the final model?