This is a work in progress.

  1. Object Oriented Code

* I think the biggest step for you in computer science now is understanding ‘object oriented coding’. It will make things easier to program, and it will make you a better programmer.
* Object oriented coding is not specific to any langue! Many programming languages actually are object oriented. This will help you with being better in PHP, Javascript, React, any other language you decide to learn!
* Videos:
  + Intro To Object Oriented Programming
    - (He explains in the JAVA language, but the syntax is not so different from javascript/C++/etc, and I think he does  a good job explaining things)
    - Part 1: <https://www.youtube.com/watch?v=jM8gOF6Oi-U>
    - Part 2: <https://www.youtube.com/watch?v=FlDt2ooovXg>
    - Part 4(I think you can skip part 3 -- it’s very java specific): https://www.youtube.com/watch?v=1rp9hG9yYHI
    - Part 5: <https://www.youtube.com/watch?v=Jfe9Xp1YrbI>
    - Part 6: <https://www.youtube.com/watch?v=7ELUCPoPgH4>
  + Questions for you after you complete watching:
    - You don’t need to do any JAVA  coding, but those videos should give you an idea of what object oriented programming is all about! Try to think about the lessons and explain, in your own words, the answers below ( a sentence or two is fine):

1. What does it mean to write object oriented code? How is it different than procedural code (the kind of code we have been writing usually)
2. What is a class in programming?
3. What is an attribute? How does it relate to a class?
   1. Say I have a class describing a pencil, what is an attribute it might have?
4. What is a class method? What do they allow us to do?
   1. Say I had an object describing a computer, what is a method it may have?
5. What is a constructor? Why do we need them?
6. What is the difference between a CLASS and an OBJECT (or ‘instance’) ? (they are very related, do you know how to explain them seperately) ?
7. What does ‘inheritence’ mean for object oriented programming? Why is it helpful?
8. What does the keyword ‘extends’ mean when defining a class? (This is kind of specific to java, but javascript uses the same word for this meaning)
9. Explain to me an interface (advanced question!)
10. Explain to me what abstraction is (advanced question!)

* **Now that you have an IDEA of how object oriented programming works, let us try in Javascript!** 
  + DID YOU KNOW:
    - Javascript didn’t (easily) support object oriented programming until VERY recently. In 2015, a major new version of javascript was released, and we call it “ES6”. Still, some browsers today are not updated enough to use ES6!
    - It introduced an easy way to do object oriented programming in javascirpt, watch this video:
      * <https://www.youtube.com/watch?v=EUtZRwA7Fqc>
      * ^^ I think it’s a great video. He explains the old way to do object oriented programming in javascript, and then moves to new version!
      * <https://www.youtube.com/watch?v=Gh2FaDqZp2g>
  + Questions for you after you complete watching:
    - Lets test out whether you can write some object oriented code in javascript!
    - **Example**:
      * Design a set of classes to represent shapes in general (polygons). Then, create some classes (I have only done square) that inherit from that class to represent different specific shapes.
      * class Polygon {  
          constructor(height, width) {  
            this.height = height;  
            this.width = width;  
          }  
        }  
          
        class Square extends Polygon {  
          constructor(sideLength) {  
            super(sideLength, sideLength); // calls the polygon constructor, which sets height/width.   
          }  
          get area() {  
            return this.height \* this.width;  
          }  
          set sideLength(newLength) {  
            this.height = newLength;  
            this.width = newLength;  
          }  
        }  
          
        var square = new Square(2);

       console.log(square.area());

       square.sideLength = 5;

* + - Question 1:
      * Design a set of classes to represent Surya School People.
        + Your classes should use **inheritence** in some way.
        + You should represent , at a minimum:

Students, teachers, guards/helpers.

* + - * + Think about:

What characteristics are shared between these things, what are different?

What characteristics are unique to students?

What characteristics are unique to teachers?

How could you expand this problem even bigger?

* + - * Design a set of classes to represent a bank system.
        + Your classes should handle the basics of the many bank acocunts stored in a bank. What can be done to each bank account? Maybe deposit money, withdraw money, ask for current balance, transfer money between accounts (if user submits ID’s for both accounts)
        + To show me your ideas work, you should be able to write code in javascript that:

Creates a bank

Creates 3 bank accounts in the bank

Adds $50 to each bank account in the bank

Asks for the balance of a bank account

Removes 50 from bank accoujnt 1.

Tries to remove 50 from bank account 1 (but should be rejected, balance would be 0)

Moves 50 dollars from bank account 2 to bank account 3.

Asks for balance of bank account 2 (shoudl be 0)

Withdraws all money from bank account 3.

2.  ES6 (New Javascript)

* Learn about ways to use javascript objects Remember that an ‘object’ in java is related, but not necesarily the same as OO programming
  + - Remember, in javascript, an object is just a HashMap/Dictionary:
      * Let myobj = {key1: ‘value’, key2: ‘value2’}
      * Myobj.key1 = ‘bla’
      * console.log(myobj.key1) // prints ‘bla’
      * console.log(myobj.key2) // prints ‘value2’
  + <https://www.youtube.com/watch?v=Gh2FaDqZp2g>
* Spread Operator
  + <https://www.youtube.com/watch?v=j2DMwUYEC88>
* Destructuring:
  + <https://www.youtube.com/watch?v=eNPlzq9kPv4>
* Arrow Functions:
  + <https://www.youtube.com/watch?v=oTRujqZYhrU>
* Let / Const / Var (variable creation)
  + <https://www.youtube.com/watch?v=LTbnmiXWs2k>
* Arrays
  + <https://www.youtube.com/watch?v=WbKTQU5qJCY>
  + Questions for you after you complete watching about ES6:
    - What is the spread operator ? How is it used?
    - What is object destructuring? Why is it useful?
    - Explain the difference between Let/Const/Var

3. Linked List

* Linked list is a very interesting data structure.
* You can learn a bit here:
  + <https://www.youtube.com/watch?v=njTh_OwMljA&t=44s>

Below I have given you a javascript example of a simple class for a linked list node ( it has a val, and a ‘next’ which is the next node). I also made a LinkedList class. It just contains a length, and ‘head’ which is the first node in the list. It will have a lot of functions though that can be used to manipulate all the nodes in the list. As you can see, I implemented ‘insert’. If you give the list a value, it will add a node to the list with that value. Play around with the list and see how it works, see if you can implement any other methods (such as sort, delete, etc)

class LinkedListNode {

 constructor(val) {

   this.val = val;

   this.next = null;

 }

}

class LinkedList {

  constructor() {

    this.head = null;

    this.length = 0;

  }

  insert(val) {

    let newNode = new LinkedListNode(val); // Make a new node.

    if (this.head == null) { // If there are no nodes in our list, this should be our head node.

      this.head = newNode;

    }

    else { // Otherwise, loop through all of our nodes until we are at the end of our list!

      let cur = this.head;

      while (cur.next != null) {

        cur = cur.next;

      }

 // At this point, cur will be our last node, so make it’s ‘next’ be the new node, and we are done!

      cur.next = newNode;

    }

// We inserted a node… so update the length of list!

    this.length++;

  }

  getLastNode() {

   // can you do this one?

  }

  sortList() {

    // hard question, any ideas how you could sort your list?

  }

  printEveryOtherNode() {

   // should print every other value. If your list looks like (2) -> (4) -> (8) -> 10)  it should print: 2 8

  }

  deleteNode(val) {

// Should remove any node in list with this val. How can you delete node? (2) -> (4) -> (1) . If you change the node holding 2 to have it’s ‘next’ node be the node holding 1, you’ve deleted 4! Challenging, but cool.

  }

  printEvenNodes() {

    // should print nodes that have even values. If your list is (2) -> (3) -> (8) -> null it would print 2 8

  }

  reverseList() {

   // hard question. can you think of a way you could reverse the order of nodes?

  }

}

let myLinkedList = new LinkedList();

console.log(myLinkedList.head);

myLinkedList.insert(4);

console.log(myLinkedList.head);

myLinkedList.insert(5);

console.log(myLinkedList.head);

console.log(myLinkedList.head.next);

4. React

* It might be easier for us to discuss react together. You can play around with the files in that download, but I think we must learn together on video chat… it’s hardder than I remember to get started without a teacher :)
* Once you have watched all of the videos above, you will have the tools ready to try out react. Even if you don’t get this far over break, you will have still learned a lot!
* We’ll be using **ES6** javascript to write react, so that’s why you have learned that first.
* React uses ‘components’, which are **classes**, so we must understand that, and object oriented coding!
* <https://react-cn.github.io/react/downloads.html>
* I suggest you download the ‘starter kit’ before you leave.
* <https://ihatetomatoes.net/wp-content/uploads/2017/01/react-cheat-sheet.pdf>