//////////// Building Blocks ///////////////////////////////////////////////

* Building Blocks
  + Solve them as fast as you can and don’t leave any unanswered
    - If stuck just ask or look it up
      * E.g. Google: MDN sort()
        + MDN documentation will come up
      * If something is not behaving the way you expect, use console.log to experiment and see what exactly is happening
      * If you still cannot figure it out, try starting from scratch. Implement an alternate solution.
      * Reach out to
        + “#mod1\_building\_blocks” channel on Slack
      * Submit a HelpDesk ticket
  + **DO NOT PRESS “SUBMIT ANYWAY” IN Repl.it**

//////////// Know The Easy Stuff Cold //////////////////////////////////////

* Know the easy stuff and should be able to fly right through it

////////// Coding Style Guide //////////////////////////////////////////////

* Indentation
  + Use spaces instead of Tab key
  + When a line closes a block, that line starts at the same level as the line that opened the block
    - i.e. if (condition) {

action();

}

* + Use a new line for each variable declaration
    - Use a new “var” statement for each line you declare a variable on
    - i.e. var ape;
    - var bat;
* Names
  + Variables
    - Single descriptive word is best
      * Descriptive means its directly describing some noun in the “domain” of your problem
      * Name your variables after their purpose, not their structure
        + i.e. var animals = [‘cat’, ‘dog’, ‘fish’];
  + Arrays
    - Should have descriptive words
  + Maps
    - Should have descriptive words
  + Boolean
    - Variables that represent Boolean values should be named accordingly
      * Since a Boolean either is true or false
        + Prefix variable name

Is

Are

* + - * + E.g.

isValid

areAvailable

* + Functions
    - Start with a verb
      * verbObject
        + e.g. calculateTotal
        + listInventory
        + e.g. var countWaterBlocks = function()
  + Capital Letters in Variables
    - Capitalize the first letter in their variable names to indicate they contain a “class”
    - Capitals letter only on functions that are written to be run with keyword “new”
    - Use all-caps for constant variables (ones that will not change throughout the life of the program)
    - Examples:
      * Capitalize class constructor function name
        + Function Animal()
      * All-caps constant variable name
        + Const MAX\_ITEMS\_IN\_QUEUE = 100;
* Quoting
  + Prefer single quotes around JavaScript strings, rather than double quotes
    - Single quotes allow for easy embedding of HTML
      * E.g. var dog = ‘dog’;
      * E.g. var cat = ‘cat’;
* Semicolons
  + Don’t forget at the end of the line
  + Semicolons are not required at the end of statements that include a block
    - if
    - for
    - while
  + A function may be used at the end of a normal assignment statement, and would require a semicolon
    - Var greet = function () {

Alert(‘hi’);

};

* Operators and Keywords
  + Use strict comparison operators
    - ===
    - !==
  + ternary operator
    - x ? y : z;
      * evaluates to y if x is true, otherwise evaluates z
    - Makes for compact code
    - Hard to read if it is too complicated
      * Only use ternary operators if its extremely clear and short to do so
  + Not-operator (!)
    - Keep the not-opeartor right next to the item it is negating
      * if (!isEqual) {
* Switch Statements
  + Avoid using switch statements
    - Prone to error due to missing “break” statements
* Mixing conditionals and returns can make the flow hard to trace
  + Instead use
    - If (!text) {

return true;

}

var mySet = new Set();

* Avoid return statements in the middle of the flow
  + Maintain one return at the end of the function body
* Brevity
  + Write the least code you can that is still completely clear
    - Function square(n) {

return n \* n;

}

* Avoid Negation
  + If you find yourself with lots of negation, have an opportunity to increase clarity by converting it to a positive
    - If(equalSizes && equalValues) {

//positive outcome

} else {

//negative outcome

}

* Return Boolean results directly
  + return charSet.size > text.length;

Given an object and a key, “getProperty” returns the value of the property at the given key.

**var obj = {**

**key: 'value'**

**};**

**var output = getProperty(obj, 'key'); console.log(output); // --> 'value'**

function getProperty(obj, key) {

// your code here

/\* START SOLUTION \*/

if(obj) {

return obj[key];

} else {

return null;

}

/\* END SOLUTION \*/

}