Data structures

1. A collection of values
   1. Values can have relationships, and have functions applied to them
2. Each data structure is good
3. A backpack, cabinet, fridge
   1. Data structures are containers, each useful for their own thing
4. A way to organize data
5. 2 problems: how to build, and how to use
   1. Data structures are usually built already

Ram

1. Where variables are stored
2. Lose memory when turned off

Storage/hard drive

1. Where files like audio, video, =docs are stored.
2. Remains after computer turns off

CPU can access RAM faster

1. When we use an app like chrome, variables are stored in RAM
2. When we want to save it, it will be stored in storage

Ram is like a massive storage, with shelves(addresses) that holds a lot of informatioin

Allows programs to be run

A bit is a number, tiny electrical switch

Each shelf has 1 byte of storage

Each shelf holds 8 bits. Each Number is a bit than can be turn on (1) or off (0)

Each shelf is a byte

Cpu connected to memory controlled which reads and write. Connected to every shelf. Very accessible

Because ram is connected to address, bit, and bytes, it is very accessible

Closer addresses create a faster program. I.. 0 to 1, 0 to 1000

Cpu cache, tiny memory that stores very recent items. I.e. cache

Data structures are ways to store info

Integers are represented by 32-bits

8 bits \* 4= 32

We can access any shelf (RAM), programs tend to access nearby memory

Faster travel = faster program

Permanent === persistent

When a number is above a threshold, it will be converted to infinity

Each data type has a certain number of bits

Arrays

1. List items in memory one after another
2. Best for iterating one by one
   1. Lookup (o1) Fast
   2. Push O(1)
   3. Insert On)
   4. Delete O(1)
3. 32 bit system (8 bits in a row, and 3 shelves)
   1. 4 items taking up 4 shelves, takes up 16 bytes of storage
   2. Strings[2] will grab the third item of the array named “strings”
   3. Push
      1. Adds something to the end of an array
      2. O(1) not looping through anything
   4. Pop()
      1. Opposite of push
      2. O(1) no looping, quick
   5. Unshift()
      1. Add to the beginning
      2. O(n) everything has to shift over in memory to fit the first item(this is a loop)
   6. Splice
   7. Append
      1. Can be o(n)

Array can be a list

2 tpyes of arrays, static and dynamic

1. Static
   1. Fixed in size, must be specified ahead of time
      1. I.e. create an array of 7 items
2. Dynamic array

Classes

Context tells us where we are in an object

1. e. what is left of the .
   1. Window.alert()
2. This points to the object we are in right now

Instantiation

Makes a copy of an object

Class Player{

Constructor(name, type){

This.name = name;

This.type = type;

}

Introduce(){

‘Hi I am………”

}

USE THIS if using a CLASS