Lecture 0 - Scratch

Binary

What is computer Science?

Programming is a big part of computer science, where you write code to express ideas and solve actual problems involving data, but computer science itself is the study of information.

What does it mean to solve a problem?

Problem solving is where you get some input (which represents the problem you want to solve) and then there will be an output (which represents your goal which is to solve the problem) and then there will be a secret recipe in the middle which will help you solve the problem.

What is Unary and Binary?

Unary is a type of system which would help us solve problems which include numbers (for ex. taking attendance at the beginning of class). In other words, unary is a very simple system of using a single symbol to solve some problems. Unary, in mathematical and technical terms is called "Base-1" meaning that the number that you are operating with has 1 digit in it.

Computers use a language other than Unary, this language is called Binary. Bi means 2, so computers have 2 digits which they would use to communicate with or solve problems within.

The technical term bit comes from the word binary digit. A bit is 0 & 1. So in simple words, us humans, in our everyday life use the Unary system to solve problems, meaning that we use the digits from 0 to 9. On the other hand computers use the Binary system to solve problems, meaning that they use the digits 0 or 1.

How do computers only speak in Binary?

The simplest way in which we could understand this by taking the light bulb as an example. So if humans were a computer, to represent the number 0 you would keep the light switch off & to represent the number 1 you would switch the light bulb on.

Now, this is relevant to computers because inside a computer there are thousands, even millions of tiny switches (we could metaphorically think of these switches as light bulbs). These switches are called transistors.

Just like a light bulb, these transistors could be switched on to represent 1 and switched off to represent 0.

Obviously, with this ability of computers they could count 0 to 1 but it turns out that they could count even higher with a little bit more electricity.

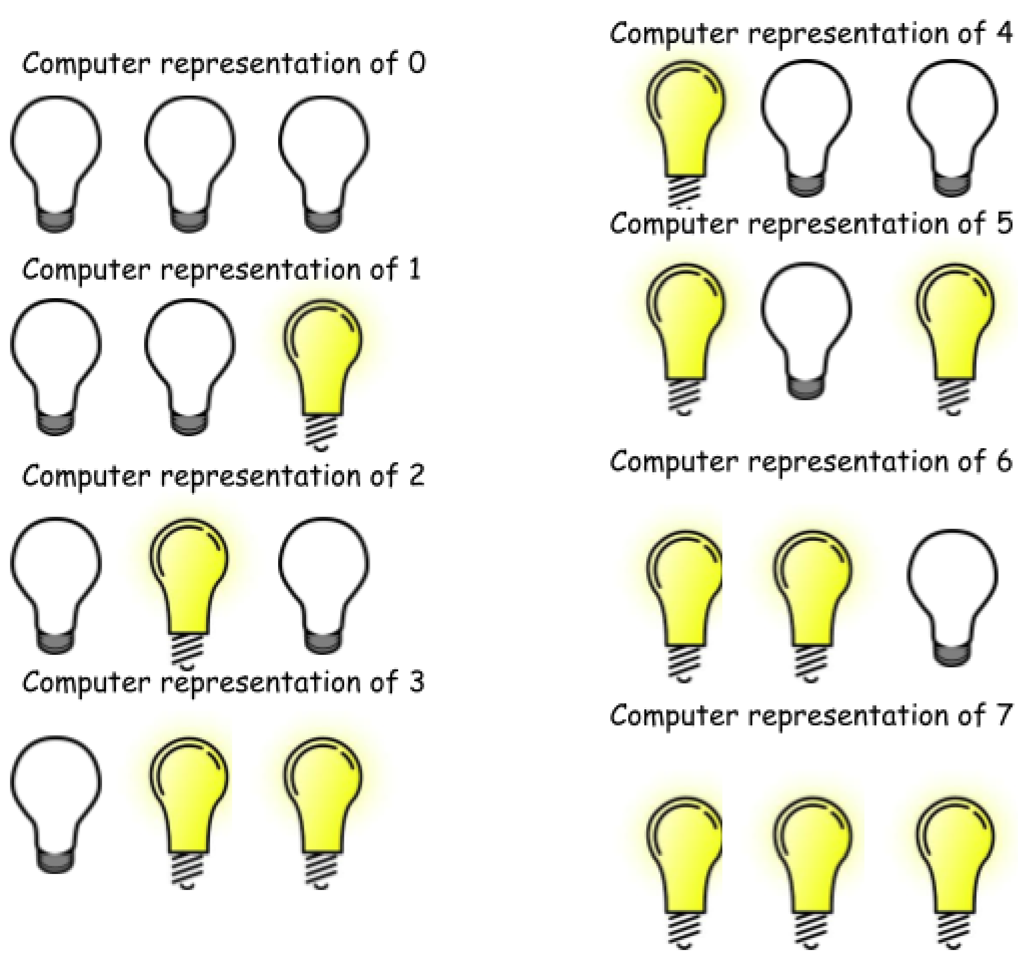
Based on the above concept, when we take a singular light bulb or a transistor and switch it on, it represents 1 and when we switch it off, it represents 0. So to represent more numbers we would need to use more light bulbs.

To understand this concept better, let us take 3 light bulbs which are all switched off. We know that when a light bulb is turned off it represents 0.

So we would think that with these 3 light bulbs (when they are switched on) could represent 0, 1, 2, 3, but when we consider the combination of bulbs being on and off the computer might be able to represent more than 0, 1, 2, 3.

For example we could say that the computer stores the number 0 when all 3 light bulbs are off. When the first light bulb is on but the other two are off, this could be how the computer stores the number 1. Similarly, when the second light bulb is turned on and the other two are off, this could be how the computer stores the number 2.

Following the above example, with 3 light bulbs the computer can store the number 0, 1, 2, 3, 4, 5, 6 & 7.



What we have to understand from this example is that the way the computer stores these number values is by memorizing the patterns in which these transistors or light bulbs turn on or off.

We cannot represent more than 0, 1, 2, 3, 4, 5, 6, 7 with 3 light bulbs because the pattern or the combinations in which all three of these lights could uniquely be arranged is only in 8 ways. Thus, we can only represent 8 numbers. The same method could be used for representing numbers with multiple light bulbs.

In the above example, the computer used base 2 (Binary) to represent numbers 0 through 7. Binary, just like the decimal system that we use, follows certain types of rules.

Let's recall back to how decimal place values work,

=> 123

When we look at the above number "123" our mind reads it as "one hundred and twenty three". This is because we know that the rightmost number (3) is in the 1's place, the middle digit (2) is in the 10's place and the leftmost digit (1) is in the 100's place. Now we just do quick mental math, which would lead to "100 x 1 + 10 x 2 + 1 x 3" which is "100 + 20 + 3" gives us the number we know as "123".

We could also use base terminology to represent these places. For example the 1's place is 10^0, the 10's place is 10^1, the 100's place is 10^2 and so forth.

Now, if we take computers, they obviously only have access to electricity on or off (0 or 1). So they tend to use a different base. 2^0, 2^1, 2^2 and so forth, a.k.a 1's place, 2's place, 4's place and if we keep going 8's place, 16's place and so on. At the end of the day, the idea for both the decimal system and the binary system is fundamentally the same.