TECHNOVATION 2020

COMPUTATIONAL THINKING WORKSHOP

DAVID LEDO

UNIVERSITY OF CALGARY

LESSON PLAN

CLASSROOM A
Logistics - 10 min
Quick Intro - 15 min
Computers Aren't Smart - 15 min
THEORY: Algorithms, conditionals, logic - 20 min
Marbles 1 - 20 min

--- BREAK --- 15 min

Marbles 2 - 20 min
Vending Machine - 15 min
THEORY: Variables, Loops, Functions, Events - 20 min
Help the Cat - 15 min

--- BREAK --- 15 min

Introduce Code.org, Scratch

CLASSROOM B
Logistics - 10 min
Computers Aren't Smart - 15 min
Quick Intro - 15 min
Marbles 1 - 20 min
THEORY: Algorithms, conditionals, logic - 20 min

--- BREAK --- 15 min

Marbles 2 - 15 min
THEORY: Variables, Loops, Functions, Events - 20 min
Vending Machine - 15 min
Help the Cat - 15 min

--- BREAK --- 15 min

Introduce Code.org, Scratch

COMPUTATIONAL THINKING

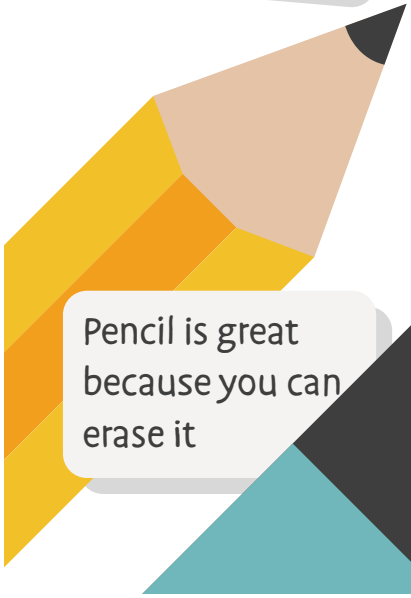
Take a moment to think about the fact that you can grab a piece of paper and start **writing**

There are a lot of processes taking place. You are able to **write** because you are able to **speak**

Your motor skills make the words come out on paper

You tie symbols together that make **words**, and give those words **meaning**

And depending on the **tools** you use, you get different effects



Pencil is great because you can erase it

Markers are great for permanent lines

Like most things... ***what tool you choose has a time and a place***

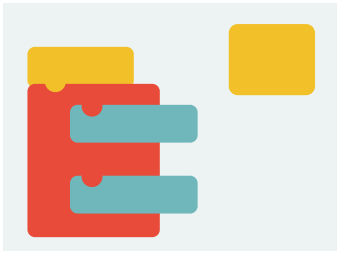
Programming is no different!

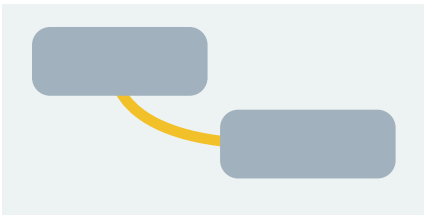
You'll learn how to talk to **computers**

And overtime you'll try many **languages**

```
mov %r1, %r3
add %r3, 5
```

```
public void Start()
{
    // code
}
```





>CODE

Different languages will have different **tools**

So before we can dive into code, let's learn **how to think** in ways we can talk to computers.

And if you struggle, figure out whether the problem lies in how you're thinking about ***problems***, the ***code***, or the ***tools***.

COMPUTERS AREN'T SMART

1. Draw! Using only simple shapes (straight lines, circles, rectangles), draw an animal in the box below. Don't let your group see it.

2. Instruct your group to draw your animal! Using only words (no pointing or gestures), and without showing your drawing to your friends, get them to try and replicate what you drew. Then switch and try drawing using your friends' instructions

3. Discuss what happened. Was it easy to draw the animal? Were you able to use those instructions to draw your partner's animal?

Notice how you use words and specific instructions. Computers rely on pre-made instructions and are even more unaware of what you might actually mean.

ALGORITHMS, CONDITIONALS, AND LOGIC

HOW DO YOU PUT A GIRAFFE INTO A REFRIGERATOR?
1. Open fridge
2. Put giraffe in
3. Close refrigerator

HOW DO YOU PUT AN ELEPHANT INTO A REFRIGERATOR?
1. Open the fridge
2. Take out the giraffe
3. Put elephant in
4. Close the refrigerator

That old joke may seem silly, but it's a great example of an algorithm. An algorithm is a sequence of steps that follows given a set of rules. Computers think in terms of small instructions done in the order that you give it to them. Think of activities in your life, such as brushing your teeth, or tying your shoes. These are all algorithms.

CONDITIONALS

Often times we can branch a problem by adding conditions, or asking "if". If the condition is true, then the code inside will execute, otherwise, it will go on to the next step which might be another condition (e.g. "otherwise"). This is denoted as "else".

For example, imagine we have an alarm set for 7 am on weekdays, and 10 am on weekends.

```
Function Alarm(Day)
   if(Day is weekday)
       Ring at 7 am;
    else
       Ring at 10 am;
```

Suppose that Tuesdays we need to get up a bit later, so 7:30 am. Then we have something like this:

```
Function Alarm(Day)
   if(Day is Tuesday)
        Ring at 7:30 am;
   else if(Day is Weekday)
       Ring at 7 am;
   else
       Ring at 10 am;
```

Note that by having if-else if-else, you are indicating that the conditions are within the same block, so they are talking about the same thing.

What happens if you change the order of that sequence of events?

ALGORITHMS, CONDITIONALS, AND LOGIC

Over time you will notice how much the previous example is taken for granted, given that we are still being relaxed about what the "rules" are. Suppose that our program is smart enough to know what the days of the week are, but then, what is a weekday and what is a weekend?

LOGIC

There are logic operations we can do when we have some information, such as the day of the week. These operations are: "and", "or", and "not". These are similar to our everyday conversations, but for code to execute, the overall statement has to be true.

OR: if we have an <A or B> operation, it will be true if A is true, if B is true, or if both A and B are true
AND: if we have an <A and B> operation, it is only true if both A and B are true
NOT: you can negate a statement by adding the word "not" at the beginning of it

Examples:
If <my pants are blue> AND <my pants are made of denim>, then I am wearing jeans
If <the temperature is below -10°C> OR <it's snowing>, then I don't want to go outside

Now let's define a Weekday: assume we have a list of possible days ranging from Monday to Sunday

Function IsWeekday(Day)

```
if(Day is Monday or Day is Tuesday or Day is Wednesday or Day is Thursday or Day is Friday)
    return true;
```

The best news is that there's always more than one right answer, you can also try this:

Function IsWeekday(Day)

```
if(Day is not Saturday or Day is not Sunday)
    return true;
```

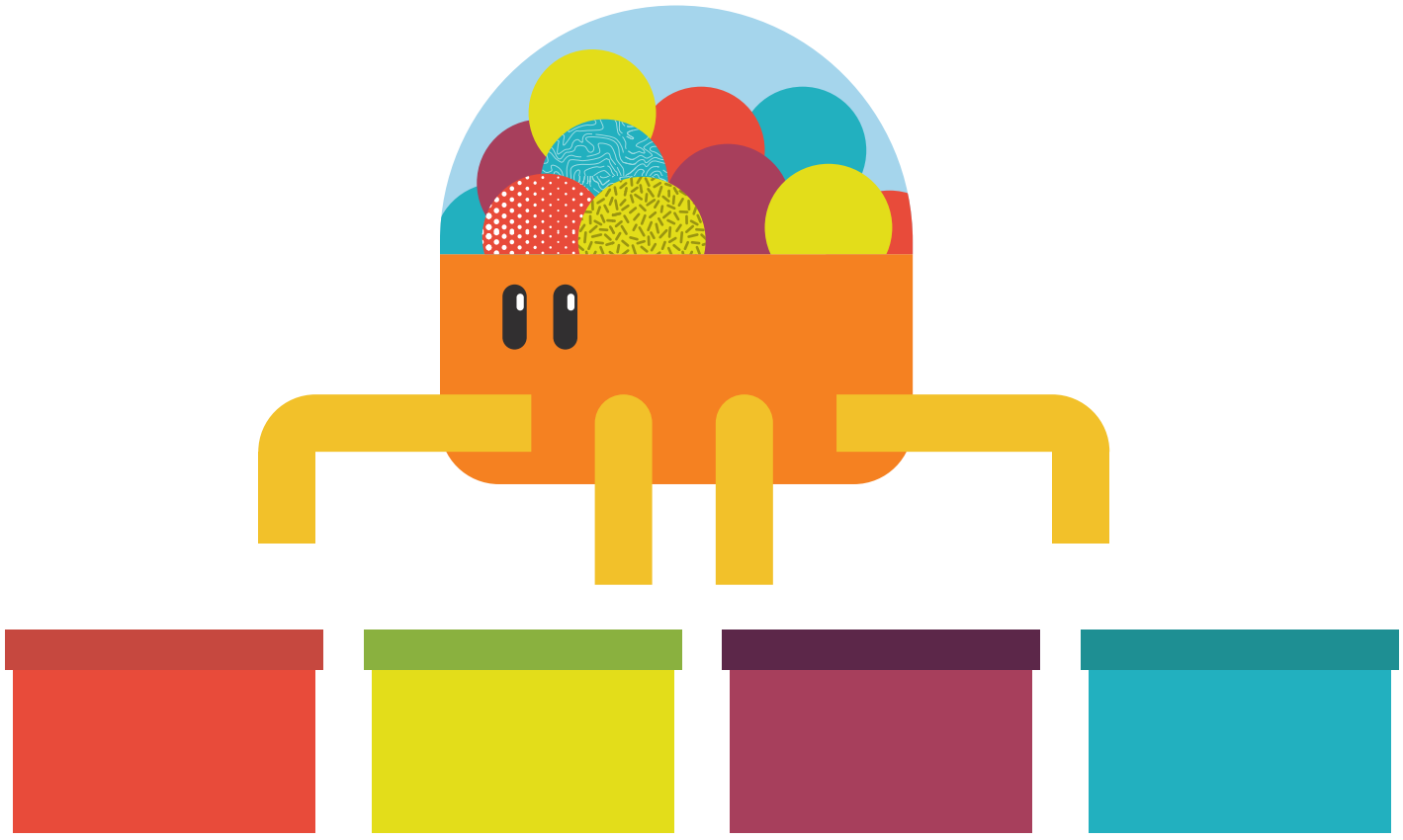
LOSING YOUR MARBLES!

We are making a marble sorting machine, which can separate marbles according to colour. However, the program for it is missing!

1. Role-play with your group. Go through the cut-out marbles and place them onto the corresponding piles. Look at the steps you are taking and write them down.

2. Write the basic program for the machine. Whenever a marble shows up, use *conditionals* to put them in the right places. The machine has a camera, so it can know the colour and pattern of marbles that go inside. Here are some commands the machine understands:
- Place on red container
- place on green container
- Place on blue container
- Place on purple container
- Dispose

3. Let's keep only solid marbles! Write a function that tells us if a marble has patterns. Then, modify your last program so that if a marble has patterns, it is disposed, otherwise it can go into the containers.



VENDING MACHINE

Your job is to program this vending machine so that people can buy a drink of their choice. There are 6 drinks to choose from, and all are worth $1. The machine only takes $1 at a time.

1. Role-play with your group. One student acts as the machine, others take turns to buy drinks. Look at the steps you are taking and when the machine does not work (such as choosing a drink without there being money).

2. Write the code for this machine to work: When a button is pressed, check which button was selected and whether there is money or not, and return the right drink.

VARIABLES, LOOPS, FUNCTIONS AND EVENTS

VARIABLES

One rule we have been taking for granted this whole time is that we need a way to hold information. Most things that computers do are stored as what is called a variable. This is essentially a "box" that you create as a programmer to store information you wish to use or manipulate. Variables can be a number, a word, or a boolean (a true or false value).

For example, a computer program may ask you to type your name, and press a button to confirm. Then show you a welcome message.

Function OnButtonClicked()

```
var Name = text entered on textbox;
ShowMessage("Hello " + Name)
```

LOOPS

Computers are really good at repeating activities. Loops are a way of telling the computer to keep doing something as long as a condition is true. This way we don't have to retype the same thing an infinite number of times, but also we can use variables to change how those loops behave.

There are two ways of doing loops, we can check the condition at the beginning (while) or at the end (repeat-until)

```
var count = 0
while(count < 10)
    Write("Hello " +  count);
    count = count + 1;

var count = 0
repeat
    Write("Hello " +  count);
    count = count + 1;
 until(count >= 10)
```

Note how we are using a variable to keep track of how many times we've gone through, what happens if we remove the count incrementing its value?

VARIABLES, LOOPS, FUNCTIONS AND EVENTS

FUNCTIONS

Functions are a way of wrapping up multiple steps together, which we can then reuse many times.

For example, we wrote a function earlier to know if it's a weekeday or not:

```
Function IsWeekday(Day)
    if(Day is Monday or Day is Tuesday or Day is Wednesday or Day is Thursday or Day is Friday)
        return true;
```

Notice how Day is a variable that gets passed into the function. We can then use that function in a larger piece of code, and we can even build on it. For example, we can define a weekend as "not a weekday", right?

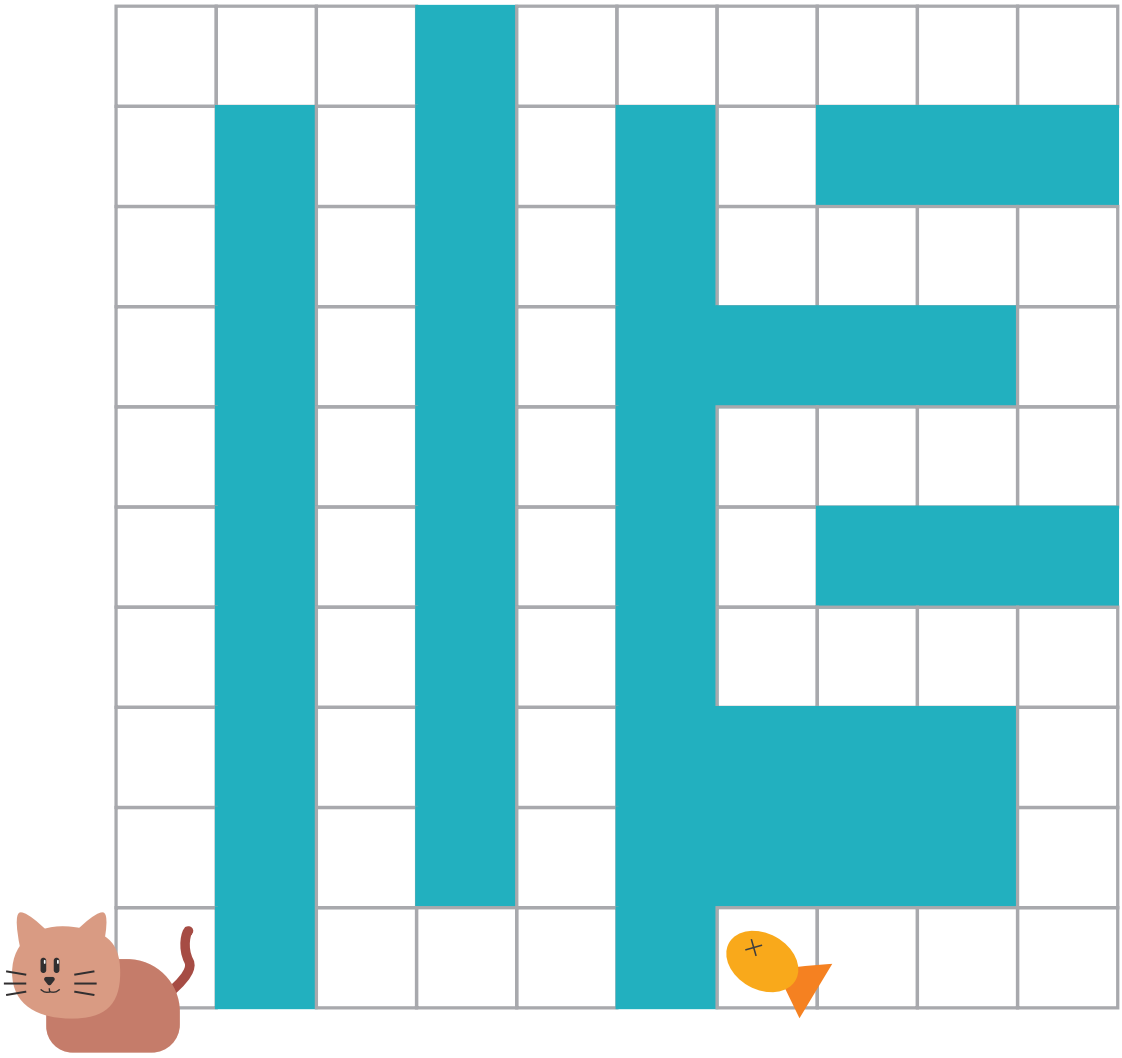
```
Function IsWeekend(Day)
    if(Day is NOT IsWeekday(Day))
        return true;
```

Events

There are special functions that are triggered "when" something happens. This can be when a person does something (e.g. when a button is pressed, when the mouse is clicked). But it can also be an internal event. For example, when the phone screen is locked.

CAN YOU HELP THE CAT?

1. Help the cat get the fish! You have the following instructions available, and make sure the kitty doesn't touch the water: UP, DOWN, LEFT, RIGHT. Each of those instructions move the cat 1 block in the specified direction. Use functions and loop to reduce the number of steps taken!

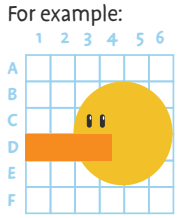


SMARTER DRAWING

1. Draw an animal! Unlike last time, you have a few rules. Now we have a grid. Here are the commands you can follow, fill in the black with the corresponding grid location:

- Draw Circle from _ to _
- Draw Rectangle from _ to _
- Draw a line from _ to _
- Draw an eye on _

For example:



Circle from B3 to E6
Rectangle from D1 to D4
Draw an eye on C3
Draw an eye on C4

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

A B C D E F G H I J K

2. Instruct your group to draw your animal! Using only words (no pointing or gestures), and without showing your drawing to your friends, get them to try and replicate what you drew. Then switch and try drawing using your friends' instructions

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

A B C D E F G H I J K

3. Discuss! Check with your group, how did it go this time compared to the beginning?