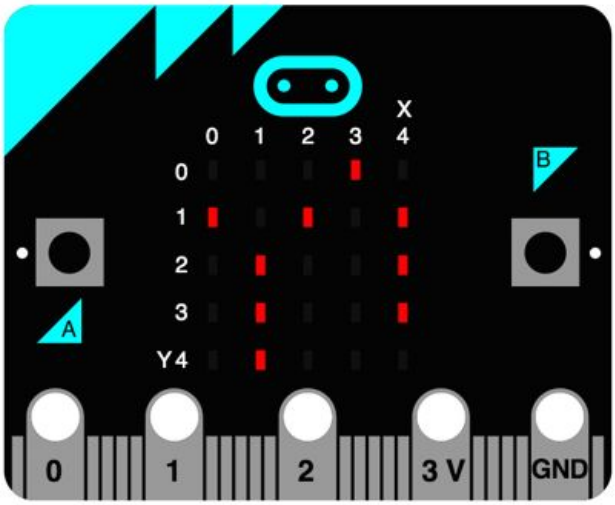
Project 1 - The Fortune Teller

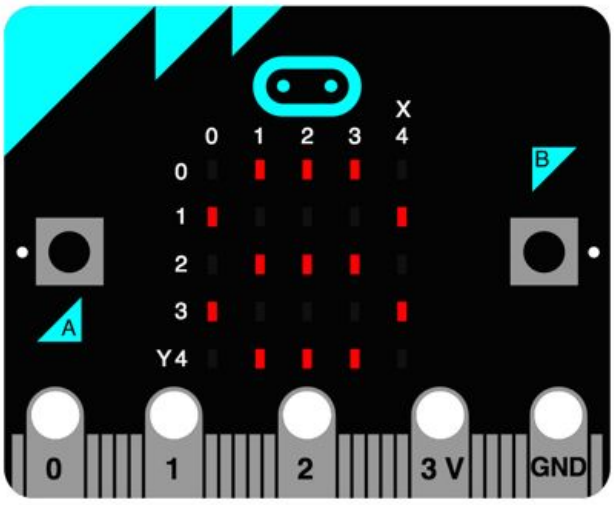
For this project, we're going to write some code that interacts with some things on the micro:bit - buttons and the *accelerometer*. You know what buttons are - you press them and they do something. But do you know what an accelerometer is?

An accelerometer is a device that measures movement. A lot of smart phones have accelerometers that can tell if the phone is moving up or down, or side to side. The accelerometer in the micro:bit isn't as powerful as that, but it can detect movement, especially if you shake the micro:bit.

Version 1: press a button



Version 2: shake!



Version 1 - Press a Button

First let's write some code. In your Mu editor, type or copy this code and then flash it to the micro:bit by clicking on the 'Flash' button.

```
from microbit import *
import random

# a list of possible fortunes
fortunes = ['Yes', 'No', 'Maybe', 'Try again later']

# start with a clear display every time
display.clear()

while True:
    a = button_a.was_pressed()
    if a:
        myrange = len(fortunes)
        myindex = random.randrange(myrange)
        random_fortune = fortunes[myindex]
        display.scroll(random_fortune)
```

One of the new things we're doing here is importing the `random` library - that tells us that we're going to be sending something different to the display every time we press a button. The 'fortunes' list is a list of possible things we can have the micro:bit say.

The part that says "`while True:`" is important. This puts the micro:bit into an *infinite loop*. Remember when we talked about `while` loops and how it's bad to get stuck in an infinite loop? Well this is a case where it's actually useful - the micro:bit is always checking for button presses, forever.

`button_a.was_pressed()` is a function that comes with the microbit library - we get to use it because we wrote `from microbit import *` at the top of our code. This function returns `True` if the button is pressed.

Next we check to see if the button was pressed, and if it was, we do a few things:

1) Figure out how many items are in our `fortunes` list, using the Python function `len`
2) Give that number to `random.randrange()`. This function generates a random number, but makes sure that the number isn't higher than the range.
3) Use the random number as an *index* on the `fortunes` list to retrieve a random item.
4) Display that randomly selected item on the micro:bit display

Version 2 - Use the Accelerometer!

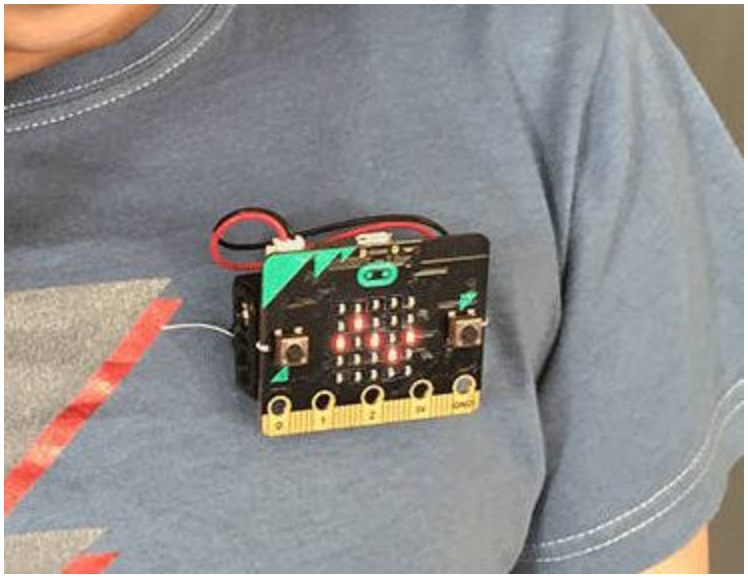
Now let's try something a little bit different. This code is very similar to what we just saw, but it uses a longer (and more interesting) list of possible answers, we use the random library in a slightly different way, and we're going to shake our micro:bit instead of pressing a button to get our fortunes!

```
from microbit import *
import random

answers = [
    "It is certain",
    "It is decidedly so",
    "Without a doubt",
    "Yes, definitely",
    "You may rely on it",
    "As I see it, yes",
    "Most likely",
    "Outlook good",
    "Yes",
    "Signs point to yes",
    "Reply hazy try again",
    "Ask again later",
    "Better not tell you now",
    "Cannot predict now",
    "Concentrate and ask again",
    "Don't count on it"
    "My sources say no",
    "Outlook not so good",
    "Very doubtful",
]

while True:
    display.show("8")
    if accelerometer.was_gesture("shake"):
        display.clear()
        sleep(1000)
        display.scroll(random.choice(answers))
```

The first thing that's really different is that we don't clear the display - instead we display the number '8' whenever the micro:bit isn't moving.

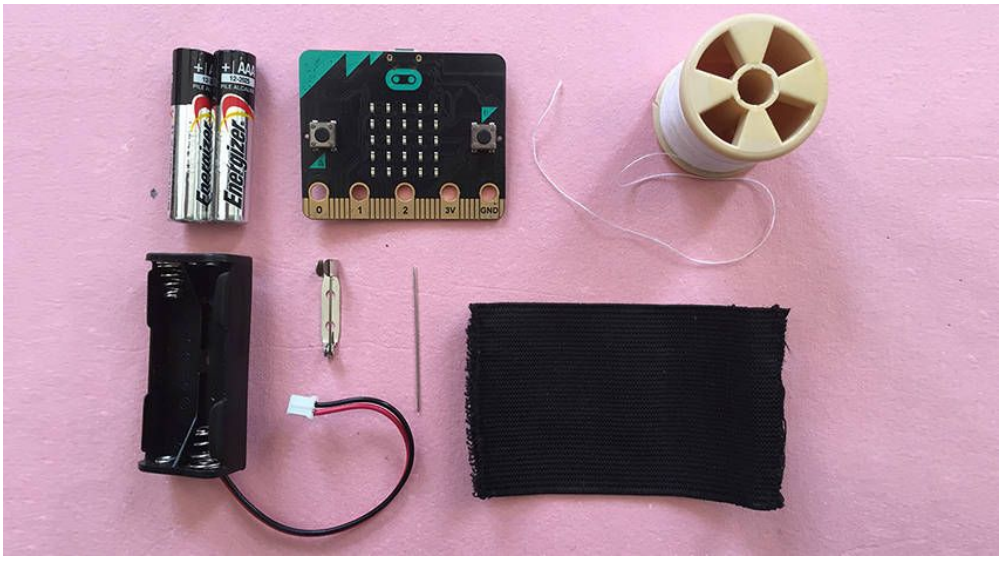
Next we check to see if the micro:bit is shaking - if it is, we clear the display, pause for a few seconds, then display a `random.choice()` from the list of `answers`. (That's a whole lot easier than how we got our random answer in the first example!)

Now comes the fun part! Let's make a badge so that we can walk around wearing our fortune telling machines!



Supplies you should have:

1 micro:bit
1 battery holder
2 AAA batteries
thread
1 needle
a piece of elastic
1 safety pin



Step 1: Stitch the elastic

Thread your needle, knot the thread, then stitch along the cut edge of the elastic to close it up.

Step 2: Roll the elastic to hide edges.

Roll the top edge over the shorter edge and tuck in so no raw edges are showing.

Step 3: Stitch the edge down.

Step 4: Checkpoint!

It should now look like a tube.

Step 5: Turn whole tube inside out.

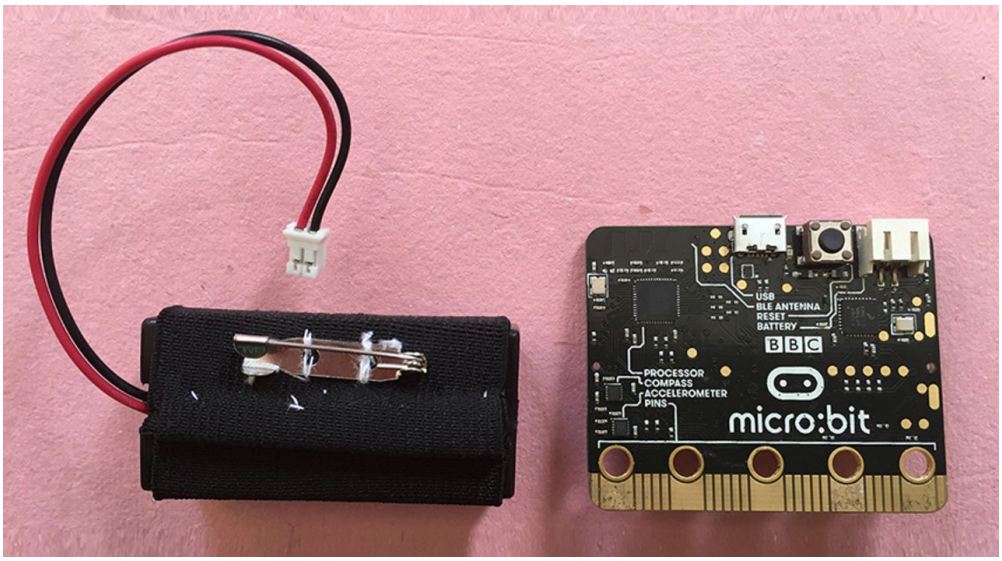
This will neaten it up and hide stitches.

Step 6: Stitch the pin.

Stitch the safety pin securely to one side of the tube, preferable with the seam just below the badge pin, so that when it's worn, it hangs well.

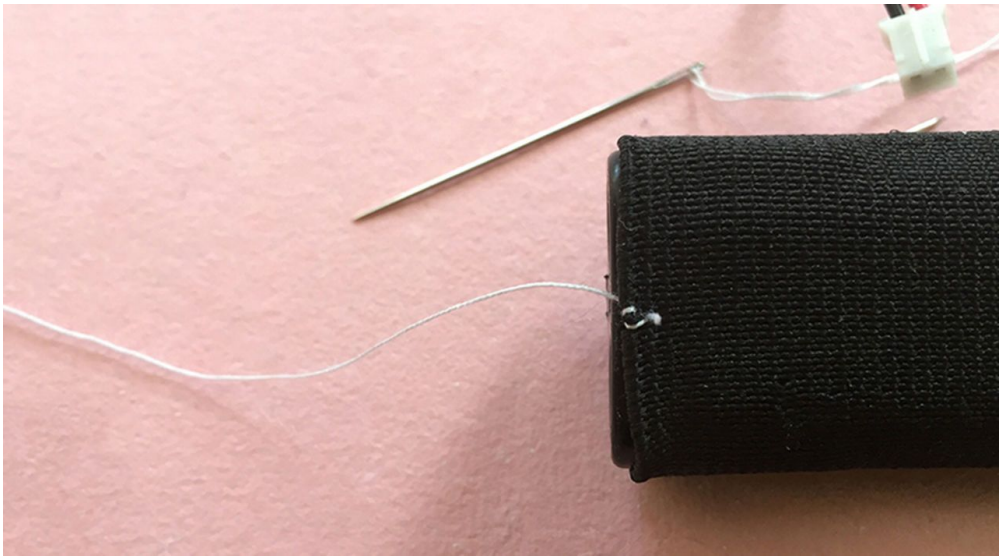
Step 7: Insert the battery holder into the elastic tube.

After inserting 2 AAA batteries, squeeze the battery holder into the tube. It should be a snug fit.



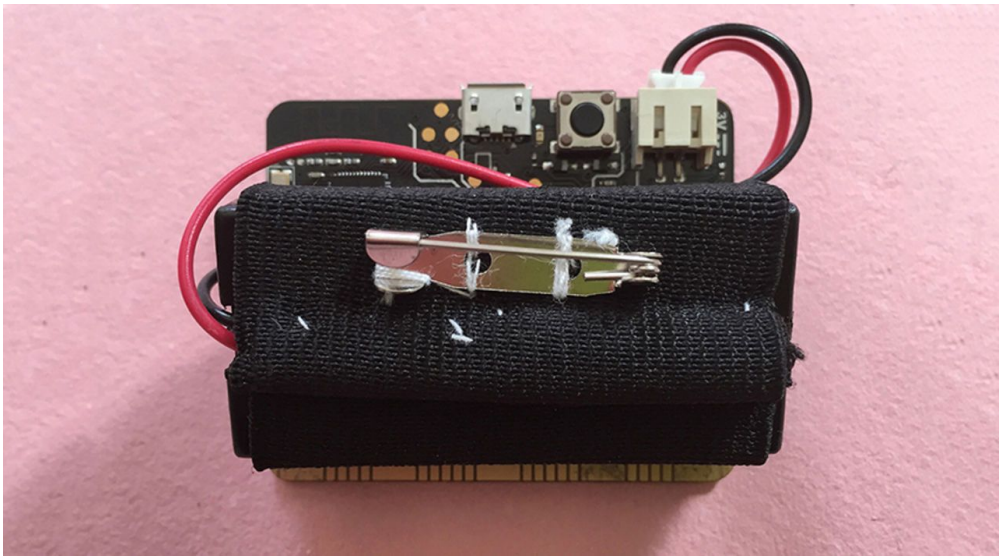
Step 8: Sew a holding stitch.

Knot your thread again and place a stitch on the opposite side to the badge pin in the center edge of the elastic.



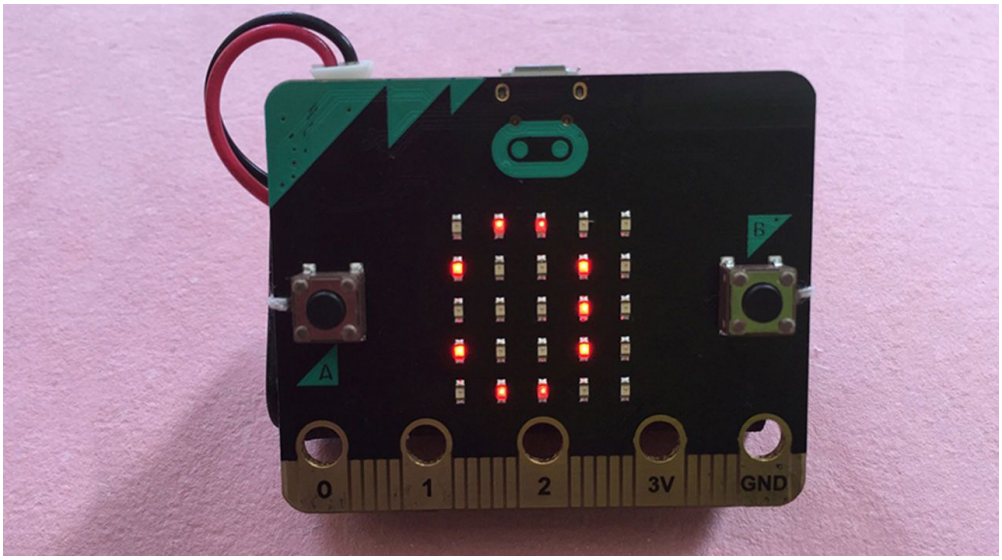
Step 9: Sew elastic to the micro:bit.

Place the tube onto the back of the micro:bit and sew through the holes located on both sides of the micro:bit, right next to the buttons.



Step 10: Plug in the power.

Your micro:bit is now ready to wear. Depending on which code you flashed to it, you can either shake it or press a button to get your fortune!



For more projects you can do with your micro:bit, check out these resources:

- **The MicroPython guide to BBC micro:bit**

  https://www.microbit.co.uk/python-guide

- **MicroPython/micro:bit documentation:**

  http://microbit-micropython.readthedocs.io/en/latest/

- **Micro:bit projects on Instructables:**

  http://www.instructables.com/howto/microbit/