Arduino

Application of CS concepts

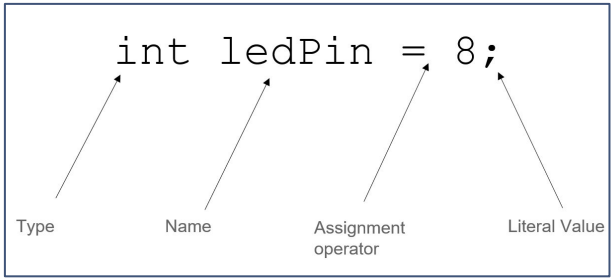
{coding&&community}

Review of Basic Commands

- **pinMode(pin, mode)**
  - Sets pin to either INPUT or OUTPUT
- **digitalRead(pin)**
  - Reads HIGH or LOW from a pin
  - input
- **digitalWrite(pin, value)**
  - Writes HIGH or LOW to a pin
  - output
- **delay(value)**
  - Pauses the program for the amount of time in milliseconds (1/1000th of a second)

Constants:

- **HIGH:** 5V or on
- **LOW:** 0V or off



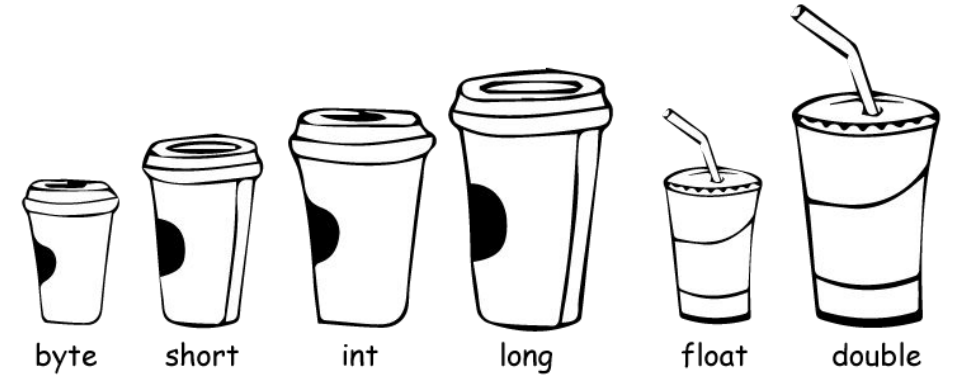
What does this do?

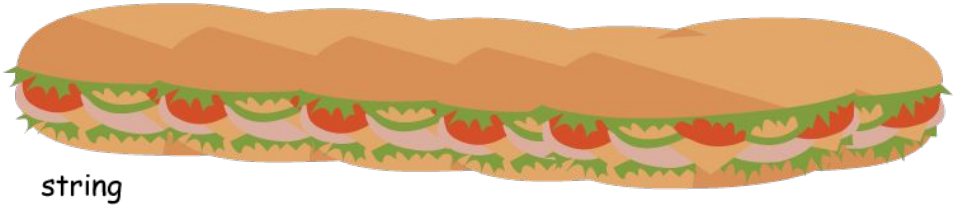
```
int pinLed = 13;
int timeDelay = 1000;

void setup()
{
    pinMode(pinLed, OUTPUT);
}


void loop()
{
    digitalWrite(pinLed, HIGH);
    delay(timeDelay);
    digitalWrite(pinLed, LOW);
    delay(timeDelay);
}
```

Variables





1

Statements and Loops

IF and ELSE Statement

- The **if()** statement allows you to make something happen or not, depending on whether a given condition is true or not.

- There is a common variation called **if-else**.

- There's also the **else if()**, where you can check a second condition if the first is false:

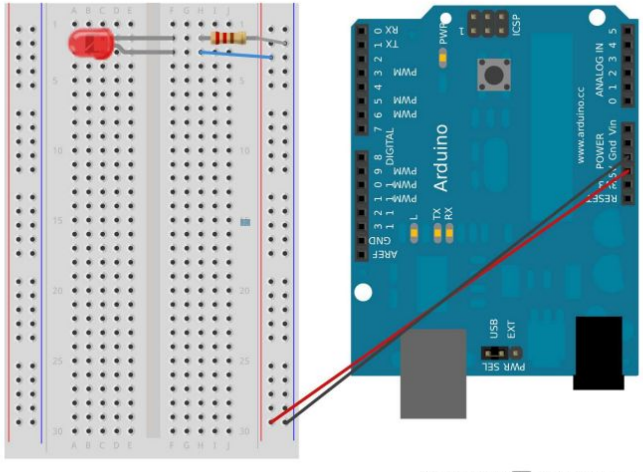
```
if (some condition) {
    //do some stuff if the condition is true
}
```

```
if (some condition) {
    //do some stuff if the condition is true
} else {
    //do some stuff if the condition is false
}
```

```
if (some condition) {
    //do some stuff if the condition is true
} else if (some condition ){
    //do some stuff if the first condition is false
    //and the second condition is true
}
```

Exercise

- Write a program using if and else statements that makes your LED blink faster and faster



Example Answer

```
int pinLed = 13;
int timeDelay = 1000;

void setup()
{
    pinMode(pinLed, OUTPUT);
}


void loop()
{
    if (delayTime >= 0){
        delayTime = 1000;
    } else {
        delayTime = delayTime - 100;
    }
    digitalWrite(pinLed, HIGH);
    delay(timeDelay);
    digitalWrite(pinLed, LOW);
    delay(timeDelay);
}
```

Logical Operator Review

Operator	Example	Meaning
`&&`	`(A < 10) && (B > 5)`	logical AND (return TRUE if condition A AND condition B are true, otherwise return FALSE.)
`||`	`(A < 10) || (B > 5)`	logical OR (return TRUE if condition A OR condition B is true, otherwise return FALSE.)
`!`	`!(A < 10)`	logical NOT (return TRUE if condition A is false, otherwise return FALSE.)

Operator	Meaning
==	is equal to
!=	is not equal to
<	is less than
>	is greater than
<=	is less than or equal to
>=	is greater than or equal to

WHILE Loops

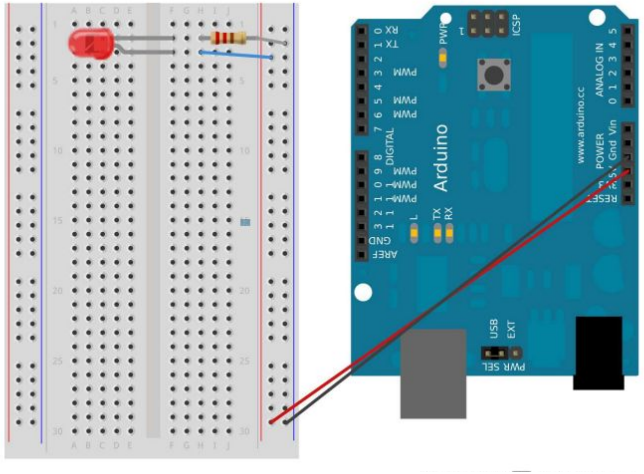
- A **while loop** will loop continuously, and infinitely, as long as the expression inside the parentheses is **true**.
- Something must change the tested variable, or the while loop will never exit.
  - This could be in an incremented variable, or an external condition, such as a button press.

```
while (some condition) {
    //repeatedly do some stuff as long as the
    //the condition is true
}
```

```
var = 0;
while(var < 200){
  // do something repetitive 200 times
  var++;
}
```

Exercise

- Write a program using a while loop that makes the LED blink faster and then slows down.



Example Answer

```
int pinLed = 13;
int delayTime = 1000;

void setup()
{
    pinMode(pinLed, OUTPUT);
}

void loop()
{
    while (delayTime > 0) {
        digiatalWrite(pinLed, HIGH);
        delay(delayTime);
        digitalWrite(pinLed, LOW);
        delay(delayTime);
        delayTime -= 100;
    }
    while (delayTime < 1000) {
        delayTime += 100; //do this first so we don't have
                          //a loop with delayTime = 0
        digitalWrite(pinLed, HIGH);
        delay(delayTime);
        digitalWrite(pinLed, LOW);
        delay(delayTime);
    }

}
```

FOR Loops

- The **for loop** is used to repeat a code block a set number of times.
- The **initialization** happens first and exactly once. Each time through the loop, the **condition** is tested; if it's true, the statement block, and the **increment** is executed, then the condition is tested again. When the condition becomes false, the loop ends.

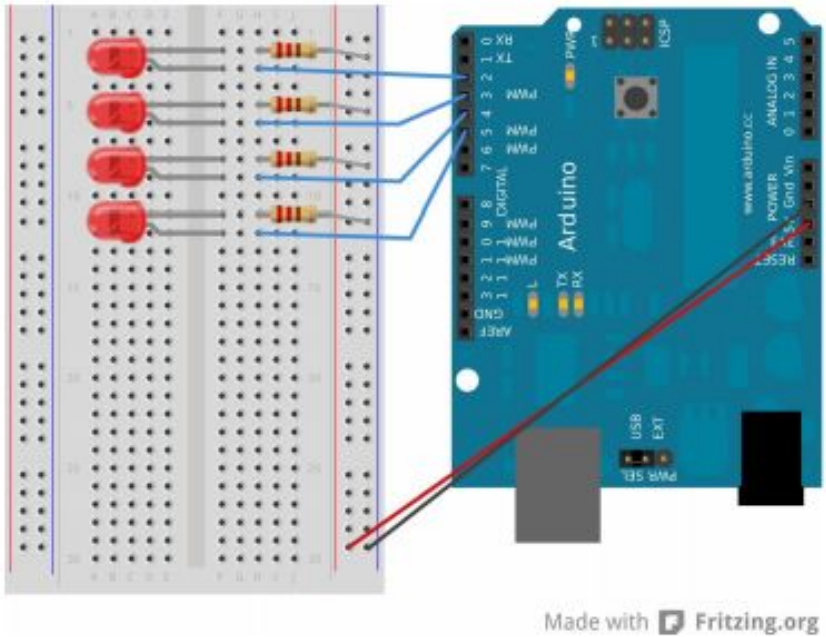
```
for (initialization; condition; increment) {
    //statement(s);
}
```

```
for(int i = 0; i < 4; i++){
    digitalWrite(kPinLed, HIGH);
    delay(200);
    digitalWrite(kPinLed, LOW);
    delay(200);
}
```

2

Statements and Loops

New Circuit!



Exercises

- Make a program (sketch) that lights up a single LED five times in a row for one second on and off, and then five times in a row for ½ of a second on and off.
- Make a program using arrays that lights up the LEDs from top to bottom and then goes backwards so only one LED is on at any time.
- Make a program that lights up the LEDs in any pattern that you like.