CPSC 1000: Introduction to Computer Science

by making Arduino projects

Robert Benkoczi, C556
robert.benkoczi@uleth.ca

11-Sep-2018

Course goals

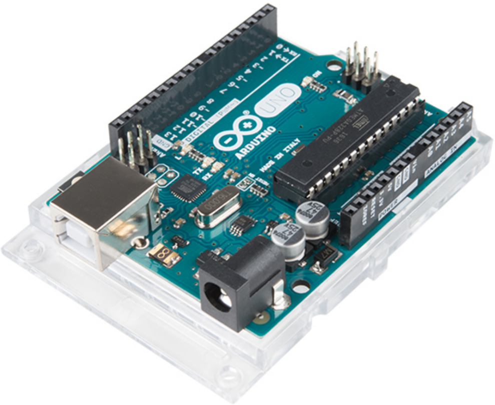
At the end of the course, students will know to make Arduino projects.

See “Arduino” on youtube.

What is an Arduino?

= Micro-controller, a computer that interfaces with electronic components such as

- LED (Light Emitting Diodes)
- sensor: distance, temperature, light
- motors, valves, relays ....



Course content

- Programming an Arduino micro-controller. → week 1
- Process input from sensors. } weeks 2-5
- Generating signals.

Week 1 lecture objectives

To learn some basic programming for the Arduino micro-controller.

- Functions / procedures.
- Communication through the serial interface.
- Expressions and arithmetic operations.
- Variables.
- Branching, if statements.
- Repetition.

Arduino simulator: `tinkercad.com`

Functions & communication

→ Arduino programs contain two mandatory functions { setup.
loop.

<u>Def</u> Function in programming: set of commands that are given a name.

Optional: – data for input
– result may be returned

<u>Ex</u>: please give me a glass of water ← name

- get up
- turn right
- grab a glass ...

← set of commands.

Functions (c'ed)

Syntax

FOR FUNCTION DEFINITION

TBD, for now, empty

↑

type name of function (input) {

(void) ( )

↑ beginning of instructions.

— instruction 1

— instruction 2

...

} ← end of the instructions

Examples

- define setup & loop functions (mandatory functions)

setup: executed once at beginning

loop: executed repeatedly "forever".

```
void setup ( ) {        ← start of code     } HEADER
                                            
    Serial. begin(9600);                    } BODY
    Serial. println("Hello");
}  ↖ end of code
```

void ↗ no return value (no calculation)

( ) ↖ no input

FUNCTION CALL (EXECUTE FUNCTION)

FUNCTION CALL :

Syntax :

name of function (input values) $\underbrace{;}_{\text{end of instruction}}$

<u>Examples</u>   delay(100);

Calling hello :

```
hello();
```

Definition :

```
void hello ( ) {
        Serial. println("Hello");
        delay (1000);
    }
```

BIG PICTURE (the .ino code)

```
     void setup () {
            Serial.begin(9600);
    }

     void loop ()  {
            hello ();
   }
   void hello () {
     Serial...
      ....
  }
```

Expressions

- example: 2+3
- function that calculates 1+2+3+...+n.

  Example of definition:

```
    int sum(int n ) {
   return n*(n+1)/2;                  sum ( sum(7));
 }
```

← expressions

Expressions using sum: `sum(100) + sum(1000);`

Expression = something that has a value

Expressions (c'td).

-> we used expressions as argument (input) to println() function.

ex: Serial.println($\underbrace{200 + \underbrace{7/2}_{\text{quotient from integer division when arguments are integers.}}}_{203}$);

$$\begin{array}{c}200 + 7.0/2\\ \downarrow\\ 203.5\end{array}$$

Variables

= symbols that hold value.

Variables → have a name } declaration
→ have a type }
→ value } assignment.

Syntax: type name;

name = value;

Example

```
int n;
n = 2;
```

OR

```
int n = 2;
```

Variables can appear in expressions.

Ex: (suppose n is an integer variable with value 2)

```
Serial.println((n+2)*(n+2));   // Produces 16 on serial monitor.
int x = n+1;
n = n+1;     ← increases the value of n by one.
n++;         ←  —||—  (shortcut notation)
```

Examples of variables that hold strings (text)

String name; ← declare the variable to hold text

name = "John Doe" ; ← text values (literals) are enclosed between " "

Serial.println(name[0]); → prints J, the first letter in the string

↑ zero

(name[1] → second letter, or 'o')

Obs operator + on strings, concatenates strings

name = name + " great" ; → name equals now "John Doe great".