Arduino Cheat Sheet

Mostly adapted from the Arduino Language Reference:
http://arduino.cc/en/Reference/HomePage

Structure & Flow

Basic Program Structure

```
void setup() {
  // runs once when sketch starts
}
void loop() {
  // runs repeatedly
}
```

Control Structures

```
if (x < 5) { ... } else { ... }
while (x < 5) { ... }
do { ... } while ( x < 5);
for (int i = 0; i < 10; i++) { ... }
break;  // exit a loop immediately
continue;  // go to next iteration
switch (myVar) {
   case 1:
     ...
     break;
   case 2:
     ...
     break;
   default:
     ...
}
return x; // or "return;" for voids
```

Operators

General Operators

```
=   (assignment operator)
+   (add)          -   (subtract)
*   (multiply)     /   (divide)
%   (modulo)
==  (equal to)     !=  (not equal to)
<   (less than)    >   (greater than)
<=  (less than or equal to)
>=  (greater than or equal to)
&&  (and)     ||  (or)     !  (not)
```

Compound Operators

```
++  (increment)
--  (decrement)
+=  (compound addition)
-=  (compound substraction)
*=  (compound multiplication)
/=  (compound division)
&=  (compound bitwise and)
|=  (compound bitwise or)
```

Bitwise Operators

```
&  (bitwise and)    |  (bitwise or)
^  (bitwise xor)    ~  (bitwise not)
<< (shift left)     >> (shift right)
```

Variables, Arrays, and Data

Data types

```
void
boolean   (0, 1, true, false)
char      (e.g. 'a'  -128 to 127)
int       (-32768 to 32767)
long      (-2147483648 to 2147483647)
unsigned char   (0 to 255)
byte            (0 to 255)
unsigned int    (0 to 65535)
word            (0 to 65535)
unsigned long   (0 to 4294967295)
float     (-3.4028e+38 to 3.4028e+38)
double    (currently same as float)
```

Qualifiers

```
static   // persists between calls
volatile // use RAM (nice for ISR)
const    // make read only
PROGMEM  // Use flash
```

Arrays

```
int myInts[6]; // array of 6 ints
int myPins[]={2, 4, 8, 3, 6};
int mySensVals[6]={2, 4, -8, 3, 2};
myInts[0]=42;  // assigning first
               // index of myInts
myInts[6]=12;  // ERROR! Indexes
               // are 0 though 5
```

Constants

```
HIGH | LOW
INPUT | OUTPUT
true | false
143         // Decimal
0173        // Octal (leading 0)
0b11011111  // Binary
0x7B        // Hex (hexadecimal)
7U          // force unsigned
10L         // force long
15UL        // force long unsigned
10.0        // force floating point
2.4e5       // 240000
```

Pointer Access

```
&  (reference: get a pointer)
*  (dereference: follow a pointer)
```

Strings

```
char S1[8] =
 {'A','r','d','u','i','n','o'};
 // unterminated string; may crash
char S2[8] =
 {'A','r','d','u','i','n','o','\0'};
 // includes \0 null termination
char S3[]="arduino";
char S4[8]="arduino";
```

Built-in Functions

Digital I/O

```
pinMode(pin,[INPUT, OUTPUT])
digitalWrite(pin, value)
int digitalread(pin)
  // Write HIGH to inputs to use
  // pull-up resistors
```

Analog I/O

```
analogReference([DEFAULT,
    INTERNAL, EXTERNAL])
int analogRead(pin)
  // Call twice if switching pin
  // from a high-Z source
analogWrite(pin, value) // PWM
```

Advanced I/O

```
tone(pin, freqhz)
tone(pin, freqhz, duration_ms)
noTone(pin)
shiftOut(dataPin, clockPin,
  [MSBFIRST,LSBFIRST], value)
unsigned long pulseIn(pin,
  [HIGH,LOW])
```

Time

```
unsigned long millis()
  // overflow at 50 days
unsigned long micros()
  // overflow at 70 minutes
delay(ms)
delayMicroseconds(us)
```

Math

```
min(x, y)    max(x, y)    abs(x)
sin(rad)     cos(rad)     tan(rad)
sqrt(x)      pow(base, exponent)
constrain(x, minval, maxval)
map(val, fromL, fromH, toL, toH)
```

Random Numbers

```
randomSeed(seed) // long or int
long random(max)
long random(min, max)
```

Bits and Bytes

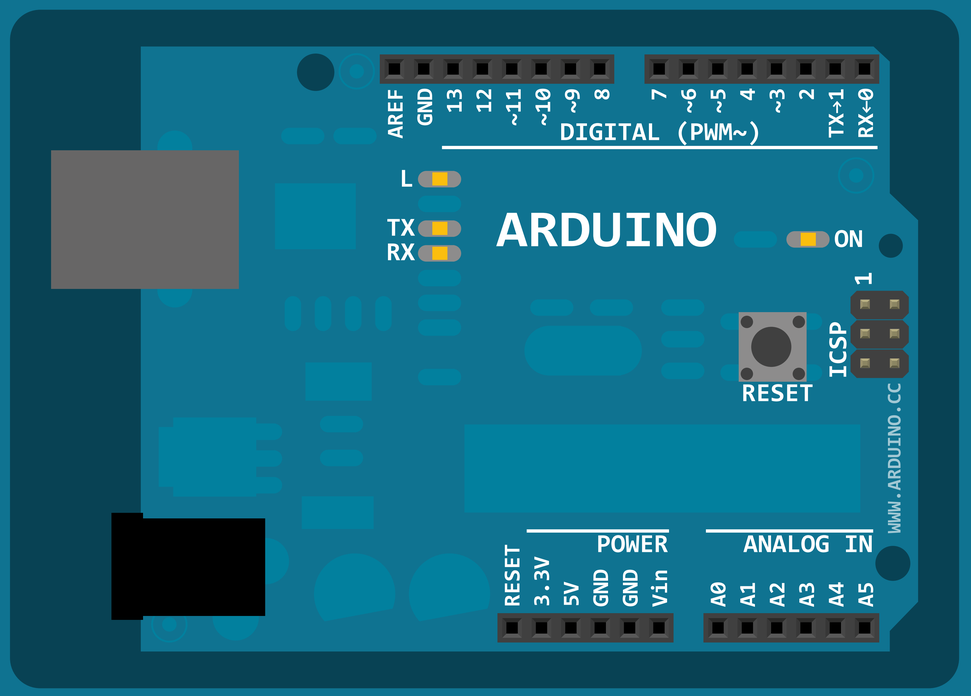
```
lowByte(x)    highByte(x)
bitRead(x, bitn)
bitWrite(x, bitn, bit)
bitSet(x, bitn)
bitClear(x, bitn)
bit(bitn)  // bitn: 0=LSB 7=MSB
```

Conversions

```
char()        byte()
int()         word()
long()        float()
```

External Interrupts

```
attachInterrupt(interrupt, func,
 [LOW, CHANGE, RISING, FALLING])
detachInterrupt(interrupt)
interrupts()
noInterrupts()
```



Libraries

Serial

```
begin([300, 1200, 2400, 4800,
       9600, 14400, 19200, 28800,
       38400, 57600, 115200])
end()
int available()
byte read()
byte peek()
flush()
print(myData)
println(myData)
write(myBytes)
flush()
```

EEPROM (#include <EEPROM.h>)

```
byte read(intAddr)
write(intAddr, myByte)
```

Servo (#include <Servo.h>)

```
attach(pin, [min_uS, max_uS])
write(angle) // 0, 180
writeMicroseconds(uS)
   // 1000-2000; 1500 is midpoint
read()        // 0 - 180
attached()    // returns boolean
detach()
```

SoftwareSerial(RxPin, TxPin)

```
  (#include <softwareSerial.h>)
begin(long Speed) // up to 9600
char read()   // blocks till data
print(myData)
println(myData)
```

Wire (#include <Wire.h>) // for I²C

```
begin()      // join a master
begin(addr) // join a slave @ addr
requestFrom(address, count)
beginTransmission(addr) // Step 1
send(myByte)            // Step 2
send(char * mystring)
send(byte * data, size)
endTransmission()       // Step 3
byte available() // Num of bytes
byte receive() // Return next byte
onReceive(handler)
onRequest(handler)
```