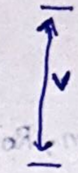
Arduino

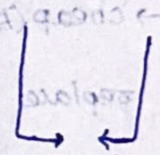
* Electricity: flow / buildup in a place - Static (fixed), current (moving)
* Static: painful jolt, crackling sound - charges accumulate at a place. (lightning)
* Ions - when it loses e^- or protons.
* current - dynamic.
* Voltage - force that makes e^-s move. (volts - V) - Emf (drives e^-s)
* Voltage - Diff b/w two (potential differences) terminals.

Water analogy



* more - tank - more pressure (Battery)
* current - rate of flow of charge (Ampere A)
* Amount of electric charge that flows when 1 coulomb of charge moves past somewhere in one second. → 1A

*current → AC (sinusoidal change)
→ DC (same direction + steady voltage)



Total = more + less.

*

Resistance

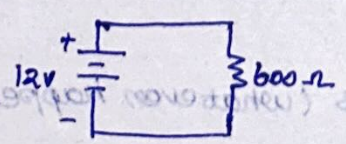
* measure of difficulty of passing electric current.
* more e^-s → high conductivity.
* Resistor - limit current - save LED from burning!

→ current - choose least resistive path

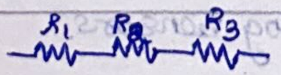
more force → more current

→ $V = IR$.

* Voltage keep pushing the current
* Resistance creates difficulty!



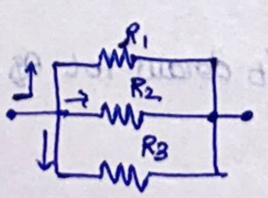
$I = \frac{V}{R} = \frac{12}{600} = 0.02$
$= 20mA$



$= R = R_1 + R_2 + R_3$ (same current)

$V = V_1 + V_2 + V_3$ (Across each resistor)

$\boxed{R_1 + R_2 + R_3 = R}$

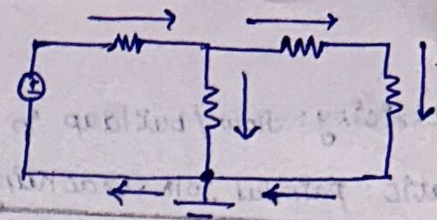


$I = I_1 + I_2 + I_3$ (multiple path)

$\frac{V}{R} = \frac{V}{R_1} + \frac{V}{R_2} + \frac{V}{R_3}$

$\boxed{\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3}}$

* Kirchoff's law → KVL
→ KCL



4 band → 1, 2 → digits 3 → no. of zeros 4 → error %.

5 band → 1, 2, 3 → digits 4 → " 5 → Tolerance.

	B	B	R	O	Y	G	B	V	G	W
Black ↓ Brown	0	1	2	3	4	5	6	7	8	9

gold → 5 %

Silver → 10 %

Brown → 1 %

Red → 2 %

* CPU - RAM, ROM (eg: CPU - a MC)

* Micro processors - Brain of any computer (CPU - Computation - need RAM, ROM) - heavy.

* micro controllers - IC - tiny computers - Control motors, display etc.. - low voltage req. memory, peripherals - same chip

→ cheap (than micro processors)

* mc can't replace micro processor.

Arduino → Circuit board having micro controller.

* ATMEGA 328 - UNO (Arduino - Italian company) - Opensource. (Clone & modify)

* Easy to use (no tech background).

* 1) Digital pins → 0 to 13

* ~ 3 (special symbol) → support PWM

* Also takes dig i/p from sensors & gives dig o/p

2) Builtin LED → connected to 13 pin (high - 13 - LED ON)

3) power LED → plugged in

4) ATMEGA 329 Chip - All main functions (whatever happening)!

5) Analog pins - A0 to A5 (6 pins) - connect analog sensors.

6) power pins → 5V

→ GND

→ GND

] make sure don't draw lot of current.

→ Power dc Jack → (up to 7 volts - 12 volts)

Tx → Transmit
Rx → Receive
LEDS

$p\left(\frac{1}{x}\right) + (1-p)\left(\frac{1}{1-x}\right)(-1) =$

5) USB Jack - Connect with computer

6) Reset - Reload entire code - doesn't clear memory

Specs

* ATMEGA 328 (5V) - operating voltage
* 7-12 V (i/p) - recommended
* Limit : 6 - 20V
* 14 Dig i/o pins - 6 PWM o/p
* 6 Analog pins

* 6 Analog pins - Connected to Arduino through 10bit ADC
* I/O pin DC current : 40mA (can provide)
* Don't run - motors
  few LEDS, sensors (fine)

* Also 3.3V - max current 50mA (Available)
* Flash memory - 32KB with 0.5KB used by bootloader - more than enough
* SRAM - 2KB (ATMEGA)
* 1KB EEPROM
* Clock speed - 16MHZ

Arduino IDE

* window → port → com3
* mac → /dev/cu.usbmodem14101...
* Linux → /dev/ttyACM0

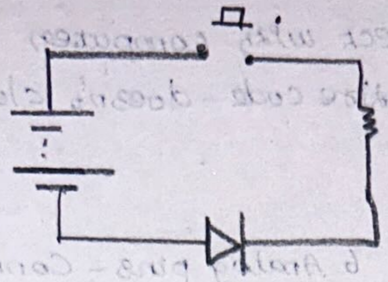
* Upload → see!

Autodesk, Tinkercad

```
void setup ()
{
  pinMode (13, OUTPUT);
}
void loop ()
{
 digitalWrite (13, HIGH);
 delay(1000);
 digitalWrite (13, LOW);
 delay(1000);
}
```

* Bread board: * Non permanent Connections

* ⋮ and .... (Gnd) → 5 Connected.



Precautions

1) Don't use >5v in any i/o pin

2) 5v to GND — Don't!

3) Take Care of polarity!

A) Caution with external power. (<12-advise)

I/p and o/p

* Sensor data ⟶ Arduino ⟶ o/p

* Sensors: (Environment data) - Arduino — according to change in v of sensors - make actions

* Rotatory potentiometer
* momentary push button
* Force sensing resistor
* LDR
* Triple axis accelerometer
* IR sensor.
* Joystick — i/p controller (games)
* microphone / Sound sensor
* Rotary encoder (uses UV sensor)
* Touch sensor
* Temperature sensor
* Humidity sensor

* o/p → Buzzer, speaker, RGB LED, LED, OLED display, LCD display, Servo motor, dot matrix DC motor, stepper motor, servo motor.

* Dig: digital - voltage changes with time
* Sound - type of analog signal!
* Get started!

mixture of C & C++ (wiring)

1) void setup() { ⟶ mandatory function. (only once)
}

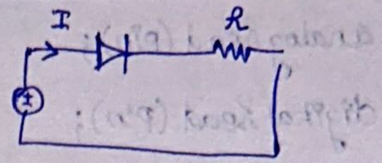
2) void loop() { ⟶ repeatedly run.
}

// comment.

* LED → longer lead (+ve)

pinMode (____ , ____) ;
↓ pin ↓ configuration

eg: pinMode (13, HIGH);
↳ LED_BUILTIN



*/ */
multiline comments

* for some time → ON → delay (1000) ; → 1second.

* int →

* float

* bool

→ In serial monitor.

```
Serial.begin(9600);
Serial.print();
Serial.println();
```

Serial.begin(9600); → get ready to exchange messages with the serial monitor at a data rate eg 9600 bits/second.

eg: bool num = 123;
Serial.print(num); → 1 (123 → true)

byte — 0 to 255 (8 bits)

byte x = 158;

* Char : character
→ "hold" – a single character.
→ char x = 'a'; → still stored as a number (ASCII)

char mychar = 65;
Serial.print(mychar); → A

$\boxed{\text{char + int = int}}$

* const int Myvariable = 5; → define constant.

#define pi 3.14

'static' – preserve data b/w function calls

* <, >, <=, >=, ==, != (Relational)

* Logical → &&, ||, !

* if { } | else | if else if if | if if

analogRead (pin);

digitalRead (pin);

analogWrite (pin, —);
→ HIGH / Low

digitalWrite (pin, —)

→ No relation with analogread / analog pins.

"uses — 3, 5, 6, 9, 10, 11 → PWM capable digital pins.

490Hz (pins 5, 6) → 980Hz

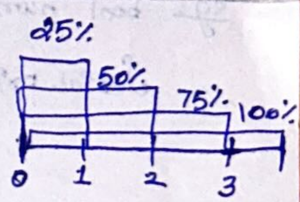
↓

3, 5, 6, 9, 10, 11 pins

AnalogWrite → generates square wave

eg: control speed of motor.

Duty cycle: fraction of time — ON



analogWrite (0) → 0% duty cycle

analogWrite (64) → 25%

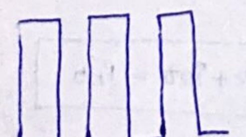
analogWrite (127) → 50%

analogWrite (191) → 75%

analogWrite (255) → 100%

PWM — technique — get analog results with digital means.

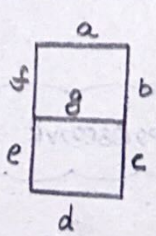
PWM frequency.

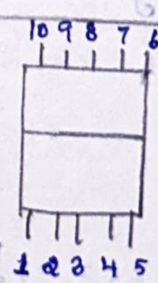


"Duty cycle — same"

analogWrite (10, 50);
↳ pin
↳ how much

(say: 64 ↳ 25%)





4 → c

1 → e

2 → d

3 → com

8 → com

6, 7 → b, a

10, 9 → g, f.

5 → dot

g, f ←

a, b →

e, d ←

c →

2 → a 5 → d 8 → g.
3 → b 6 → e
4 → c 7 → f

2-a	5-d	8-g
3-b	6-e	9-dot
4-c	7-f	

8 → All ON.

$\boxed{\text{dot → only 5 on}}$ → over: 9 on

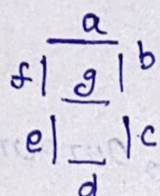
Common Cathode → g f Gnd a b
e d Gnd c dot

CA → Common anode.

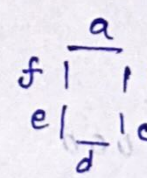
Common anode → g f Vcc a b
e d Vcc c dot.

Common anode
Cathode → ON
Anode → OFF

(DP) G F E D C B A
1 0000000



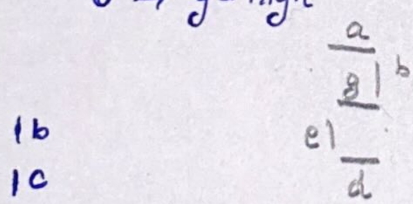
0 → a, b, c, d, e, f. (ON)
g → off
(cathode)



$\boxed{0}$

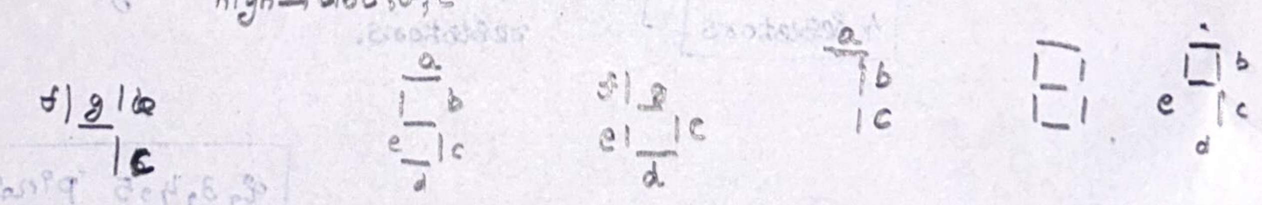
ANODE: - what I have! (High-OFF)

0 → g - high



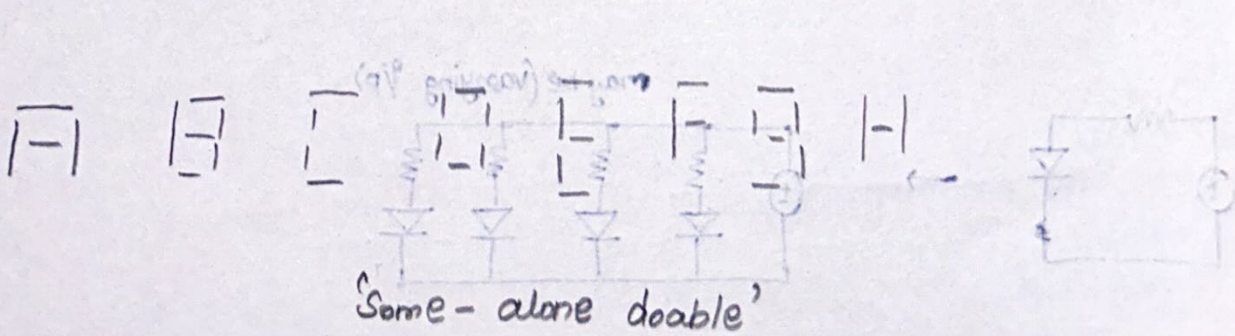
$\boxed{1}$

high → dot, f, e



ON → d, e, a

b, e



'Some - alone doable'

* Builtin - library
  - SPI
  - Servo
  - SD
  - EEPROM
  - Ethernet
  - Software Serial
  - WiFi
  - Firmata
  - GSM
  - LiquidCrystal
  - TFT
  - Stepper
  - Wire.

#include <LiquidCrystal.h>

#include <SD.h>

#include <DHT.h>

#include <...>

See manage library' → to install.

'we can add our libraries too'

---

① void setup → only once

② void loop → Repeatedly.

③

① pinMode (13 , OUTPUT);

② digitalWrite (13 , HIGH/LOW);

③ delay (1000);

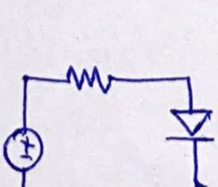
④

0	0000
1	0001
2	.
3	.
4	.
:	:
15	1111

4 LEDS
4 Resistors

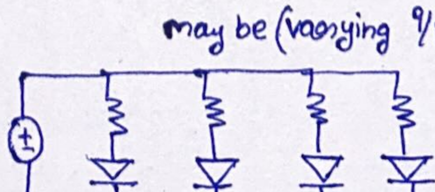
each LED - a current limiting resistors.

2, 3, 4, 5 pins.



→

may be (varying i/p)



0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

$\boxed{A3 \rightarrow \text{Analog } 3}$ readPin = A3;

```
pinMode (readpin, INPUT);

V2 = AnalogRead (readpin); → loop

Serial.begin (9600);
Serial.Println (V2);

delay(1000);
```

AnalogRead → doesn't read actual voltage → a scaled voltage.

b/w (0 to 1023)

$5 \rightarrow 1023$

$0 \rightarrow 0$

$v_2 = 2^{2 \times n}$

$\log_2 (v_2) = n \log_2 2$

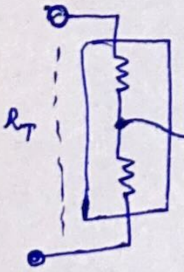
$\boxed{n = \log_2 (v_2)}$

(or)

$\frac{Val}{1023} \times 5$ (convert to 5)

$2^2 = 4$
$2^4 = 16$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$
$2^9 = 512$
$2^{10} = 1024.$

Potentiometer



$R_T = R_1 + R_2$

Left

G

$R_2 = 0$

$\boxed{R_T = R_1}$

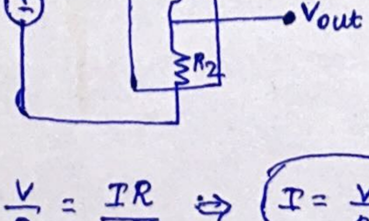
Right

$R_2 = R_T$

$\boxed{R_1 = 0}$

① $R_1 = 5K\Omega$, $R_2 = 5K\Omega$ say!

'many cases'



(Left)

$\boxed{V_o = I . R_2}$

$\boxed{V_o = 0}$ If $R_2 = 0$

Right

$\boxed{V_o = 10K (I)}$

$= 5A.$

$\frac{V}{R} = \frac{IR}{R} \Rightarrow I = \frac{V}{R} = \frac{5}{10000} = 0.5mA$

'Controlling LED brightness'

Read from monitor

→ wait until → i/p is received → ∞ loop

Serial.available() → 0 (No)
1 (Yes)

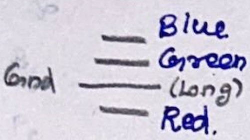
get float:

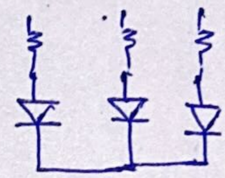
```
Serial.parseFloat();
Serial.ParseInt();
```

Read strings

```
String myName;
String msg = "What's your name:";
```

```
Serial.readString()
```





RGB LED

RGB colors

analogWrite (10, 255); → PWM pins.

$\boxed{\text{RGB}}$ → 3 terminals, 3pins

'colors'!

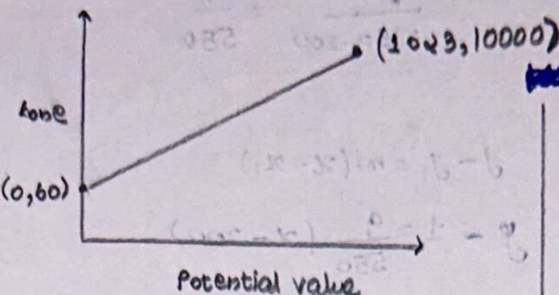
Buzzers

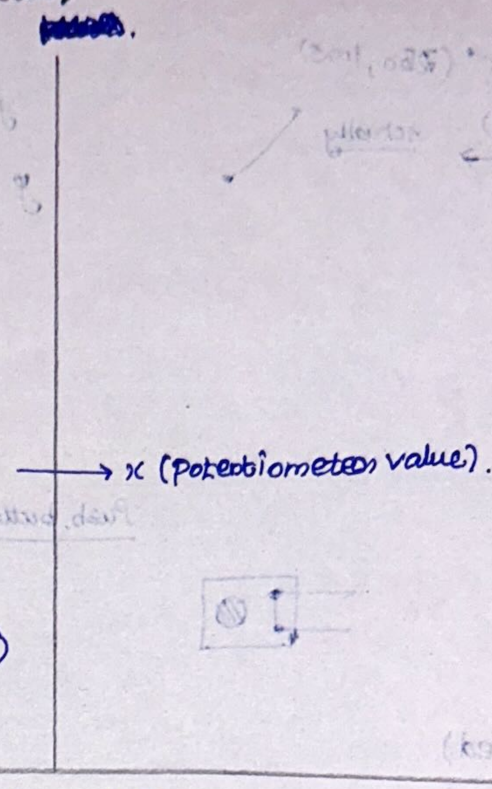
Changing tone - Active buzzer.

tones: (manually generate)

```
for (i=1; i<=100; i++) {
  digitalWrite (8, High);
  delay (1);
  digitalWrite (8, LOW);
  delay (1);
}
for (i=1; i<=100; i++){
  digitalWrite (8, HIGH);
  delay (2);
  digitalWrite (8, LOW);
  delay (2);
}
}
```

* delayMicroSeconds(__); → delay in microseconds





$$\frac{10000-60}{1023} = \frac{99940}{1023}$$

$$\boxed{y-y_1=m(x-x_1)}$$

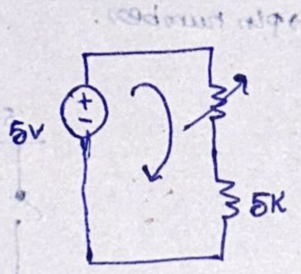
$$y-60=\frac{99940}{1023}(x-0)$$

$$\boxed{y=60+\frac{99940}{1023}x}$$ tone

Photoresistor – LDR

L ↑, Resistance ↓

L ↓, Resistance ↓



$$I=\frac{5}{R_S+5K}$$

$\boxed{L\uparrow,\ R_S\downarrow}$, $I\uparrow$

$L\downarrow,\ R_S\uparrow$, $I\downarrow$

$$V = IR$$

$$V_{5K}=\left(\frac{5}{R_S+5K}\right)\cdot 5K$$

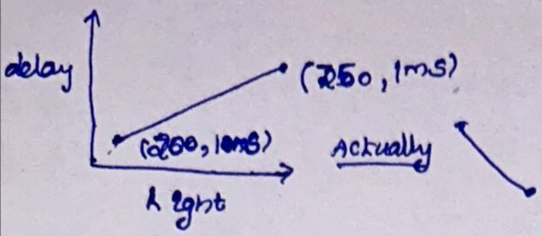
$$\boxed{V_{5K}=\frac{25K}{R_S+5K}}$$

analogRead < 350 → (meaning high resistance)

'Need to ON light'

750 → ON → 1msec

200 → OF → 10msec



$$\frac{10-1}{750-200} = \frac{9}{550}$$

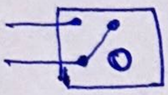
$$y - y_1 = m(x - x_1)$$

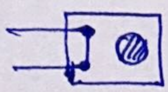
$$y - 1 = \frac{9}{550}(x - 200)$$

$$y = \frac{9}{550}x - \frac{200(9)}{550} + 1$$

$$= \frac{9x}{550} - \frac{1800 + 550}{550}$$

Push buttons

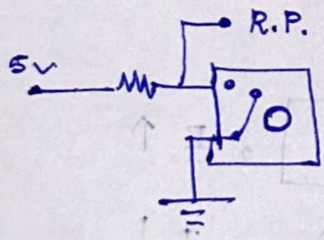




'open'
(not pushed)

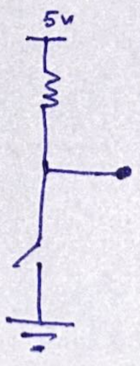
» digitalRead( );
→ pin number.

Pull up resistor



ON → 0V

OFF → 5V

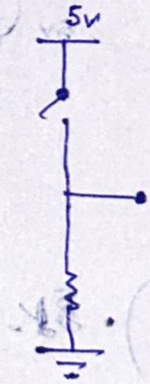


pull up

when

Switch off → 5v

ON → 0v



pull down

OFF → 0V

ON → 5V

'Toggle switch'

0 → 1

1 → 0

* Take care of old value too!

Servo meter - change angle

* No rotation - but just angular turning.

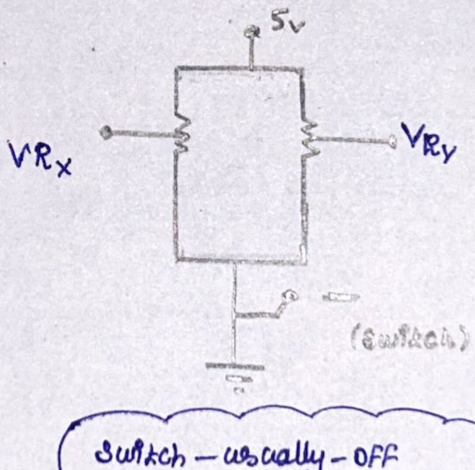
```
#include <Servo.h>
int ServoPin = 9;
int ServoPos = 0;
Servo myServo;  → creating a virtual object.

void setup() {
   Serial.begin(9600);
   myServo.attach(servoPin);
}
void loop() {
   myServo.write(servoPos);
}
```

90°
180°] 0 to 360° → i/p.

LDR L↑, R↓, V↑ → day
L↓, R↑, V↓. → Night.

Joystick



Potentiometers

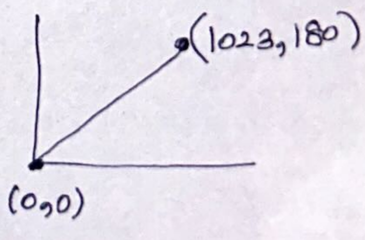
1) VCC
2) Gnd
3)] voltage across
4)] potentiometers
5) Switch.

① Analogread → VRx
→ VRy

② digitalread → switch

switch – usually – OFF
click – open & close switch

	Read (Potentiometer)	Write (Servo)
Read (Analog)	0	0°
	1023	180°



$$\frac{180-0}{1023-0} = \left(\frac{180}{1023}\right)$$

$$y - y_1 = m(x - x_1) \Rightarrow \boxed{y = \left(\frac{180}{1023}\right)x}$$

Potentiometer → 1 → To Servo 1

2 → To Servo 2

```
xval = analogRead (xpin);
yval = analogRead (xpin);
xval = (180./1023.) * xval;
yval = (180./1023.) * yval;
XServo.attach (Xspin);
YServo.attach (YPin);
XServo.write (xval);
YServo.write (yval);
```

* Stepper motor: Tricky, different (one to another)
* Need driver.

String

```
>> char str[6];
>> char str[] = "I like Coffee and cake";
>> Serial.begin(9600)
>> Serial.println(like);
```

```
strlen()
sizeof()
strcpy(out_str, str);
```

copy

```
strcat(out_str, " sketch.");
```

Append at its end.

String object – Construct having both data & functions

my_str = "This is my String";

1) Serial.println(my_str);

2) my_str.toUpperCase();

3) my_str = "mynew string"; → overwrite

4) my_str.replace("string", "mod"); → This is my mod.

5) my_str.length() → gives length

time

delay()

delayMicroSeconds()

millis() → return time in ms (arduino running)

micros() → returns no. of micro seconds from the time arduino started work (overflow: above 70 minutes)

Array

int c[3] = {1, 2, 3};

I/O functions

pinMode()

digitalWrite()

analogRead()

analogReference() → Configures reference voltage used for analog I/P.

Character functions

c → letter

int isdigit (int c);

int isalpha (int c);

int isalnum (int c);

int isxdigit (int c); → hexadecimal

islower()

isupper()

isspace()

iscntrl() → control character \n, \f

ispunct() → printing char - other than digit / space / letter

isprint() → Printing char including ' '

isgraph() → printing char other than ' '

math.h

cos, fabs, fmod, sin, sqrt, tan, exp, atan, atan2, log, log10, pow, square

Trignometry

sin, cos, tan, acos, asin, atan

Random number

randomSeed (seed)

random()

Keyboard – library   mouse library

>> Serial.print()

>> Serial.print(25, BIN) → 11001

>> Serial.print(58, HEX) → 3A

>> Serial.print(58, OCT) → 72

>> Serial.print(25, DEC); → 25

Flash memory based Strings

Serial.print(F("Hello"));

print tab space: \t

println → newline

Serial.available() → UNO

Serial1.available()
Serial2.available()
Serial3.available()
] Mega

>> Serial.read() → read incoming bytes

>> Serial.readString() → reads incoming serial data from serial buffer in the string.

>> Serial.write() → sends binary data to serial port in arduino (In bytes)

Manual

write(str) / write(value)

write(buffer, len)

AnalogRead()

* Operating voltage – 0 to 1023 (Scaled) – 10 bits resol

>> analogReference(type) → we can change reference type (resolution) ←

>>

Datatypes

* void
* int
* char
* float
* double
* unsigned int
* short
* long
* unsigned long
* byte
* word