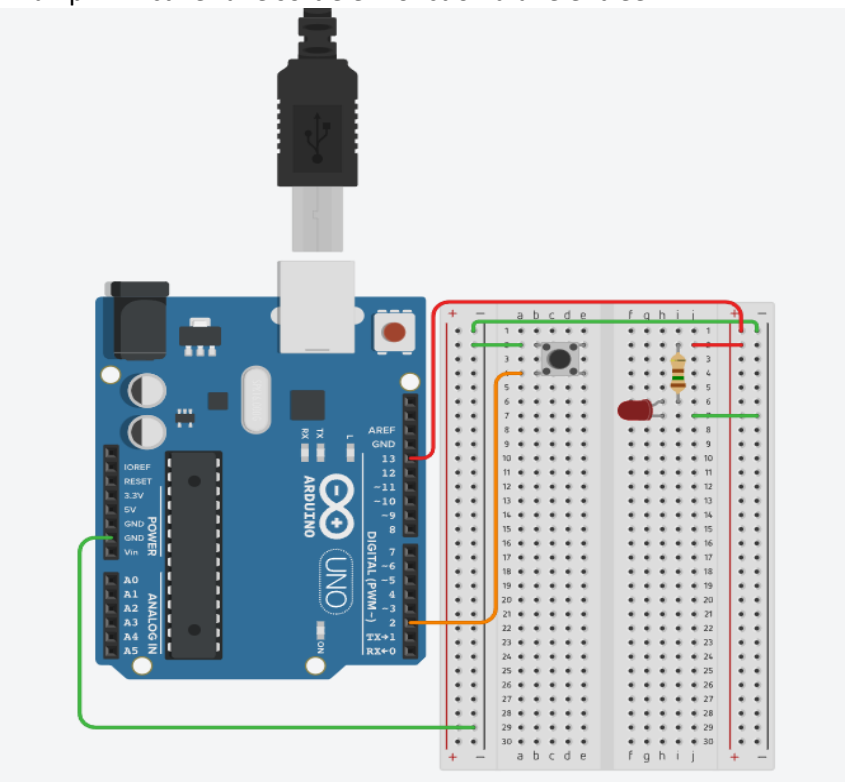
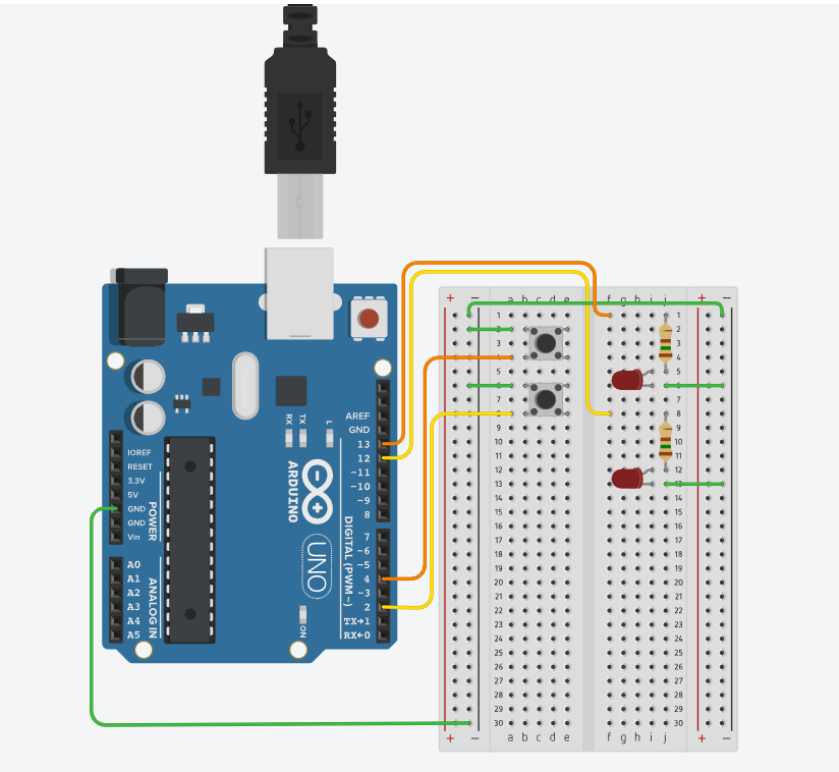
.Manip 1 : Activer une sortie en fonction d’une entrée



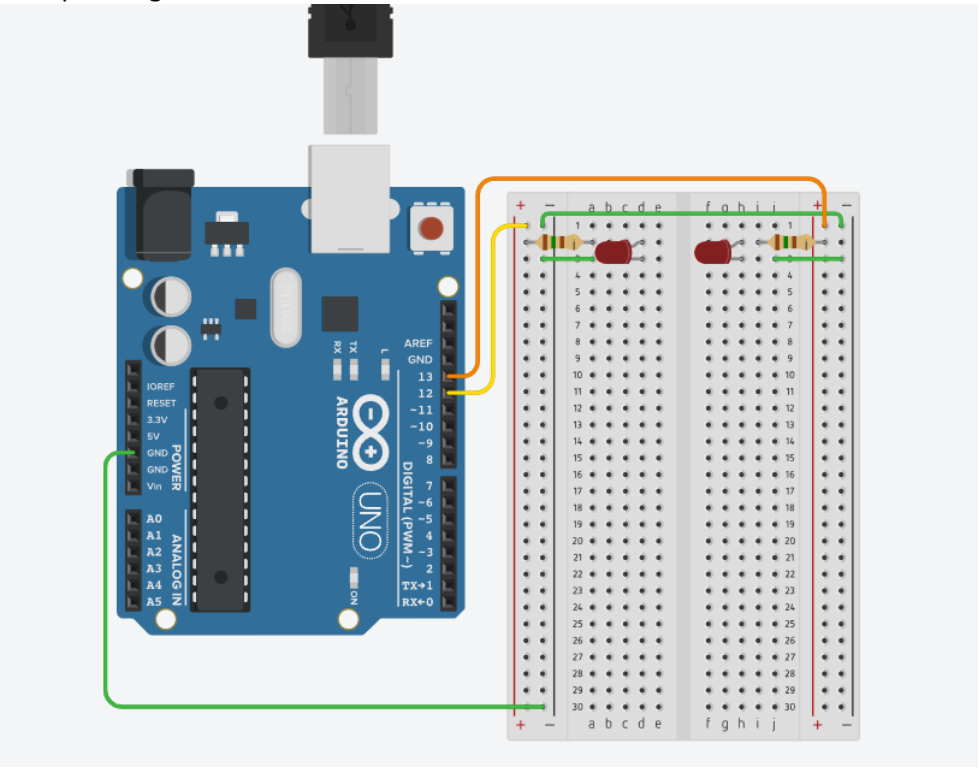
```
#define BP 2
#define LED 13

void setup() {
  pinMode(BP, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  boolean etat = !digitalRead(BP);
  digitalWrite(LED, etat);
}
```

Manip 2 : Activer plusieurs sorties en fonction de plusieurs entrées



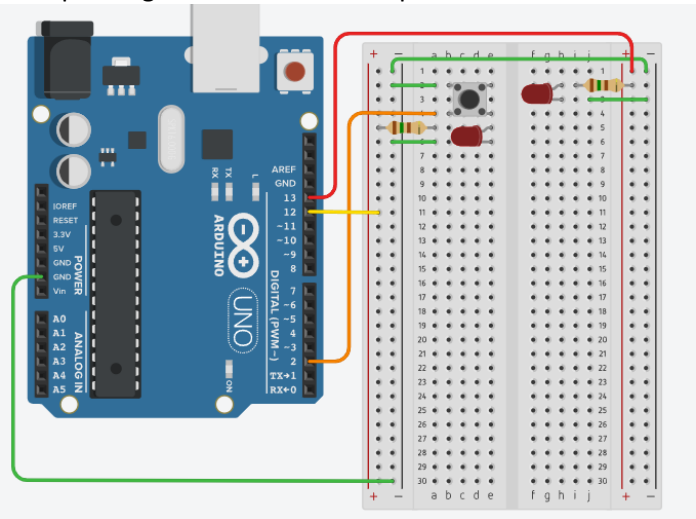
```
#define BP1 4
#define BP2 2
#define LED1 13
#define LED2 12

void setup() {
  pinMode(BP1, INPUT_PULLUP);
  pinMode(BP2, INPUT_PULLUP);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  boolean etat1 = !digitalRead(BP1);
  digitalWrite(LED1, etat1);
  boolean etat2 = !digitalRead(BP2);
  digitalWrite(LED2, etat2);
}
```

Manip 3 : Clignotement LED



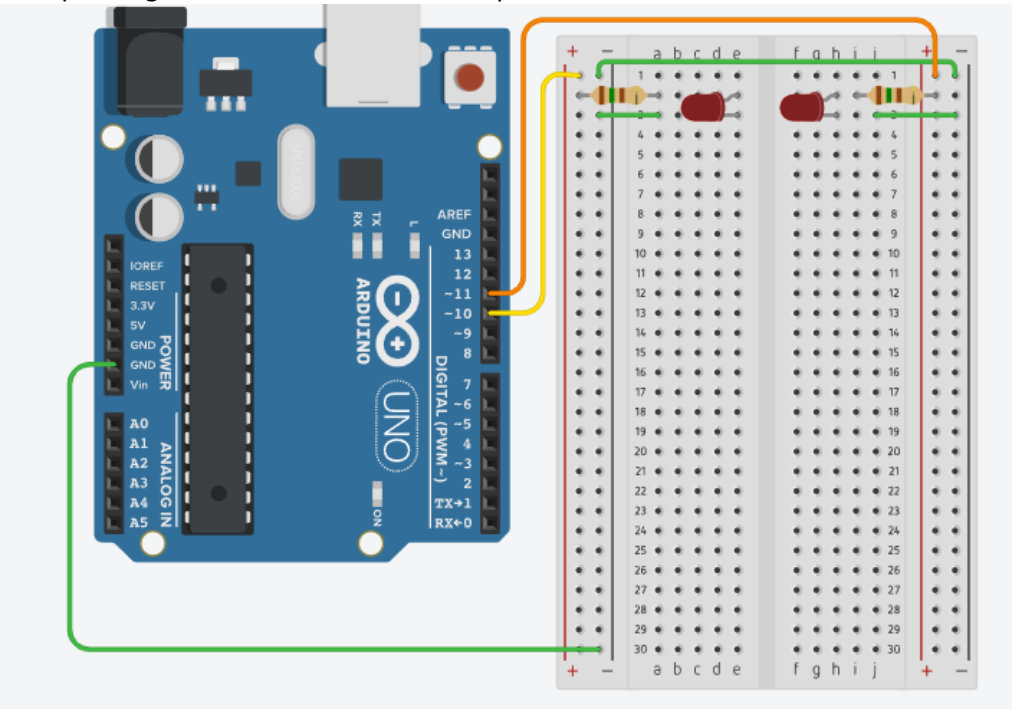

```
#define LED1 12
#define LED2 13

void setup() {
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  digitalWrite(LED1, !digitalRead(LED1));
  digitalWrite(LED2, !digitalRead(LED1));
  delay(1000);
}
```

Manip 4 :Clignotement d’une led plus une autre



```
#define BP 2
#define LED1 13
#define LED2 12

void setup(){
  pinMode(BP, INPUT_PULLUP);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  Serial.begin(9600);
}
unsigned long TICK = 1000;
unsigned long LAST_TICK = 0;

void loop(){
  unsigned long current_time = millis();

  if (current_time - LAST_TICK >= TICK) {
    digitalWrite(LED1, !digitalRead(LED1));
    LAST_TICK = current_time;
  }

  bool etat = !digitalRead(BP);
  digitalWrite(LED2, etat);
}
```

Manip 5 : clignotement de 2 led à des fréquences différences

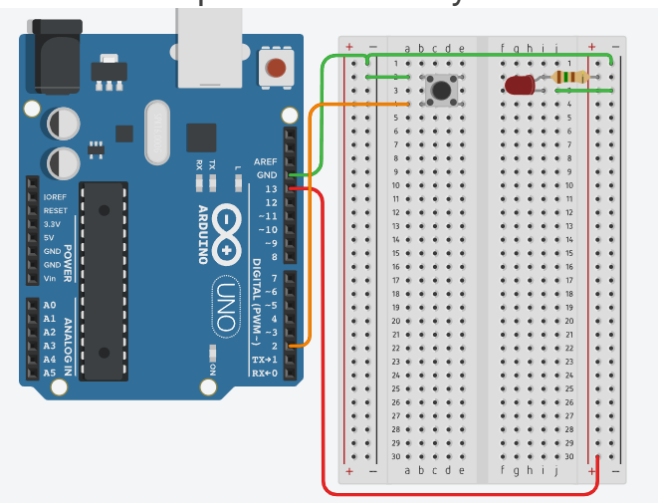


```
#define LED1 10
#define LED2 11

const float F1 = 1.0;
const float F2 = 0.8;

const float DUTY_CYCLE = 50;

const int VALUE = map(DUTY_CYCLE, 0, 100, 0, 255);

void setup() {
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  Serial.begin(9600);
}

unsigned long get_tick(float f) {
  return (1/f) * 1000;
}

const unsigned long TICK1 = get_tick(F1);
const unsigned long TICK2 = get_tick(F2);

unsigned long LAST_TICK1 = 0;
unsigned long LAST_TICK2 = 0;

bool ETAT1 = false;
bool ETAT2 = false;

void loop() {
```

```
  unsigned long current_time = millis();

  if (current_time - LAST_TICK1 >= TICK1) {
    if (!ETAT1) {
      analogWrite(LED1, VALUE);
    } else {
      digitalWrite(LED1, false);
    }
    ETAT1 = !ETAT1;
    LAST_TICK1 = current_time;
  }

  if (current_time - LAST_TICK2 >= TICK2) {
    if (!ETAT2) {
      analogWrite(LED2, VALUE);
    } else {
      digitalWrite(LED2, false);
    }
    ETAT2 = !ETAT2;
    LAST_TICK2 = current_time;
  }
}
```

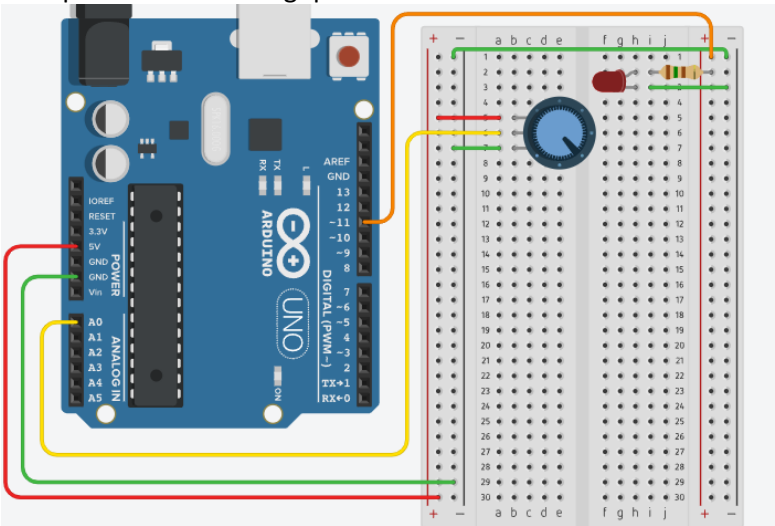
Manip 6 : Le télérupteur

Le télérupteur a besoin d'un condensateur pour éviter l'effet rebond. Celui-ci n'a pas été mis dans la manip 6 et 8. Pensez-y lors du montage.



```
#define BP 2
#define LED 13

void setup() {
  pinMode(BP, INPUT_PULLUP);
  pinMode(LED,OUTPUT);
  Serial.begin(9600);
}

bool WAS_PRESSED = false;

void loop() {
  bool pressed = !digitalRead(BP);

  if (!pressed && WAS_PRESSED){
    digitalWrite(LED, !digitalRead(LED));
  }

  WAS_PRESSED = pressed;
}
```

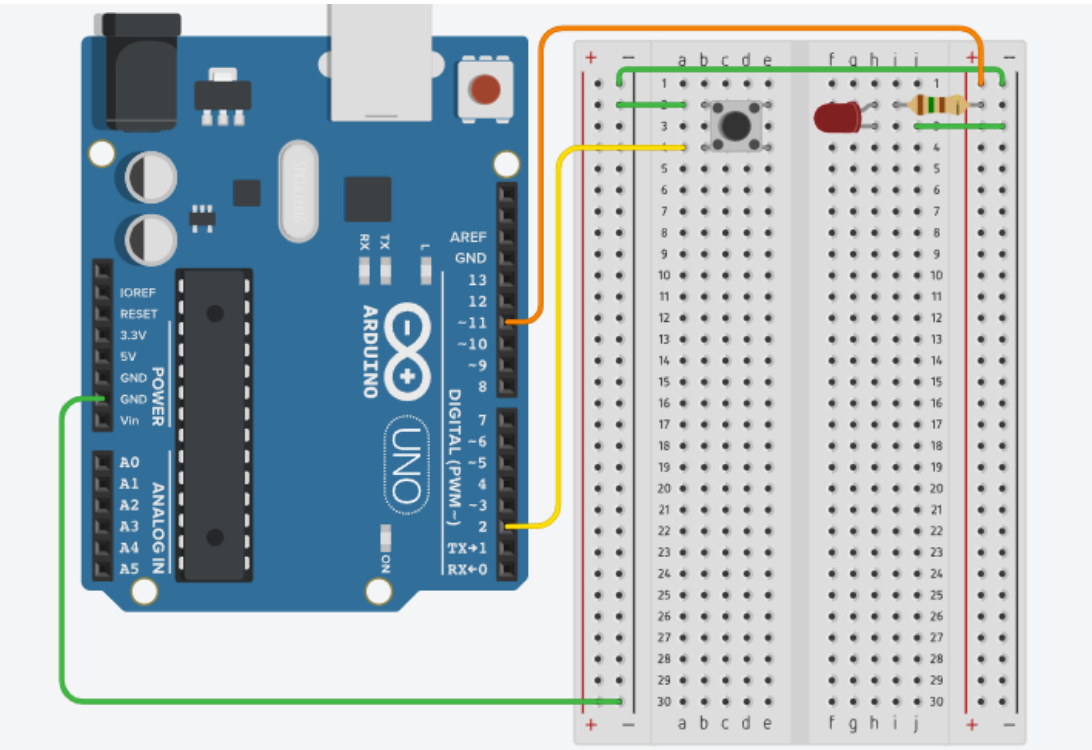
Manip 7 : Mesure analogique




```
#define PT 0
#define LED 11

void setup() {
  pinMode(PT, INPUT);
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  unsigned int value = map(analogRead(PT), 0, 1023, 0, 255);
  analogWrite(LED, value);
}
```

Manip 8 : Le dimmer

Le télérupteur a besoin d'un condensateur pour éviter l'effet rebond. Celui-ci n'a pas été mis dans la manip 6 et 8. Pensez-y lors du montage.




```
#define BP 2
#define LED 11

void setup() {
  pinMode(BP, INPUT_PULLUP);
  pinMode(LED,OUTPUT);
  Serial.begin(9600);
}

unsigned long TIME =0;

const unsigned long TICK = 2000;

unsigned long ELAPSED = 0;

unsigned int COUNTER = 0;
unsigned int VALUE = 250;

bool WAS_PRESSED = false;
bool NEXT = false;

void loop() {
```

```
  unsigned long current_time = millis();

  bool pressed = !digitalRead(BP);

  if (pressed) {
    ELAPSED = current_time - TIME;
  }

  if (digitalRead(LED) && ELAPSED >= TICK && ELAPSED >= COUNTER * TICK) {
    if (NEXT) {
      if (VALUE < 250) {VALUE += 25;}
    } else {
      if (VALUE > 25) {VALUE -= 25;}
    }
    analogWrite(LED, VALUE);
    COUNTER++;
  }


  if (!pressed) {
    if (WAS_PRESSED) {
      if (ELAPSED <= TICK) {
        digitalWrite(LED, !digitalRead(LED));
      } else {
        NEXT = !NEXT;
      }
    }
    COUNTER = 0;
    ELAPSED = 0;
    TIME = millis();
  }

  WAS_PRESSED = pressed;
}
```