Utilizzare e Creare una Libreria per il Display a 7 Segmenti

Obiettivo: Utilizzare e creare una libreria (file header e cpp) per un Display a Sette Segmenti.

Puoi scaricare i file di libreria cliccando nel seguente link: http://www.arduinofacile.it/wp-content/uploads/2021/03/SevenSegment.zip
I file scaricati devono essere inseriti all'interno della cartella di progetto insieme al file .ino

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Display a 7 Segmenti
- 8 Resistenze

Pre-requisiti:

Creare funzioni con Arduino … per un Display a 7 Segmenti

<u>Teoria</u>: la realizzazione di funzioni di libreria permette di facilitare l'operazione di **riutilizzo del codice** rendendo più veloce e più rapido lo sviluppo. Nel caso specifico la funzione di libreria implementata sarà costituta da un file header (.h) e da un file sorgente (.cpp).

Un file header è un file di testo che contiene i prototipi dei metodi (funzioni) definite nel relativo file sorgente. Nel caso in questione il file header contiene anche la dichiarazione della classe "SevenSegment" utilizzata per modellare il display a sette segmenti.
Tale classe sarà caratterizzata da 10 attributi:

- **int pinA**: il pin A del display a sette segmenti
- **int pinB**: il pin B del display a sette segmenti
- **int pinC**: il pin C del display a sette segmenti
- **int pinD**: il pin D del display a sette segmenti
- **int pinE**: il pin E del display a sette segmenti
- **int pinF**: il pin F del display a sette segmenti
- **int pinG**: il pin G del display a sette segmenti
- **int pinDP**: il pin DP del display a sette segmenti
- **bool isCommonAnode**: indica se il display è di tipo anodo comune oppure no

e da 11 metodi

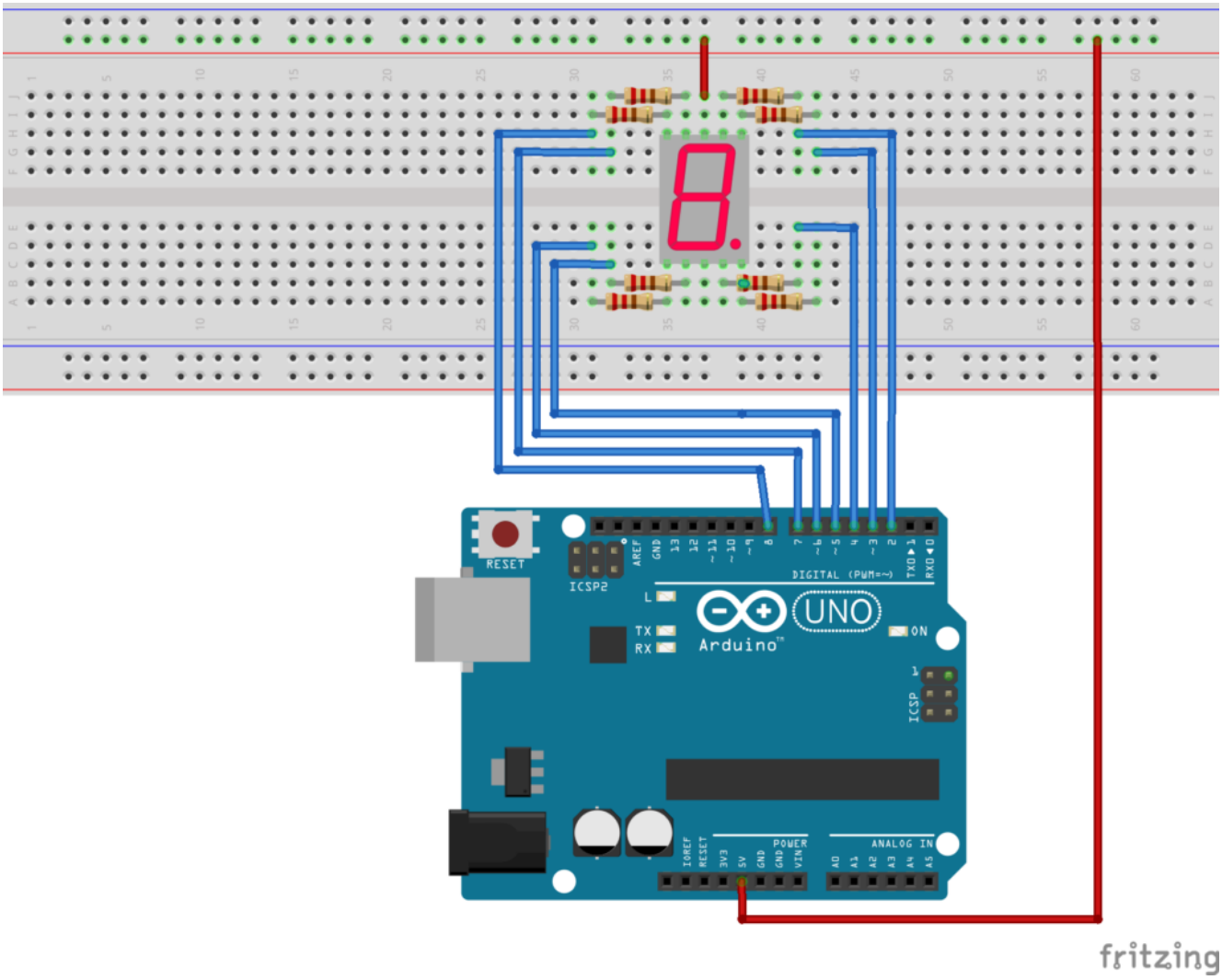
- **void Print0()**: metodo utilizzato per stampare il numero 0
- **void Print1()**: metodo utilizzato per stampare il numero 1
- **void Print2()**: metodo utilizzato per stampare il numero 2
- **void Print3()**: metodo utilizzato per stampare il numero

3

- **void Print4()**: metodo utilizzato per stampare il numero 4
- **void Print5()**: metodo utilizzato per stampare il numero 5
- **void Print6()**: metodo utilizzato per stampare il numero 6
- **void Print7()**: metodo utilizzato per stampare il numero 7
- **void Print8()**: metodo utilizzato per stampare il numero 8
- **void Print9()**: metodo utilizzato per stampare il numero 9
- **void CountDown()**: metodo utilizzato per eseguire il countdown.

Nel file sorgente viene invece riportata l’implementazione dei prototipi delle funzioni dichiarate nel file header.

Collegamento Circuitale:



Collegamento Circuitale

<u>Codice</u>:

Vengono in seguito riportate le tre porzioni di codice utilizzate per creare la funzione di libreria relativa al display a sette segmenti.

- File Header: contiene la definizione della classe con i propri attributi (i.e., pinA, pinB, etc) ed i prototipi dei relativi metodi.

- File Sorgente: contiene le implementazioni dei metodi riportati nel file header.

- File Arduino: Utilizzato per fornire un esempio di come utilizzare la libreria per la gestione del display a sette segmenti.

Se tutti i file sono correttamente posizionati sullo stesso livello all’interno della cartella di progetto, due nuove tab compariranno nell’ambiente di sviluppo utilizzato per programmare Arduino. Attraverso queste tab sarà possibile visionare e modificare il file sorgente (.cpp) ed il file header (.h)

IDE con l’utilizzo della libreria SevenSegment.h

Carica di un Condensatore

Obiettivo : Analizzare la carica di un condensatore mediante

una lettura analogica.

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Condensatore (1000 microFarad)
- 1 Resistenza (100 Ohm) per carica condensatore
- 1 Pulsante
- 1 Resistenza (1 kOhm) per resistenza pulsante

Teoria: Il condensatore è un dispositivo elettronico che ha la capacità di immagazzinare carica elettrostatica. Tale dispositivo modifica il suo comportamento in funzione del segnale di alimentazione utilizzato nell'applicazione in questione (e.g., segnale continuo, segnale alternato, etc …).

In regime di tensione costante (corrente continua), il condensatore si carica inizialmente (**regime transitorio**) fino a raggiunge una situazione di equilibrio dove la carica sulle armature corrisponde esattamente alla tensione applicata mediante il generatore (**regime stazionario**). Una volta carico, il condensatore si comporta come un circuito aperto ovvero interrompe ogni flusso di corrente all'interno del circuito.
Al cessare dell'eccitazione sul circuito l'energia elettrica accumulata nel condensatore torna a scaricarsi sotto forma di corrente elettrica rilasciata nel circuito.

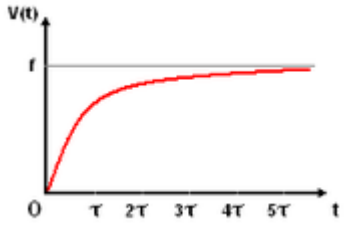
Il tempo di carica e di scarica dipende alla costante di tempo *Tau* definita come:

Tau = R*C

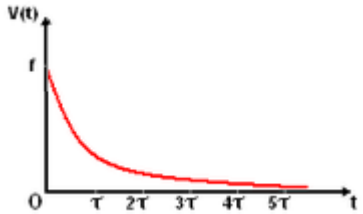
Nelle seguenti figure sono riportati i grafici relativi alla carica e alla scarica del condensatore. Nell'immagine di sinistra il condensatore, inizialmente scarico, si carica fino a raggiungere il livello di tensione finale *f*. Differentemente, nell'immagine di destra il condensatore, inizialmente carico e con una tensione *f* ai suoi capi, si scarica fino a raggiungere un livello di tensione finale pari a zero volt.

In questa esperienza si propone l'utilizzo del monitor seriale per visualizzare i tempi di carica del condensatore. E' importante considerare che i dati riportati sul monitor seriale (tempo e tensione) possono essere facilmente copiati in un foglio excel per ottenere una migliore rappresentazione grafica dell'evoluzione della tensione nel tempo.

L'utilizzo di un pulsante permette di avviare la carica del condensatore mediante l'utilizzo di una istruzione di digitalWrite.



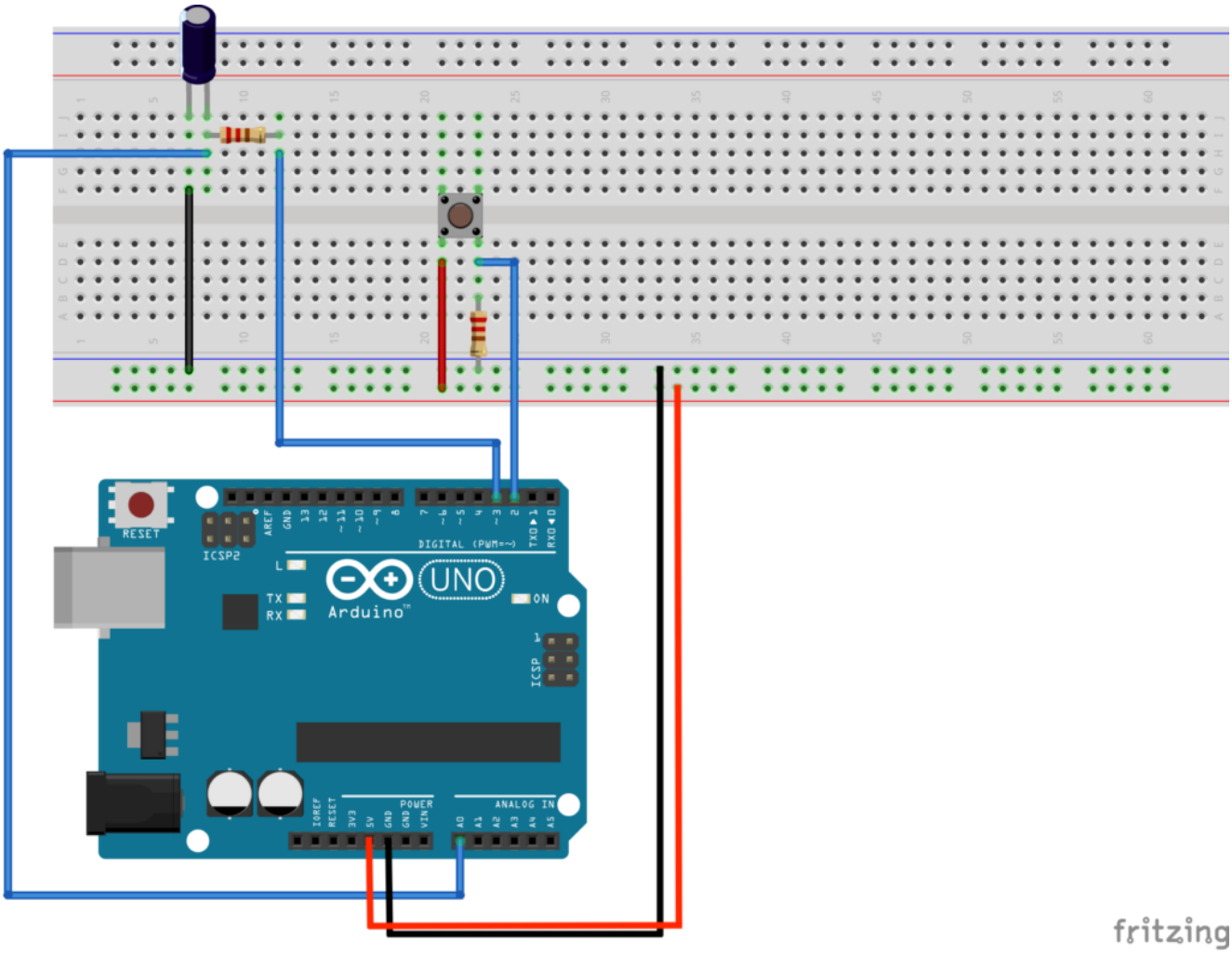
Carica di un condensatore



Scarica di un condensatore

Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per analizzare il tempo di carica di un condensatore. Nel dettaglio il tempo di carica viene monitorato utilizzando la lettura analogica del controllore Arduino.



Collegamento Circuitale

Codice:

Personalizzazioni:

E’ possibile modificare il codice e l’hardware per simulare anche la scarica del condensatore.

Creare funzioni con Arduino … per un Display a 7 Segmenti

Obiettivo: Creare una serie di funzioni con Arduino per utilizzare un display a 7 segmenti .

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Display a 7 Segmenti
- 7 Resistenza (100 Ohm)

Pre-Requisiti:

1..2..3.. Il Display a 7 Segmenti

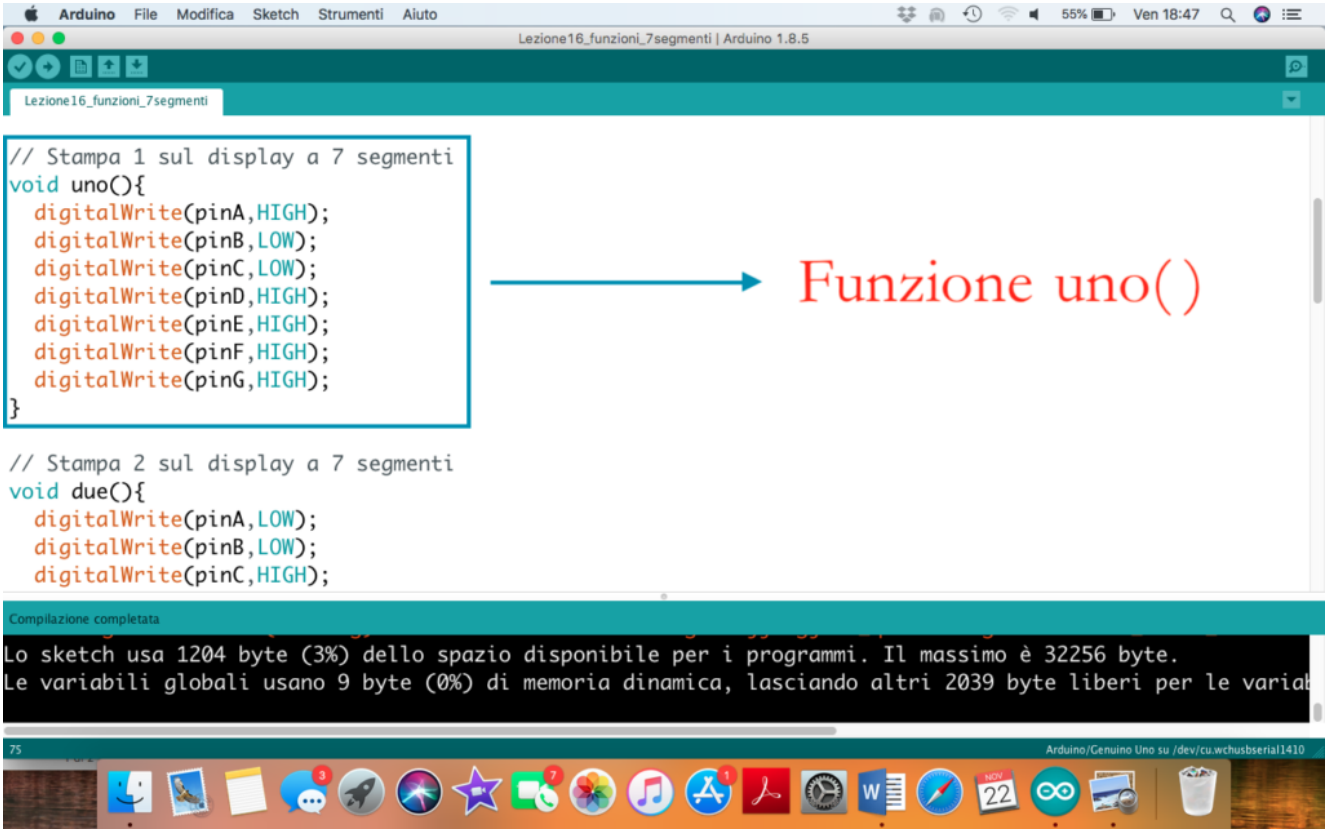
Teoria: La possibilità di riutilizzare il codice precedentemente scritto (**code re-use**) rappresenta una delle pratiche più comuni nella programmazione ovvero richiamare/invocare parti di codice precedentemente già sviluppate, ogni qualvolta risulta necessario, senza doverle riscrivere daccapo.

Nello specifico, questa possibilità si concretizza attraverso la scrittura di funzioni che possono essere richiamate/invocate all'interno dello stesso programma.

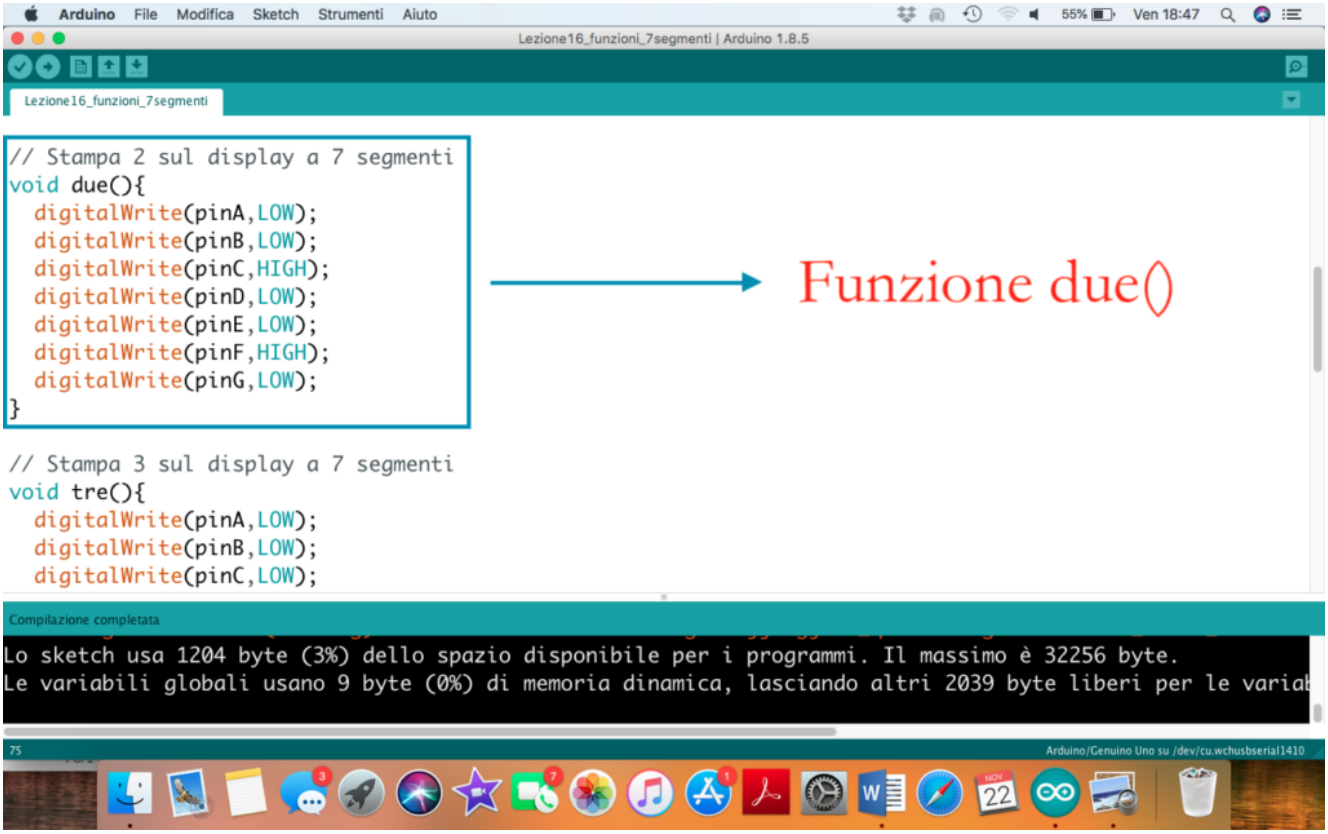
Queste funzioni possono svolgere diverse operazioni, che permettono di manipolare degli input dati ed eventualmente restituire degli output desiderati.

Nel caso in questione le tre funzioni implementate (denominate, uno, due e tre) non prevedono né il passaggio specifico di parametri di input né la restituzione di un output dato; queste tre funzioni svolgono solamente una serie di azioni sequenziali volte ad accendere alcuni elementi di un display a 7 segmenti. E' importante considerare che le tre funzioni sono tutte definite prima del loro utilizzo (ovvero prima delle due funzioni principali di setup e loop).

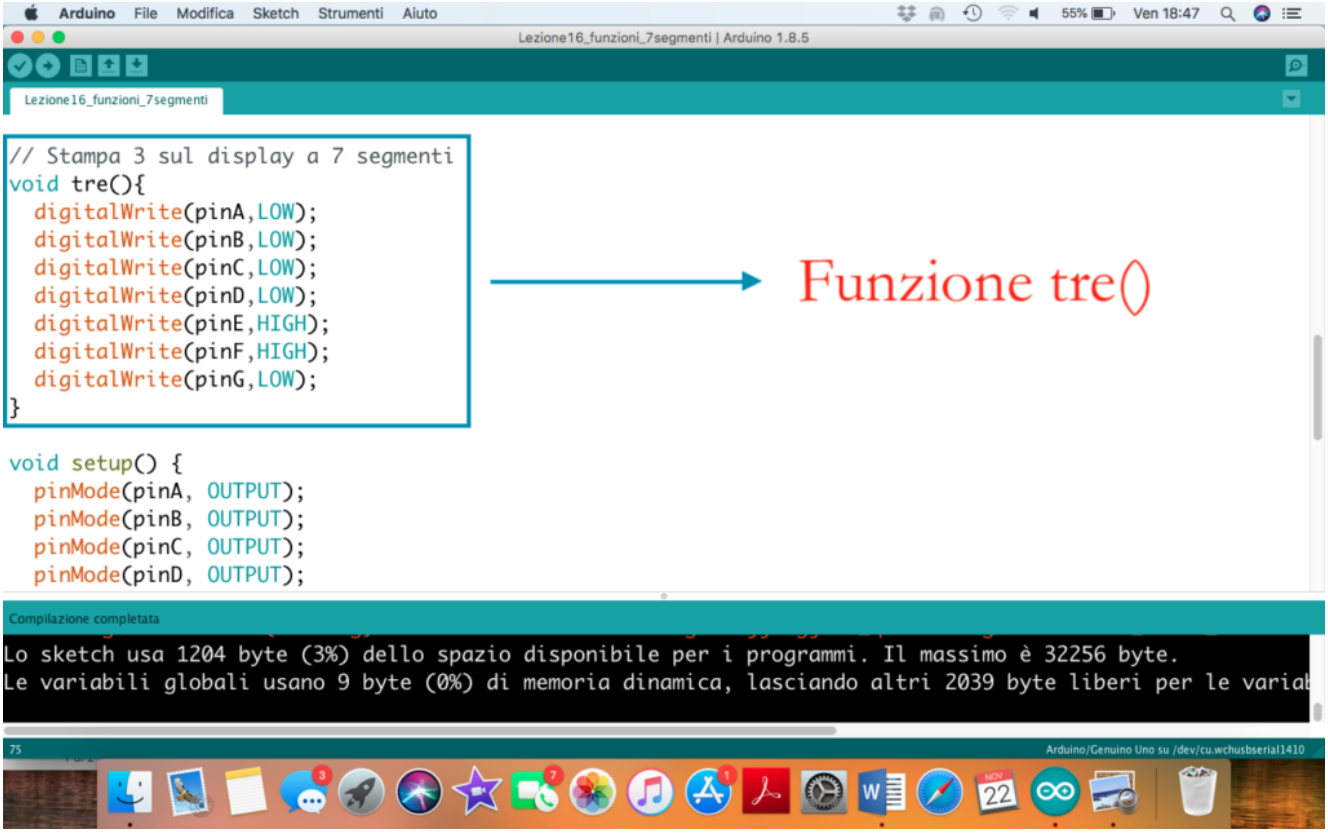
- **Funzione Uno()**: Stampa il carattere 1 sul display a 7 segmenti



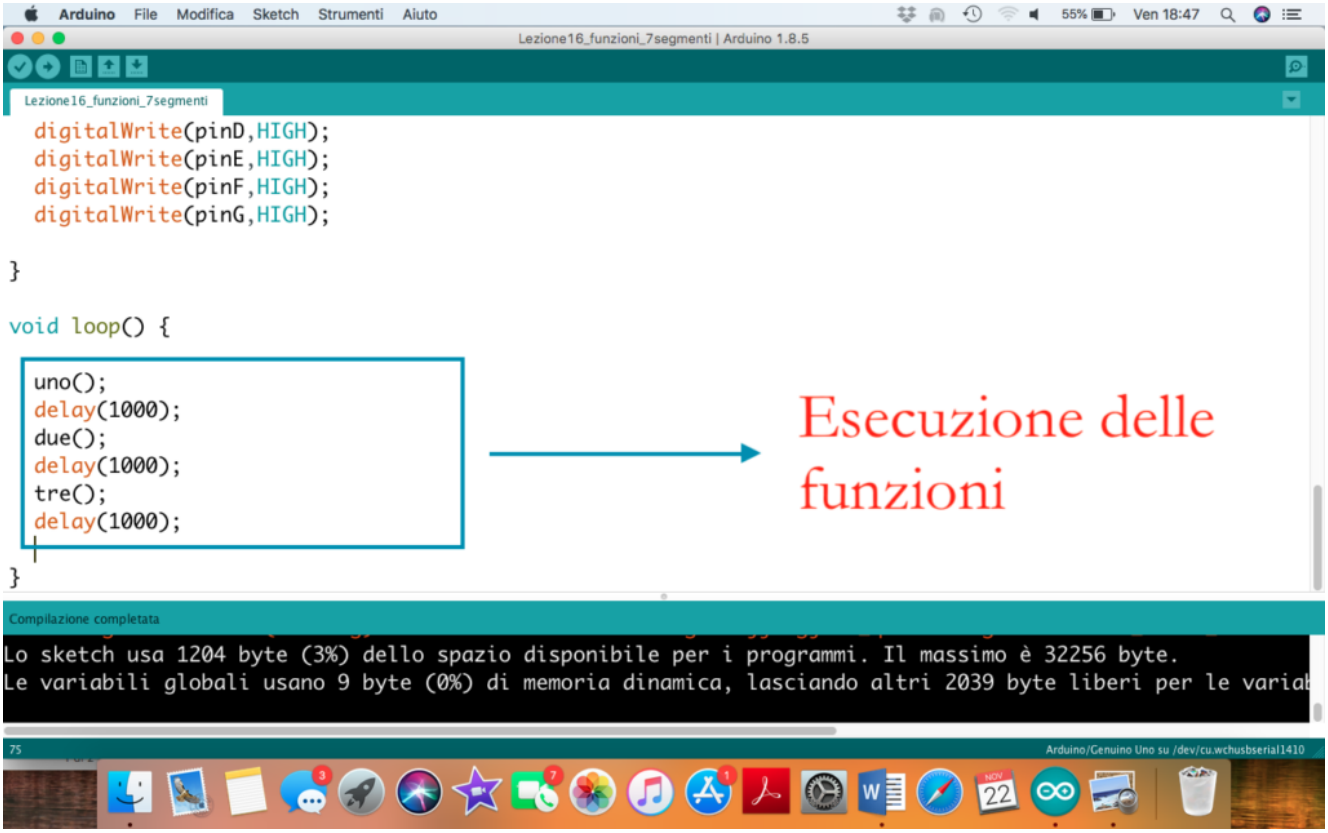
- **Funzione Due()**: Stampa il carattere 2 sul display a 7 segmenti



- **Funzione Tre()**: Stampa il carattere 3 sul display a 7 segmenti

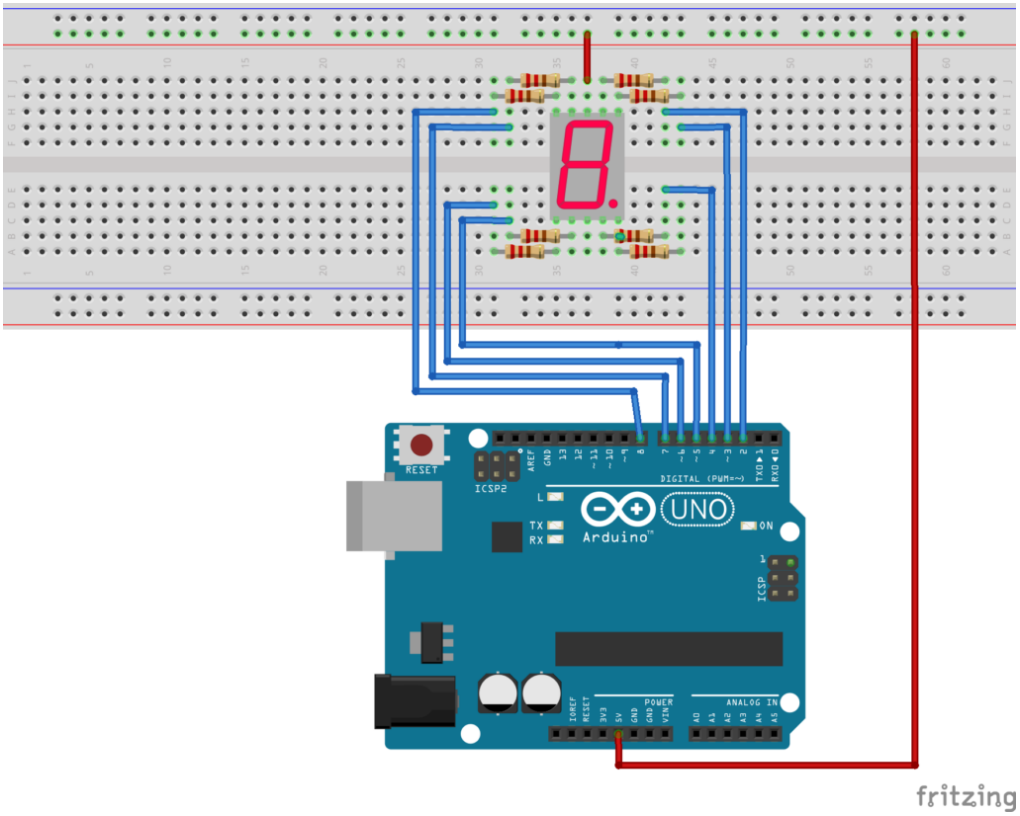


- **Invocazione delle Funzioni**: Le funzioni possono essere invocate semplicemente richiamandole nel punto in cui devono essere eseguite. Nel caso in questione le funzioni sono richiamate all’interno del blocco loop().



<u>Collegamento Circuitale</u>:

Viene in seguito riportato lo schema elettrico utilizzato per comandare un display a 7 segmenti mediante funzioni. Nel dettaglio il display impiegato è della modalità anodo comune.



Collegamento Circuitale

<u>Codice</u>:

Considerando l’impiego di un display ad anodo comune i singoli LED possono essere accesi mediante l’istruzione <u>digitalWrite(pinLed,LOW)</u>;

Tre differenti funzioni sono state create per gestire il display. La funzioni sono, rispettivamente denominate, uno(), due() e tre(). Queste funzioni devono essere definite prima dei blocchi setup e loop, non prevedono l’impiego di parametri in ingresso né la restituzione di output in uscita (funzioni di tipo void).

Personalizzazioni:

E’ possibile modificare il software aggiungendo altre funzioni per la rappresentazione dei vari numeri sul display a 7 segmenti.

1..2..3.. Il Display a 7 Segmenti

Obiettivo: Utilizzo di un display a 7 segmenti .

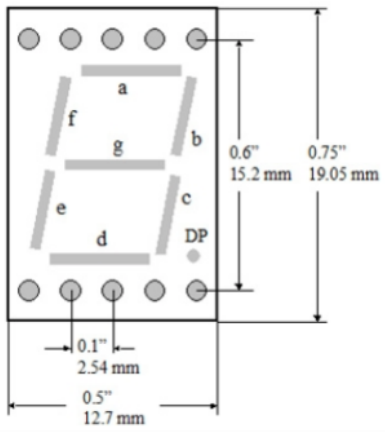
Componenti elettronici:

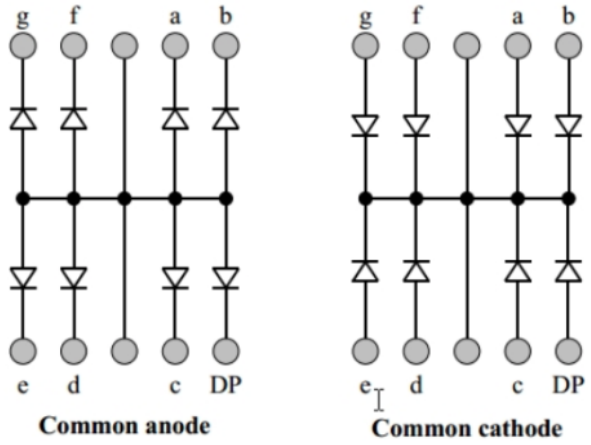
- Arduino UNO
- Breadboard
- 1 Display a 7 Segmenti
- 7 Resistenza (100 Ohm)

Teoria: Il display a 7 segmenti è un dispositivo elettronico in grado di visualizzare, attraverso l’accensione di combinazioni di sette differenti segmenti luminosi, le 10 cifre numeriche, alcune lettere alfabetiche e simboli grafici.

I display a 7 segmenti sono dotati di 10 differenti pin e sono così classificati:

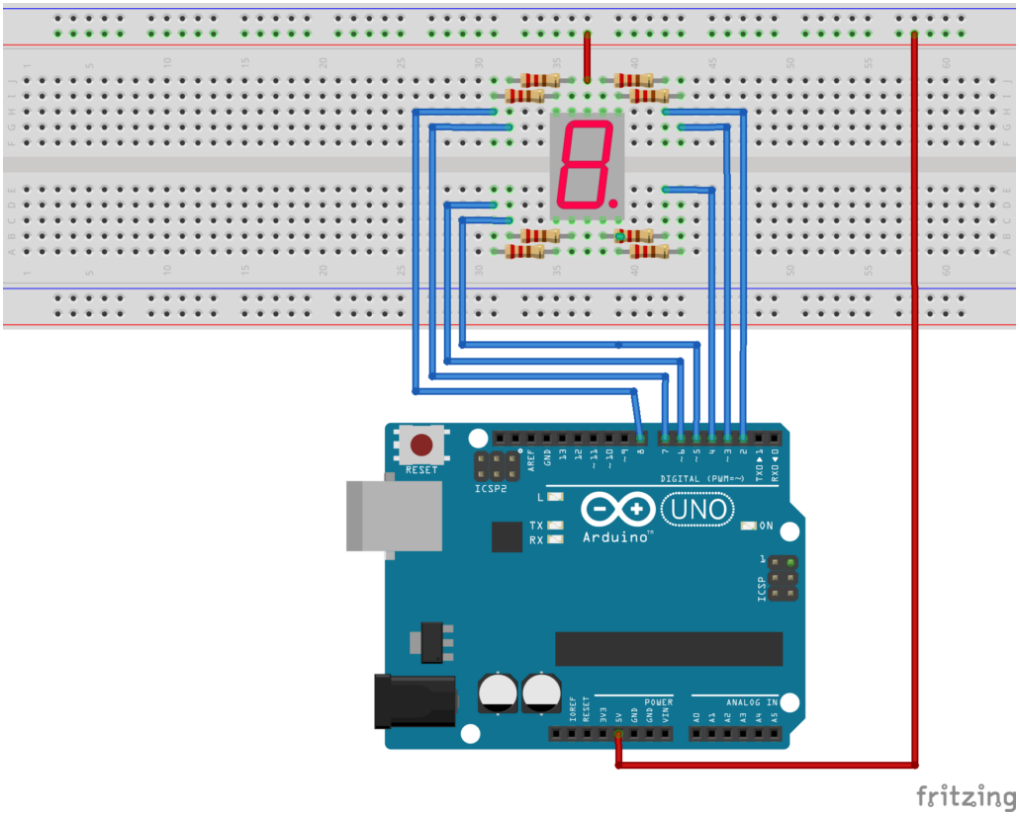
- **Display 7 segmenti ad Catodo Comune:** il pin centrale del LED deve essere collegato a massa (GND) mentre gli altri 8 differenti PIN sono utilizzati per comandare i 7 segmenti ed il punto DP. Nel caso in questione i singoli LED sono accesi attraverso l'istruzione Arduino digitalWrite(pinLed, HIGH);
- **Display 7 segmenti ad Anodo Comune:** il pin centrale del LED deve essere collegato a 5V (VCC) mentre gli altri 8 differenti PIN sono utilizzati per comandare i 7 segmenti ed il punto DP. Nel caso in questione i singoli LED sono accesi attraverso l'istruzione Arduino digitalWrite(pinLed, LOW);





Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per comandare un display a 7 segmenti. Nel dettaglio il display impiegato è della modalità anodo comune.



Collegamento Circuitale

<u>Codice</u>:

Considerando l’impiego di un display ad anodo comune i singoli LED possono essere accesi mediante l’istruzione <u>digitalWrite(pinLed,LOW)</u>;

Personalizzazioni:

E’ possibile modificare il codice aggiungendo la possibilità di visualizzare altri numeri e non solo 1, 2 e 3.