Università di Bologna

Corso di Didattica dell'Informatica

Dipartimento di Informatica

Approfondimento sulla programmazione a blocchi tramite Microcontrollori.

Documento per l'insegnante

Alessandro Benetton [0001038887]

4 luglio 2023

Referente: alessandro.benetton@studio.unibo.it

Indice

1 Introduzione

1.1 Obiettivi

L'obiettivo di questa unità didattica è quello di consolidare le conoscenze acquisite nel corso di programmazione a blocchi applicandole ad un nuovo ambito, quello della programmazione di microcontrollori.
Le logiche di programmazione sono quelle viste in classe, il microcontrollore aggiunge la possibilità di interagire con il mondo esterno attraverso sensori e attuatori, consentendo esercizi e progetti più interattivi e vicini alla vita di tutti i giorni.

Data la natura di questa unità didattica, non sono previste metodologie di valutazione formativa a fine unità come compiti e quiz. Durante le attività saranno forniti degli spunti per consentire una discussione sul codice e la raccolta di osservazioni informali.

1.2 Materiali

Gli esercizi proposti nei capitoli seguenti sono stati pensati per essere svolti da studenti divisi in gruppi di 2-3 persone.

Ogni gruppo necessita di un computer con connessione ad internet. Sarebbe ottimale avere a disposizione un microbit per ogni gruppo, ma è possibile lavorare con il dispositivo simulato presente nell'editor online e mettere a disposizione alcuni dispositivi per test da passare tra gli studenti.

1.3 Micro:Bit

MicroBit è un microcontrollore sviluppato da BBC, un'azienda britannica, per insegnare le basi della programmazione e dell'informatica.

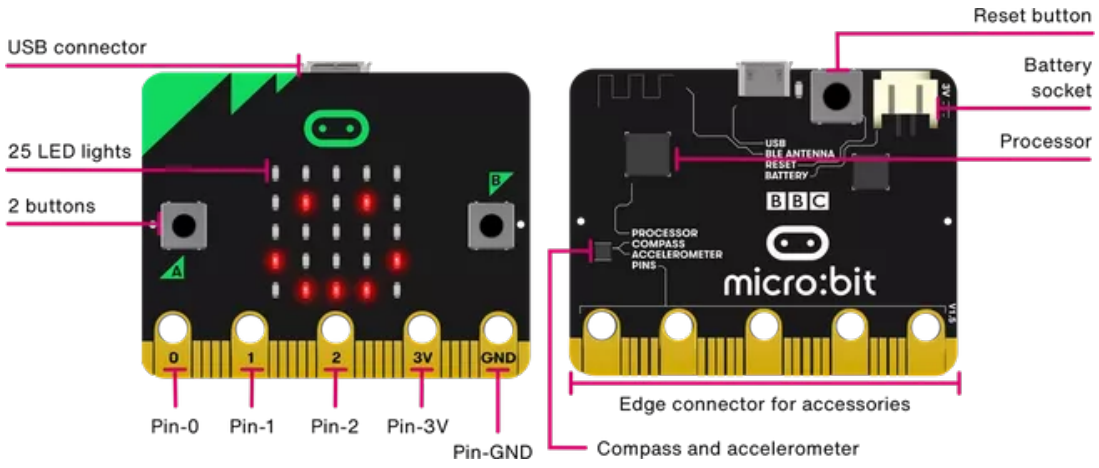

Figura 1: Sensori ed attuatori MicroBit versione 1 [2]

Il dispositivo è dotato di un display (una griglia di 5x5 LED), due pulsanti e molti sensori[1] tra cui:

- accelerometro
- bussola
- sensore di luminosità

1.4 Make:Code

MakeCode è un editor online per la programmazione di microcontrollori sviluppato da Microsoft e BBC per MicroBit.
L'editor è gratuito e non richiede alcuna installazione, è sufficiente collegarsi al sito https://makecode.microbit.org/ per iniziare a programmare.

MakeCode consente di programmare utilizzando programmazione a blocchi oppure utilizzando codice Javascript o Python. In questa unità didattica verrà utilizzata la programmazione a blocchi ma è possibile in qualunque momento converire il codice nelle altre opzioni semplicemente cambiando schermata dell'editor.

[1] Una lista completa di sensori e attuatori con esempi di utilizzo è disponibile al seguente indirizzo: https://microbit.org/get-started/user-guide/overview

2 Strumenti Utili

2.1 Risorse MicroBit Foundation

MicroBit Foundation offre molte risorse utili per l'insegnamento della programmazione con MicroBit.
Tutte le risorse sono disponibili al seguente indirizzo: https://microbit.org/teach.

2.2 Make:Code

Come detto nelle sezioni precedenti, Make:Code è un editor online che permette di programmare MicroBit in modo semplice e intuitivo.

2.2.1 Salvataggio e caricamento

Il codice creato con MakeCode può essere salvato in locale (tramite l'icona di salvataggio in basso o il pulsante *Scarica*), questo genererà un file *.hex*.
In qualunque momento si può trascinare un file *.hex* nell'editor per caricare il codice salvato oppure si può usare il pulsante *Importa* nella home di MakeCode.

Caricamento su MicroBit Per caricare il codice su MicroBit è necessario collegare la scheda al computer tramite cavo USB e trascinare il file *.hex* (vedere paragrafo sopra) nel dispositivo *MICROBIT* che apparirà sul computer.

2.2.2 MicroBit Classroom

MicroBit Classroom (https://classroom.microbit.org) è un servizio online che permette di creare classi virtuali (in ambiente Make:Code) con gli studenti e monitorare in tempo reale i loro progressi.

Questo strumento è molto utile poichè consente di condividere codice con gli studenti mentre si spiega e vedere tutti i loro editor mentre svolgono esercizi (è inoltre possibile scaricare il codice degli studenti per una successiva revisione).

La piattaforma non richiede alcuna registrazione e può essere utilizzata da chiunque.

Salvataggio e caricamento in MicroBit Classroom Le procedure di salvataggio e caricamento sono differenti per MicroBit Classroom.
Per salvare la sessione (Codice del docente e quello di tutti gli studenti) è necessario andare nella sezione *Save Classroom* e scaricare il file *HTML*.

Per caricare una sessione basta aprire il file *HTML* scaricato in precedenza e cliccare su *Resume classroom session*.

2.3 Suggerimenti per lo svolgimento delle attività

Le attività proposte sono pensate per essere svolte in gruppi di 2-3 persone.
Si suggerisce di formare i gruppi in modo che gli studenti con più difficoltà possano essere aiutati dai compagni più esperti, così da favorire la progressione uniforme di tutti i gruppi.
È consigliato non fornire soluzioni già pronte agli studenti, ma di aiutarli a raggiungere la soluzione da soli, in modo che possano imparare a risolvere i problemi in autonomia.

3 Passaggi Fondamentali

Questa sezione fungerà da introduzione all'editor Make:Code e ad un argomento fondamentale nella programmazione di microcontrollori: *Input/Output*.

3.1 Make:Code

Make:Code è un editor online per programmare MicroBit. Per accedervi basta andare sul sito https://makecode.microbit.org/ e cliccare su *Nuovo Progetto*.

Il linguaggio a blocchi è simile a scratch, ma non uguale. Scratch nasce con l'obiettivo di insegnare a programmare tramite teatralizzazione e storytelling mentre Make:Code nasce con l'obiettivo di insegnare a programmare usando un microcontrollore e gamification.

Questa differenza è evidente nel modo in cui l'editor invita a strutturare il codice, scratch invita a creare molti codici "in parallelo" utilizzando un punto di partenza per ogni sequenza di comandi, makecode, invece, invita una struttura del codice sequenziale.
La struttura del codice in makecode è molto simile a quella di un programma per microcontrollori scritto in un linguaggio di programmazione testuale, ovvero composta da due blocchi di codice, il "setup" e il "loop".
Il primo blocco racchiude il codice che viene eseguito una sola volta all'avvio del microcontrollore, mentre il secondo blocco racchiude il codice che viene eseguito in loop, ovvero ripetutamente, fino a quando il microcontrollore è acceso.

È possibile programmare Micro:Bit utilizzando scratchX e l'estensione per Micro:Bit, ma non è consigliato in quanto non è possibile utilizzare tutte le funzionalità di Make:Code e non si ha a disposizione il simulatore.

3.1.1 [Attività] Discussione sulle differenze tra Scratch e MakeCode

Questo potrebbe essere un ottimo momento per raccogliere delle osservazioni informali osservando come il gruppo si approccia ad un nuovo editor, sia in fase di "gioco" che in quella di confronto.

L'obiettivo di questa attività è quello di far discutere gli studenti sulle differenze tra Scratch e MakeCode.
Per fare ciò si consiglia di dividere gli studenti in gruppi da 2-3 persone e dare 5 minuti di tempo per giocare con l'editor e capire come funziona.
In seguito dare 10 minuti agli studenti e chiedere loro di identificare i punti in comune e le differenze tra Scratch e MakeCode (volendo si può fornire un esempio prima di iniziare).
Scaduto il tempo chiedere alla classe di esporre le proprie idee e discuterle.

Esempi

- Scratch ha una finestra in cui si possono vedere ed animare i personaggi, MakeCode ha una finestra con un simulatore del microcontrollore.
- La categoria "Sensori" di scratch corrisponde alla categoria "Input" di MakeCode.
- Il comando "Dire" di scratch corrisponde al comando "Mostra Stringa/Numero" di MakeCode.

3.2 Input e Output

MicroBit presenta molte periferiche di input e output, come visto nelle sezioni precedenti.
In questa sezione verranno introdotti i concetti di input e output e verranno mostrati alcuni esempi di come utilizzarli. Le seguenti attività sono pensate per consentire agli studenti di familiarizzare con l'editor di makecode, con il simulatore e con i concetti di input e output relativi a microbit.

3.2.1 Attività 1 - Display

MicroBit ha un display a matrice di led che può essere utilizzato per mostrare testo, numeri e immagini.
L'obiettivo di questa attività è quello di scrivere un programma che scriva sul display il contenuto di una variabile.
Il valore verrà impostato nel blocco di setup e sarà costante per tutto il programma.

Soluzione Sono possibili più soluzioni, l'importante è che rispettino i seguenti criteri:

- Il valore della variabile deve essere impostato nel blocco di setup.
- Deve essere usato il comando corretto tra "Mostra Stringa" e "Mostra Numero" per mostrare il valore in base al tipo di dato salvato nella variabile.

Discussione *Il comando "Mostra Stringa/Numero" può essere inserito sia nel blocco di setup che in quello di loop, ci sono differenze? Cambia qualcosa se la stringa è composta da un solo carattere o da più caratteri? (Lo stesso discorso vale per il numero)*

Si consiglia di chiedere agli studenti cosa si aspettano che succeda e di far scrivere il codice per verificare le loro ipotesi.

Se il comando "Mostra Stringa/Numero" si trova nel blocco di setup, il display mostrerà solo una volta il contenuto della variabile. Se invece si trova nel blocco di loop, il display mostrerà il contenuto della variabile ripetutamente fino a quando il microcontrollore è acceso.
Il display di MicroBit può mostrare un carattere alla volta, per questo motivo il comando "Mostra Stringa/Numero" agirà in maniera diversa se la stringa è composta da un solo carattere o da più

caratteri.
Nel primo caso verrà visualizzato il carattere e questo resterà nel display fino a quando un nuovo comando non modificherà il display (In questo caso mettere il comando nel blocco di setup o loop è indifferente).
Nel secondo caso il display visualizzerà la sequenza di caratteri a scorrimento e, una volta terminata, il display tornerà vuoto. (In questo caso vi è una differenza tra mettere il comando nel blocco di setup o in quello di loop)

3.2.2 Attività 2 - Contatore

Esistono due modi di interagire con un pulsante, il primo è quello di controllare se il pulsante è premuto in un dato momento (se in questo momento il pulsante A è premuto, allora ..., altrimenti ...), il secondo consiste nell'utilizzare gli eventi (Quando A viene premuto esegui ...).
La principale differenza è che il primo metodo funziona in maniera sequenziale rispetto al resto del codice (se A viene premuto ma il codice non è arrivato al controllo allora la pressione viene ignorata), il secondo agisce in parallelo rispetto al resto del codice (Il codice viene eseguito immediatamente ogni volta che A viene premuto).

L'obiettivo di questa attività è quello di estendere il programma precedente al fine di aumentare o diminuire il valore di una variabile numerica mostrata a schermo in base al pulsante premuto.

Specifiche Il programma deve

- Mostrare il valore di una variabile numerica a schermo.
- Decrementare il valore della variabile di 1 se il pulsante A viene premuto.
- Incrementare il valore della variabile di 1 se il pulsante B viene premuto.
- Aggiornare costantemente il valore mostrato a schermo per riflettere quello della variabile.
- (OPZIONALE) Fare in modo che il valore non esca dal range $0 \leq x \leq 9$

Soluzione La soluzione dovrebbe estendere il codice scritto precedentemente (utilizzando "Mostra numero" nel blocco loop) aggiungendo due punti di partenza per la rilevazione degli eventi "Pulsante A premuto" e "Pulsante B premuto".
Si può trovare una possibile soluzione all'interno della cartella *sorgenti* con il nome *microbit-1-input_output.hex*[2]

È possibile una seconda soluzione che prevede il comando "Monstra numero" a seguito di ogni incremento / decremento, consentendo così di non avere il blocco loop.

[2] Per vedere come caricare un file in MakeCode consultare sottosottosezione 2.2.1

3.3 Time

Nell'ambito della programmazione di microcontrollori è raro avere accesso al tempo reale (orario dell'orologio), generalmente si ha accesso ad un timer interno al microcontrollore che viene inizializzato all'avvio e può essere utilizzato per misurare intervalli di tempo.

3.3.1 Attività 3 - Cronometro

L'obiettivo di questa attività è quello di scrivere un programma che misuri il tempo trascorso tra due pressioni di un pulsante e lo mostri a schermo.

Specifiche Il programma deve:

- Mostrare un simbolo sul display per indicare che il cronometro è pronto.
- Alla prima pressione del pulsante A iniziare a misurare il tempo trascorso e cambiare il simbolo sullo schermo.
- Alla successiva pressione del pulsante A deve terminare la misurazione del tempo e scrivere il risultato sullo schermo

Soluzione Esistono diverse soluzioni a questo problema, nella cartella *sorgenti* sono presenti due delle soluzioni possibili, *microbit-2-time-a.hex* e *microbit-2-time-b.hex*.
La prima soluzione utilizza un comando "pausa" nel blocco di loop per rallentare il programma ed ogni secondo incrementa di 1 la variabile contatore.
La seconda soluzione salva in una variabile i millisecondi passati dall'avvio del programma alla prima pressione di A, poi salva quelli passati dall'avvio alla seconda pressione di A, infine calcola la differenza tra i due valori e la mostra a schermo.

La seconda soluzione è preferibile in quanto non rallenta il programma e non richiede di utilizzare un comando "pausa" ma è meno intuitiva dal punto di vista dello studente.

Discussione (Dopo aver mostrato entrambe le soluzioni) *Quali sono i vantaggi e gli svantaggi delle due soluzioni? Qual'è la soluzione più precisa?*

La prima soluzione è più semplice da capire e da implementare dato che usa meno variabili ma è meno precisa. Spesso si cerca di evitare il comando "pausa" nella programmazione di microcontrollori poichè questo interrompe l'esecuzione dell'intero programma (non è detto che sia sbagliato usarlo, dipende molto da cosa si sta creando).
Inoltre la prima soluzione è meno precisa in quanto l'incremento viene fatto ogni secondo, questo significa che se il tempo trascorso è 1.9 secondi verrà mostrato 1 secondo, se il tempo trascorso è 2.1 secondi verrà mostrato 2 secondi.

La seconda soluzione è più precisa in quanto utilizza i millisecondi per misurare il tempo trascorso, inoltre non blocca l'esecuzione del programma. Lo svantaggio principale di questa soluzione è l'introduzione di due variabili aggiuntive e la conseguente complicazione del codice.

4 Reaction Game

L'obiettivo del progetto finale è quello di realizzare un gioco in cui si valutano i riflessi dei giocatori.

Il gioco prevederà che due giocatori impugnino il MicroBit rispettivamente sul tasto A e B. Il microcontrollore fornirà un sengale visivo e che indicherà ai giocatori di premere il tasto, il primo giocatore a premerlo vince.

Il gioco è semplice, ma per implementarlo è necessaria una buona comprensione di come funzionino variabili e cicli.

Per favorire la comprensione del progetto, si consiglia di sviluppare il lavoro in maniera progressiva, facendo realizzare prima una versione a giocatore singolo e poi una a due giocatori (come illustrato nelle sezioni 1.1 e 1.2).

4.1 Prima implementazione

L'obiettivo è realizzare un misuratore di tempi di reazione che misuri il tempo che passa tra l'emissione di un segnale dal microcontrollore e la pressione del tasto A da parte del giocatore.

4.1.1 [Attività] Identificazione delle 3 fasi principali del programma

Si consiglia di impostare questa attività come discussione con tutta la classe, in modo da favorire la comprensione del progetto e la partecipazione di tutti gli studenti.
Chiedere agli studenti di identificare le tre fasi principali del programma e descrivere cosa deve fare ognuna di queste.
Le tre fasi sono:

1. **Attesa / Preparazione**
    - Mostrare un segnale visivo di attesa (ad esempio una croce);
    - Attendere un intervallo randomico di tempo;
2. **Misurazione**
    - Mostrare un segnale visivo di conferma (ad esempio una spunta o un cerchio);
    - Iniziare a misurare il tempo;
    - Attendere la pressione del tasto A;
3. **Visualizzazione**
    - Calcolare il tempo impiegato dal giocatore;
    - Mostrarlo sul display;

4.1.2 Attività 4 - Reaction Game 1

La prima implementazione deve funzionare come segue:

1. Il giocatore impugna il MicroBit sul tasto A;
2. Il dislay del MicroBit mostra un segnale visivo di attesa (ad esempio una croce);
3. Dopo un numero variabile di secondi (in un intervallo scelto dallo studente) il display mostra un segnale visivo di conferma (ad esempio una spunta o un cerchio) ed avvia un timer;
4. Nel momento in cui il giocatore vede il segnale visivo di conferma, deve premere il tasto A; appena viene rilevata la pressione del tasto, il microcontrollore deve fermare il timer e mostrare il tempo impiegato dal giocatore in millisecondi.
5. EXTRA: Impedire che il giocatore possa barare premendo il tasto prima che venga mostrato il segnale visivo di conferma.

In questa prima implementazione, il giocatore gioca da solo e non è prevista una competizione, l'obiettivo è ottenere un tempo più basso possibile.

È stato inserito un obiettivo extra per permettere agli studenti più veloci di approfondire l'argomento e fornire più tempo agli studenti che possono averne bisogno.

Soluzione La soluzione può essere trovata nei file *microbit-3-reaction_game-1p.hex* e *microbit-3-reaction_game-1p-extra.hex*

4.2 Attività 5 - Reaction Game 2

Questa implementazione sarà una modifica della prima, in cui verrà aggiunto un secondo giocatore.
L'obiettivo è realizzare un gioco in cui due giocatori si sfidano per vedere chi ha i riflessi migliori.

4.2.1 [Attività] Identificazione delle variazioni rispetto alla prima implementazione

Le differenze si trovano in fase di misurazione e visualizzazione.

1. **Misurazione**
   - Attendere la pressione del tasto A o del tasto B;
2. **Visualizzazione**
   - Calcolare quale giocatore ha premuto il tasto per primo;
   - Visualizzare A o B sul display per indicare il vincitore;

4.2.2 Attività 5 - Reaction Game 2

La seconda fase deve estendere la prima, aggiungendo la possibilità di giocare in due utilizzando tasto A e tasto B.
La visuazzazione del tempo impiegato verrà sostituita con la visualizzazione del giocatore che ha premuto per primo il tasto.

L'implementazione deve funzionare come segue:

1. I giocatori impugnano il MicroBit sul tasto A e B;
2. Il dislay del MicroBit mostra un segnale visivo di attesa (ad esempio una croce);
3. Dopo un numero variabile di secondi (in un intervallo scelto dallo studente) il display mostra un segnale visivo di conferma (ad esempio una spunta o un cerchio) ed avvia un timer;
4. Nel momento in cui i giocatori vedono il segnale visivo di conferma, devono premere il tasto; appena viene rilevata la pressione del tasto, il microcontrollore salva il tempo di stop per quel tasto;
5. Il microcontrollore confronta i due tempi e mostra sul display il giocatore che ha premuto per primo il tasto.
6. EXTRA: Impedire che il giocatore possa barare premendo il tasto prima che venga mostrato il segnale visivo di conferma.
7. EXTRA: Mostrare anche lo scarto tra i tempi dei due giocatori (differenza tra i due).

[Attività] Discussione su possibili soluzioni alternative Il paragrafo *Soluzione* presenterà due soluzioni, la soluzione A è quella attesa dato l'esercizio, la soluzione B è una soluzione alternativa che non usa il concetto di tempo.

Una volta ottenuta in aula la soluzione A, chiedere se gli studenti riescono a pensare ad una soluzione alternativa che non usi il concetto di tempo.

Valutare vantaggi delle due soluzioni:

- **A**
  - Facilmente espandibile per mostrare anche il tempo impiegato (o lo scarto tra i due tempi);
- **B**
  - Minor numero di variabili (numero di variabili non dipendente dal numero di giocatori);
  - Codice di più semplice comprensione;

Soluzione Sono state identificate due soluzioni possibili, salvate nei file *microbit-4-reaction_game-2p-a.hex* e *microbit-4-reaction_game-2p-b.hex*.

La prima soluzione è quella attesa, ovvero un evoluzione della prima implementazione in cui si misurano i tempi di reazione di entrambi i giocatori e si sceglie il giocatore con il tempo minore.

La seconda soluzione, invece, rimuove completamente il concetto di tempo, tenendo traccia direttamente di quale pulsante è stato premuto per primo.
Per quanto non sia un evoluzione della prima implementazione, è comunque valida e può essere utilizzata per mostrare agli studenti come esistano più soluzioni per un problema.

5 Livella

La livella (comunemente nota come *bolla*) è uno strumento utilizzato per verificare la planarità di una superficie.
Micro:Bit è dotato di giroscopio, sensore che consente di rilevare l'orientamento del dispositivo nello spazio.

L'obiettivo di questa sezione sarà quello di sviluppare una livella digitale utilizzando il giroscopio di Micro:Bit e il display.

5.1 Rollio e Beccheggio

L'inclinazione lungo i due assi del giroscopio è chiamata *rollio* (*roll*) e *beccheggio* (*pitch*).

Si parla di rollio quando il dispositivo ruota lungo l'asse verticale del dispositivo (inclinato verso destra o sinistra).
Si parla di beccheggio quando il dispositivo ruota lungo l'asse orizzontale del dispositivo (inclinato in avanti o indietro).

Micro:Bit rappresenta rollio e beccheggio come valori numerici compresi tra -180 e 180 gradi, dove il valore 0 rappresenta lo stato "piano" rispetto all'asse.

Il comando per accedere ai dati del giroscopio si trova in *Ingressi – altro* e si chiama *rotazione*.

5.2 Griglia LED

Fino ad ora la griglia LED è sempre stata utilizzata per disegnare immagini o scritte.
La successiva attività richiederà il controllo dei singoli LED per rappresentare l'inclinazione del dispositivo.

Si può accendere un singolo LED della griglia utilizzando il comando *disegna* presente nel pannello *LED* e specificando le coordinate del LED da accendere.
Notare che la griglia è composta da 5x5 LED e che le coordinate vanno da 0 a 4, dove (0, 0) si trova in alto a sinistra.

5.3 Attività 6 - Livella

L'obiettivo di questa attività è quello di sviluppare una livella digitale che mostri all'utente l'inclinazione del dispositivo lungo i due assi sfruttando la griglia di LED.

L'implementazione deve funzionare come segue:

- Il dispositivo deve misurare l'inclinazione lungo i due assi e salvarla in apposite variabili;
- Tramite la formula scritta di seguito il programma deve determinare le coordinate del LED da accendere per rappresentare l'inclinazione lungo i due assi;
- Il programma deve pulire lo schermo e accedere il LED calcolato al punto precedente;
- (EXTRA) Tramite la pressione dei pulsanti A o B l'utente deve poter scegliere di focalizzarsi solo su uno dei due assi.
  - Il pulsante A deve visualizzare solo le variazioni in beccheggio;
  - Il pulsante B deve visualizzare solo le variazioni in rollio;

Calcolo delle coordinate La coordinata del LED da accendere può essere calcolata partendo dall'angolo α per casi come segue:

$$\begin{cases} 0 & \text{se } \alpha > 20 \\ 1 & \text{se } \alpha > 10 \\ 2 & \text{se } -10 \leq \alpha \geq 10 \\ 3 & \text{se } \alpha < -10 \\ 4 & \text{se } \alpha < -20 \end{cases}$$

Suggerimenti É consigliato implementare la conversione da angolo a coordinata in una funzione a parte per rendere il programma più leggibile ed evitare ripetizioni di codice.

Soluzione La soluzione può essere trovata nei file *microbit-5.livella.hex* e *microbit-5.livella-extra.hex*.

6 Progetti

Di seguito vi sono alcuni progetti che possono essere svolti dagli studenti per consolidare le conoscenze acquisite durante le lezioni di programmazione a blocchi.

- bussola
- Stazione meteo in / out

7 Licenza

Riferimenti bibliografici

[1] Creative Commons. *Creative Commons — Attribution-NonCommercial-ShareAlike 4.0 International — CC BY-NC-SA 4.0*. URL: https://creativecommons.org/licenses/by-sa/4.0/.

[2] Micro:Bit-Educational-Foundation. *MicroBit Overview*. URL: https://microbit.org/get-started/user-guide/overview. Rilasciato con licenza CC BY-NC-SA 4.0.