Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea Magistrale in Informatica

Didattica dell'informatica tramite making con Raspberry Pi e Arduino

Relatore:
Chiar.mo Prof.
Renzo Davoli

Presentata da:
Alessandro Cocilova

Correlatore:
Dott.
Michael Lodi

Sessione I
Anno Accademico 2017/2018

A chi mi vuole bene

Sommario

Dopo aver introdotto origine e significato del termine making e descritto il "movimento maker", verrà discussa l'introduzione del making nelle scuole, esaminando anche la situazione nel contesto della scuola italiana. Questo lavoro di tesi propone una serie di lezioni off-the-shelf, ciascuna contenente indicazioni di materiali e competenze necessarie, competenze acquisite e possibili collegamenti interdisciplinari da approfondire in classe. Per ogni lezione viene fornito anche il codice del risultato atteso (anche se appoggiando un approccio costruzionista, sono possibili più soluzioni e caldeggiate anche modifiche da parte degli studenti degli obiettivi), una descrizione utile al docente per guidare gli studenti nell'esperienza e la consegna da dare agli studenti per lo svolgimento di ogni esercizio in cui la lezione consiste. Le lezioni sono di due tipologie: quelle indirizzate agli studenti delle medie, da svolgersi su Raspberry Pi equipaggiati di Sense Hat, e quelle agli studenti delle superiori, da svolgersi con Raspberry Pi, Arduino e sensori vari, il cui tema comune è costruire una stazione meteorologica. Infine si presenta un resoconto e una valutazione qualitativa dell'applicazione, con un gruppo di studenti delle superiori, di una delle attività proposte nella tesi.

Indice

Capitolo 1

Introduzione

Questa tesi nasce da un'aspirazione per il mio futuro: quella di fare l'insegnante di informatica nella scuola secondaria. Nell'ottica di questa aspirazione, ho voluto immaginare una maniera per rendere più interessante ed intuitivo l'approccio alla programmazione, avvalendomi degli strumenti economici e potenti forniti dal mondo del making: Raspberry Pi e Arduino.

Si tratta di strumenti molto versatili e dalle grandi potenzialità, che, con l'aggiunta di sensori e attuatori dal prezzo modico possono essere usati per realizzare in classe attività molto coinvolgenti per gli studenti.

Anche i linguaggi di programmazione usati sono stati scelti con cura: Scratch, che con il suo approccio a blocchi si presta alle esigenze dei più piccoli, e Python che con la sua pulizia e potenza costituisce il linguaggio testuale ideale per cominciare a fare sul serio.

L'idea iniziale era quella di realizzare una serie di lezioni che, guidando passo passo gli studenti nella realizzazione di una stazione meteorologica, insegnassero ogni volta nuovi concetti di programmazione. Prevedendo però due formati: uno più semplice indirizzato agli studenti delle scuole medie ed uno più evoluto e complesso per gli studenti delle superiori.

A questo scopo sono stati individuati due approcci diversi per realizzare una stazione meteorologica: per i più piccoli è previsto il Sense Hat, una scheda che si collega ai pin del Raspberry Pi e che include sensori di tem-

peratura, pressione ed umidità; mentre per i più grandi si è individuata una serie di sensori singoli, da collegare al dispositivo principale, sporcandosi un po' le mani con l'elettronica.

In seguito, conoscendo meglio le potenzialità degli strumenti a disposizione, ho ideato altre lezioni non ispirate direttamente al progetto della stazione meteorologica. Questi esercizi in particolare si prestano maggiormente a personalizzazioni da parte degli studenti, che possono usarli come punti di partenza per progetti e giochi più complessi. Quindi queste lezioni presentano un approccio maggiormente costruttivista.

A prescindere dal tipo, ciascuna lezione, pensata per essere adottata facilmente da ogni insegnante di informatica, consiste innanzitutto in una serie di liste: i materiali necessari per lo svolgimento della lezione, le competenze necessarie allo svolgimento della stessa, le competenze che dovrebbero essere acquisite dagli studenti al termine ed i possibili collegamenti interdisciplinari a cui ci si può legare per approfondire la lezione in altre materie. Includono inoltre il codice di una possibile implementazione, la descrizione della lezione, utile al docente, e la consegna, da fornire agli studenti.

La tesi è strutturata come segue: dopo l'introduzione, in un capitolo teorico viene trattato brevemente il termine making, la comunità dei makers e la situazione del making a scuola (anche in Italia), accennando poi al concetto di pysical computing e ai materiali proposti. Nel terzo capitolo viene formulata la proposta che porto avanti con la tesi e presentate le lezioni per le scuole medie. Nel quarto capitolo sono presenti le proposte per le superiori. Nel quinto capitolo si descrivono i risultati di un'applicazione di una lezione presso una scuola superiore, per poi giungere alle conclusioni dell'ultimo capitolo.

Capitolo 2

Making come strumento per la didattica dell'informatica

2.1 Il movimento maker

Con il termine *movimento maker* si riferisce al crescente numero di persone che sono coinvolte quotidianamente nella produzione attiva di artefatti e che trovano luoghi di incontro fisici e digitali per condividere i loro processi e prodotti con gli altri [3].

L'espressione *maker* è stata coniata da Dale Dougherty nel 2005, con la nascita della della rivista *MAKE* [2]. Tale rivista, rifacendosi alle riviste hobbystiche della metà del ventesimo secolo, le quali stimolavano l'attitudine delle persone ad iniziare nuovi hobby e sviluppare nuove abilità creando una comunità in cui condividere i propri interessi in comune, auspicava un ritorno ad un'epoca in cui tutti fossero capaci di riparare, migliorare o creare i propri dispositivi, case e artefatti, proprio come nel secolo scorso molti erano capaci di riparare, migliorare o creare la propria macchina, casa o vestiti. L'anno successivo Dougherty ha dato il via alla prima Maker Faire, a San Francisco, con l'obiettivo di espandere l'idea di apprendimento e la comunità, creando uno spazio dove i lettori della rivista potessero incontrarsi e conversare diffusamente [2]. Una Maker Faire nasce semplicemente come un posto in

cui ogni Maker presenta l'oggetto da lui creato al pubblico, che fa domande su di esso: questo scambio di progetti ed idee è l'essenza del movimento; a questo si sono poi aggiunte possibilità di tenere workshop e competizioni per partecipanti di tutte le età, visto che molti maker che partecipano a questi eventi sono giovani ed anche bambini. Da quella prima Maker Faire se ne sono tenute moltissime altre, in ogni parte degli Stati Uniti e del mondo: quelle ufficiali sono 3 all'anno, con eventi di portata minore con una maggior frequenza e sparsi per tutto il globo.

Attualmente il movimento trova il suo fulcro in Rete, dove è possibile scambiare istantaneamente informazioni, pareri e codice, ma le Maker Fair rimangono luoghi d'incontro fondamentali per rimescolare insieme idee e competenze di generi di persone che raramente vengono a contatto nella vita di tutti i giorni [2].

Secondo Chris Anderson, ex direttore della rivista *Wired*, il movimento maker è definibile come una nuova rivoluzione industriale; i maker si differenziano dai tinkerer, creatori dalla natura più improvvisata, che hanno come scopo realizzare oggetti di vario genere utilizzando principalmente materiali di recupero, dagli inventori, i quali erano più orientati ad un percorso di non condivisione come la realizzazione di brevetti, e dagli imprenditori delle epoche precedenti per tre caratteristiche principali: l'uso di strumenti digitali, la norma culturale di condividere progetti e di collaborare online e l'uso di standard di design comuni per facilitare la condivisione ed una rapida iterazione [1]. Anderson e Hatch fanno anche notare la natura democratica del making, grazie all'uso di hardware economico, fabbricazione digitale, nonché software e design condivisi. Infatti è indubbio che sia cresciuta la disponibilità per la gente comune, sia in termini di quantità che di prezzo, di potenti strumenti computazionali e di fabbricazione, basti pensare al caso delle stampanti 3D od a single-board computer come il Raspberry Pi.

Come fa notare Hatch, nel suo manifesto del movimento maker, il vero potere di questa rivoluzione sono gli effetti democraticizzanti: adesso praticamente chiunque può innovare, chiunque può creare, chiunque, con gli

strumenti disponibili in un makerspace, può cambiare il mondo [4].

Questo manifesto del movimento maker, pubblicato nel 2014, fornisce nove parole chiave per il movimento: *make, share, give, learn, tool up, play, participate, support, change*. Ora andiamo ad esaminare il significato di ognuna.

Make: "il making è fondamentale nella natura di essere umani, dobbiamo fare, creare per esprimere interamente noi stessi. C'è qualcosa di unico nel fabbricare oggetti. Queste cose sono come piccoli pezzi di noi e sembrano integrare parti delle nostre anime."

Share: "condividere quello che hai fatto e quello che sai con gli altri è il metodo in cui il desiderio di interezza di un maker è soddisfatto. Non puoi fare e non condividere."

Give: "ci sono poche cose più disinteressate e soddisfacenti di dare via un oggetto che hai creato. L'atto di fabbricare mette un piccolo pezzo di te in quell'oggetto. Darlo a qualcun altro è come dare a qualcuno un piccolo pezzo di te. Questi oggetti sono spesso i più cari che possediamo."

Learn: "bisogna imparare per fare. Bisogna sempre cercare di imparare qualcosa in più dall'oggetto che si sta fabbricando. Potrai diventare un operaio o un artigiano, ma vorrai sempre imparare e spingerti ad apprendere nuovi materiali, tecniche e processi. Costruire un percorso di apprendimento a vita assicura una vita ricca e gratificante e, molto importante, permette di condividere."

Tool-Up: "ognuno deve avere accesso ai giusti strumenti per il progetto in corso. Bisogna investire negli strumenti per fare il making come lo vuoi fare. Gli strumenti per il making non sono mai stati più economici, facili da usare o potenti."

Play: "bisogna avere un atteggiamento giocoso con quello che si sta fabbricando e si sarà sorpresi, eccitati ed orgogliosi di quello che si scoprirà."

Participate: "Unisciti al movimento maker e raggiungerai quelli attorno a te che stanno scoprendo le gioie del making. Tieni seminari, feste, eventi, fiere, esposizioni, lezioni e cene con e per gli altri maker della tua comunità."

Support: "questo è un movimento e richiede supporto emozionale, intellettuale, finanziario, politico ed istituzionale. Siamo la migliore speranza per migliorare il mondo e siamo responsabili di creare un miglior futuro."

Change: "accogli il cambiamento che accadrà naturalmente mentre percorri il tuo viaggio da maker. Dal momento che il making è fondamentale per ciò che significa essere umani, facendo, diventarai una versione più completa di te [4]."

2.2 Il making a scuola

2.2.1 Strumenti e scopo del making a scuola

Uno dei punti cardine del movimento maker è costituito dai makerspace, i luoghi fisici in cui i maker si incontrano per comunicare, condividere e portare avanti i propri progetti; strutture generalmente dotate di molteplici dispositivi tecnologici come macchine CNC, stampanti 3D, piccoli robot o materiali per costruirli, microcontrollori, single-board computer, componenti elettronici di consumo e tanto altro. Negli ultimi anni c'è stata una esplosione della diffusione di questi luoghi, che possono essere gestiti da diverse entità, come librerie, musei, organizzazioni no-profit indipendenti, organizzazioni for-profit, università, ma, soprattutto, scuole.

Ma il making a scuola non è avere gli strumenti più costosi e all'ultimo grido o il più bel makerspace. Il making in classe è avere potere e fiducia e forse, in maniera più importante, trasferire potere ai discenti - giovani che guideranno il mondo in un futuro non troppo lontano. E dando ai discenti agency (il potere di agire) e responsabilità sulla materia del loro apprendimento, guadagnano fiducia in sé stessi come potenti problem solvers e agenti di cambiamento [7].

2.2.2 La visione di Papert

Quella di insegnare informatica e materie affini mediante il making non è un'idea campata in aria, anzi, si rifà ad una nota e stimata teoria educativa: il *costruzionismo*. Il costruzionismo, citando il suo teorico Papert, è definito come una teoria che prende dalle teorie costruttiviste della psicologia - che vede la conoscenza come costruzione dell'esperienza personale - la visione dell'apprendimento come ricostruzione invece che come una trasmissione di conoscenza. Poi estende l'idea dei materiali manipolativi all'idea che l'apprendimento è più efficace quando è parte di una attività in cui il discente costruisce un prodotto significativo [5]. Questo è poi applicato specificamente all'informatica, infatti, citando lo stesso Papert, ogni bambino normale impara a parlare, per quale motivo allora non dovrebbe imparare a "parlare" ad un computer? [6] Fondamentale in questo senso il suo apporto con l'invenzione di Logo, linguaggio di programmazione esplicitamente progettato per i bambini, di cui l'attuale Scratch è un pronipote.

Papert viene considerato il padre del movimento making [10], visto che, già nel lontano 1971, aveva una visione di come sarebbe dovuto essere un computer ed un laboratorio ideale per l'insegnamento dell'informatica a scuola: "il computer per la scuola dovrebbe avere un grande numero di porte per l'output, per permettere al computer di accendere e spegnere luci [..], mostrare slide su un proiettore e far partire ed arrestare tanti tipi di piccoli macchinari. Dovrebbero esserci anche porte di input per permettere di spedire segnali al computer. Nella nostra immagine di laboratorio di computazione scolastico, un ruolo importante è giocato da le numerose porte controller, che permettono ad ogni studente di collegare qualsiasi dispositivo al computer.[..] Il laboratorio avrà una fornitura di motori, solenoidi, relè, sensori di vario tipo, ecc. Usandoli gli studenti saranno capaci di inventare e costruire una varietà infinita di dispositivi cibernetici [9]."

Quasi cinquant'anni dopo la sua è ancora una visione da realizzare, pur essendo disponbili hardware con le caratteristiche indicate (la sua sembra proprio una descrizione del Raspberry Pi), ancora molte scuole non dispongono

di laboratori di questo tipo [10].

2.2.3 La valutazione in un lavoro di making

Ma cosa succede quando la responsabilità di costruire il curriculum non è più completamente del docente, in una visione dell'ambiente di apprendimento completamente guidata da esso? Cosa accade quando si adotta un approccio più guidato dallo studente, in cui essi avranno un certo grado di scelta nei contenuti che studieranno, nelle abilità che costruiranno? Sicuramente sarà difficile trovare degli strumenti di valutazione quantitativi per effettuare un giudizio degli studenti, dal momento che non è più possibilie usare prove uniche e standardizzate per valutare il lavoro degli studenti. E' possibile usare strumenti di valutazione qualitativa, ottenendo una valutazione formativa che è vitale per il processo di apprendimento, come feedback orali da pari e adulti, narrativa del progetto ed autovalutazione. Ovviamente a questi strumenti manca lo status dato ad un voto numerico.

Questa è una delle ragioni per cui molte scuole saltano la parte di valutazione del lavoro di making e si limitano ad offrire il making come un programma extracurricolare [7].

2.3 Il making a scuola in Italia

Secondo una valutazione fatta dal Ministero dell'Istruzione l'Italia è agli ultimi posti in Europa per presenza di connessione broadband nelle scuole, numero di studenti per computer, numero di studenti per laptop connesso in Rete.[8] Sempre lo stesso studio afferma che oltre che per la presenza dei computer l'Italia presenta ritardi anche nella loro dislocazione: la maggioranza (più del 75%) è nelle aule informatiche, mentre è bassa la presenza dei computer in classe, fattore essenziale per l'integrazione delgi studenti digitali nella didattica quotidiana. Ciononostante, è obiettivo del ministero promuovere l'alfabetizzazione digitale per rispondere alle richieste del mercato del lavoro.

In questa direzione esistono, nonostante un panorama in alcuni versi desolante, iniziative ed esempi virtuosi. Rimanendo nell'ambito della regione Emilia-Romagna, da un paio d'anni esiste lo "School Maker Day", dove sono tenuti workshop per docenti e spazi espositivi in cui i studenti possono mostrare ad altri pari i progetti da loro sviluppati nell'ambito di iniziative scolastiche. Sempre nel territorio bolognese esiste anche il Laboratorio "Opus Facere", formato dall'unione delle forze di una decina di scuole secondarie di secondo grado e istituti comprensivi con partner pubblici e privati del territorio. Tra i numerosi laboratori che propongono nelle loro strutture, ve ne sono anche alcuni di making. I corsi sono inquadrati nell'ambito di alternanza scuola lavoro. Come abbiamo già osservato nel caso americano, è difficile portare questi tipi di esperienze fra i banchi di scuola, ovvero durante le ore di lezione, come esperienza curricolare.

Un'iniziativa lodevole in questo senso è quella degli *Atelier Creativi*, che fanno parte del Piano Nazionale Scuola Digitale. Si tratta di realizzare ambienti di apprendimento che favoriscano la creatività, la collaborazione e la manualità, facendo sì che gli studenti possano sviluppare competenze trasversali mediante la realizzazione di progetti. Molte scuole hanno approfittato di questi fondi per realizzare veri e propri laboratori di making e tinkering.

Infine le *Maker Faire* sono presenti anche in Italia e spesso propongono iniziative per le scuole. Per fare un esempio, durante l'ultima Maker Faire svoltasi a Roma era presente una *Call for Schools* per le scuole superiori, che prevedeva una selezione tra i progetti più interessanti ed innovativi svolti a scuola e l'esposizione durante la fiera dei progetti ritenuti più meritevoli.

2.4 Physical computing e materiali proposti

Nel vario settore del making, la mia proposta di tesi è incentrata su di un ambito: quello del *physical computing*, che consiste in creare macchine che interagiscono con il loro ambiente [10]. Questo è reso fattibile dalla possibilità di avere hardware economico, come Raspberry Pi e Arduino, interfacce

innovative connettono il computer al mondo reale, come Scratch e linguaggi di programmazione più classici come Python.

Ma perché proprio il Raspberry Pi e l'Arduino?

Raspberry Pi è la risposta al desiderio di Papert di avere un computer piccolo ed estensibile, in grado di comunicare con dispositivi di input. E' di fatto una scheda grande come una carta di credito - viene definito single board computer - che integra tutto quanto serve ad un computer (processore, RAM, porte), tranne memoria di archiviazione (a questo scopo viene utilizzata una scheda SD, non inclusa) e alimentatore. Collegandolo a tastiera, mouse e schermo si ottiene una workstation, che può girare un sistema operativo linux, più che valida per una cifra irrisoria. Ma la magia del Raspberry Pi non sta solo qui, esso ha anche la capacità di essere facilmente collegabile a dispositivi elettronici (come LED, sensori o altro), mediante i numerosi (26 o 40 a seconda delle versioni) pin di general purpose input output, alimentazione e messa a terra. Mediante l'uso di breadboard e ponticelli tutto questo è possibile senza neanche dover saldare le componenti, grande vantaggio se si lavora in una classe, sia per motivi pratici che di sicurezza.

Arduino invece è un piccolo microcontrollore che può ricevere input da sensori, usare queste informazioni in un programma (dentro sketch, il programma per Arduino, che consente di caricare sul dispositivo, programmi scritti in C), e agire controllando luci, motori, altoparlanti ed altri dispositivi di output. A differenza del Raspberry Pi, che supporta solo general purpose input output di tipo digitale, Arduino può comunicare anche su pin analogici e questo fa sì che il numero di dispositivi e componenti con cui può interagire sia molto più alto, da luci a innaffiatoi automatici fino a igrometri e anemometri. Molti progetti di Physical programming non necessitano un computer vero e proprio, basta una piccola quantità di capacità computazionale. Arduino fornisce questa potenza necessaria, unito a leggerezza, economicità e facilità d'uso.

Entrambi, da soli o combinati, permettono la realizzazione di progetti e prototipi concreti, che rappresentano proprio quello che è lo scopo del physical

computing.

Scratch è un linguaggio di programmazione visuale, in cui le unità elementari non sono parole chiave ma blocchi di codice colorato, ognuno con un nome che spiega la propria funzione, traducibili in moltissime lingue. Nonostante l'aspetto giocoso, si tratta di tutt'altro che un giocattolo, infatti è un linguaggio di programmazione Turing completo. E' stato progettato dal MIT Media Lab e la prima versione è stata rilasciata nel 2003. Dalla semplicità con cui è possibile creare programmi curiosando con i blocchi e vederne subito un risultato visivo sulle sprite grafiche, vero e proprio output del sistema, si può comprendere che è stato creato tenendo a mente un approccio costruttivista.

Python è un linguaggio di programmazione orientato agli oggetti, noto per la sua sinteticità e per la pulizia e la compattezza della sua sintassi, ideato da Guido van Rossum all'inizio degli anni novanta. Per le sue qualità precentemente citate è stato scelto in questo lavoro di tesi come linguaggio con cui portare avanti la didattica della programmazione alle scuole medie e superiori. Oltretutto, come verrà mostrato nei capitoli successivi, molte delle librerie per i sensori ed i dispositivi usati in questa proposta sono scritti in Python, quindi in un certo senso si tratta anche di una scelta obbligata.

Capitolo 3

Making alle Scuole Medie

3.1 La mia proposta

La mia proposta si compone in una serie di esercizi, divisi in esercizi per le scuole medie e per quelle superiori, che possono essere tenuti in lezioni, in ordine sequenziale o anche sparso, per insegnare la programmazione mediante l'interazione con dispositivi fisici, avvantaggiandomi perciò in questo difficile compito del potere di coinvolgimento e chiarificazione dato dal making. Gli esercizi vogliono essere forniti come materiale completo, *off-the-shelf*, pronto per essere usato in classe.

Per ogni esercizio proposto è fornita la lista dei materiali necessari, delle competenze necessarie, delle comptenze che ci si aspetta che lo studente acquisisca e dei possibili collegamenti interdisciplinari. In più viene fornita una descrizione utile ad docente che vuole mettere in atto l'esercizio, la consegna che bisogna fornire agli studenti ed il codice dell'esercizio completato, chiaramente questo consiste solo in una soluzione di riferimento, dato che gli esercizi sono quasi sempre risolvibili in più maniere differenti.

Nel caso particolare degli esercizi per le scuole superiori c'è un filo che lega tutti gli esercizi e si tratta della realizzazione di una stazione meteorologica, pensato per coinvolgere maggiormente gli studenti, realizzando un progetto lungo l'arco dell'anno scolastico, più complesso e concreto, che possa poi

essere effettivamente utilizzato per uno scopo.

Nella mia proposta si può notare che si è deciso di sacrificare parzialmente la prospettiva costruttivista, rinunciare in parte alla libertà che essa propone di fornire agli studenti, in modo da ottenere una proposta fattibile di making durante le lezioni e non come attività extracurricolare. Questo approccio più chiuso va quindi a risolvere il problema principale dell'approccio costruzionista, che consiste nella difficoltà di applicare un metodo di valutazione quantitativo.

3.2 Sense Hat ed il progetto Astro Pi

Figura 3.1: Un Sense Hat montato su di un Rasperry Pi

Il progetto Astro Pi (`https://astro-pi.org/`) è una competizione di pro-

grammazione annuale per studenti fino a 19 anni, aperta a tutti i paesi membri dell'ESA, in cui il codice scritto è mandato in esecuzione sulla stazione spaziale internazionale.

Si presta quindi ottimamente come stimolo per insegnare la programmazione attraverso il making.

Una volta presentata l'idea dell'esperimento da realizzare, se si viene selezionati per la fase successiva, si riceve gratuitamente l'hardware necessario per partecipare. Questo consiste in un Raspberry Pi, il popolare single-board computer dal contenutissimo prezzo e dagli utilizzi pressoché illimitati, e in un Sense Hat, board di espansione compatibile con tutti i Raspberry Pi a 40 pin, che include una matrice di led 8x8 RGB, un joystick a quattro direzioni, giroscopio, accelerometro, magnetometro ed infine sensori di temperatura, pressione ed umidità. Quest'ultimo è anche acquistabile al di fuori del progetto Astro Pi, per una cifra inferiore ai 30 euro. Il modo di interfacciarsi con il Sense Hat è estremamente semplice e ad alto livello, sia in Scratch (`https://github.com/raspberrypi/documentation/blob/master/usage/gpio/scratch1/README.md` qui si possono trovare le API per Scratch), sia in Python (`https://pythonhosted.org/sense-hat/api/` qui si trovano le API per Python), dove tutte le funzionalità possono essere facilmente usate mediante chiamate a metodi, dopo aver installato ed importato la corrispondente libreria.

Una volta realizzato il codice per l'esperimento, questo verrà effettivamente eseguito nello spazio, su calcolatori identici a quelli con cui gli studenti hanno lavorato, per due orbite (circa tre ore), al termine delle quali gli output dell'esperimento saranno resi disponibili al download da parte degli studenti, per trarre le conclusioni finali.

L'obiettivo degli esercizi che verranno descritti qui in seguito è quindi quello di fornire agli studenti senza basi, o con solo rudimenti, di programmazione gli strumenti per partecipare fruttuosamente al concorso Astro Pi. Sicuramente eccitati dalla prospettiva, nemmeno si accorgeranno dei concetti di programmazione fondamentali che immagazzineranno.

Lo scopo finale è insegnare loro il linguaggio Python, linguaggio richiesto anche per la partecipazione al concorso Astro Pi, ma per far prendere loro dimestichezza con la programmazione, imparando le primitive fondamentali, si è ritenuto opportuno farli inizialmente interagire con il Sense Hat utilizzando il linguaggio a blocchi Scratch. Infatti è possibile utilizzarne tutte le funzionalità anche tramite questo linguaggio.

3.3 Esercizi in Scratch

Per ogni esercizio verrà fornita la lista dei materiali richiesti, delle competenze richieste, delle competenze acquisite, dei collegamenti interdisciplinari, oltre ad una descrizione dell'esercizio ed alla consegna dello stesso. Gli estratti di codice forniti consistono nell'esito atteso degli esercizi, chiaramente altre soluzioni sono possibili.

3.3.1 Usare la Bussola

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Concetto di variabile
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di ciclo infinito
  - Concetto di busy wait

- Collegamenti interdisciplinari:
  - Fisica: Concetto di magnetismo terrestre

Descrizione: in questo primo esercizio si vuole far realizzare allo studente un piccolo gioco, in cui lo sprite di Scratch, grazie al sensore magnetometro[1] integrato nel sense hat, si muove solidale ai movimenti applicati al Raspberry. Ruotando quindi di 15 gradi a sinistra il Raspberry, si otterà un movimento equivalente dello sprite sullo schermo. Come si può vedere in Fig. 3.2, la realizzazione si compie con pochi blocchi, mediante l'uso di un loop infinito e di una variabile in cui viene registrato ogni secondo il valore rilevato dal sensore (blocco sensor value), che va da 0 a 360, per poi far puntare la sprite in quella direzione mediante il blocco dedicato.

Consegna: Realizzare un gioco in Scratch in cui la sprite ruota al ruotare del Raspberry Pi dello stesso angolo nella stessa direzione. Per farlo è necessario usare la bussola integrata nel Sense Hat, che si può leggere attivando il blocco azzurro `sensor value`, sull'opzione `compassZ`.

[1] Il magnetometro integrato necessita di essere calibrato, le istruzioni su come farlo possono essere trovate al seguente indirizzo: `https://www.raspberrypi.org/forums/viewtopic.php?f=104&t=109064&hilit=calibration`

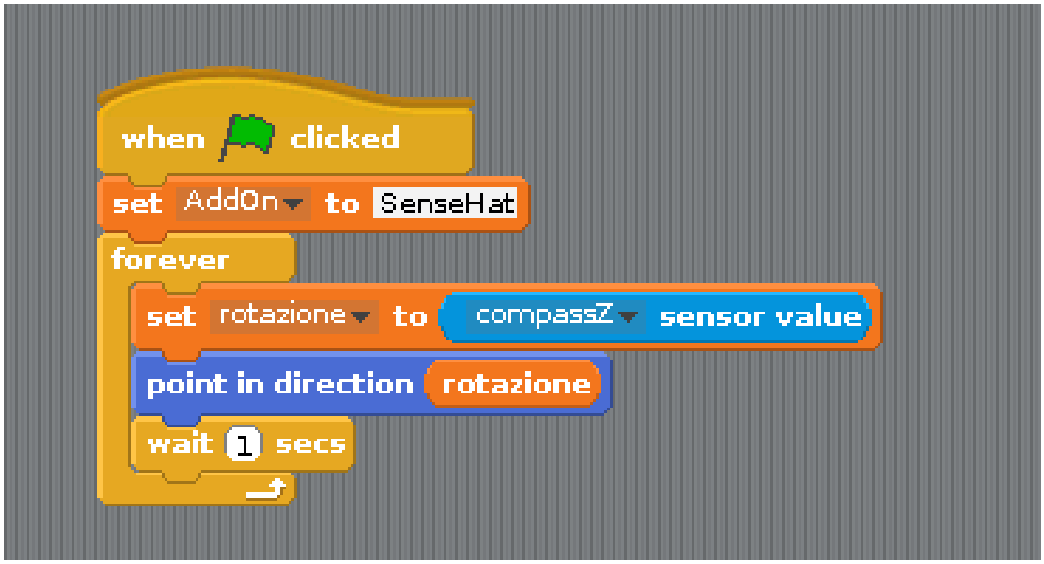

Figura 3.2: Il codice dell'esercizio della bussola

Figura 3.3: L'output dell'esercizio della bussola

3.3.2 Stampa pressione, temperatura e umidità

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Concetto di variabile
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di ciclo finito
  - Concetto di output
  - Concetto di concatenazione di stringhe
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: in questo secondo esercizio si vanno ad usare i sensori di temperatura[2], pressione ed umidità integrati a bordo del Sense Hat. Precisamente lo scopo di questo esercizio è di stampare, o, meglio, far dire dalla sprite

[2]I valori di temperatura che si riscontrano sono sempre una sovrastima di circa 7-8 gradi rispetto a quelli reali. Questo è probabilente dovuto all'eccessiva vicinanza del sensore di temperatura alla CPU. Per debellarlo sono possibili sia soluzioni hardware (isolare il sense hat, aggiungere spessore per montarlo più lontano) o software, riducendo la temperatura rilevata di un fattore dipendente dalla temperatura della CPU. La seconda verrà messa in atto, ma in uno degli esercizi svolti in python

mediante un fumetto, i valori di queste grandezze per tre minuti; questo può essere ottenuto semplicemente (con un minimo di approssimazione fisiologica all'implementazione della wait), iterando la sequenza dei comandi per 60 volte, con una wait di un secondo alla fine di ogni comando say. Per realizzare la stringa che verrà data in output bisogna concatenare il testo con la grandezza rilevata e con l'unità di misura, mediante quindi due blocchi join annidati.

Consegna: Realizzare un programma che faccia dire, con un fumetto, alla sprite per 60 volte i valori rilevati rispettivamente di temperatura, pressione ed umidità. Questi possono essere letti con il blocco azzurro `sensor value`.

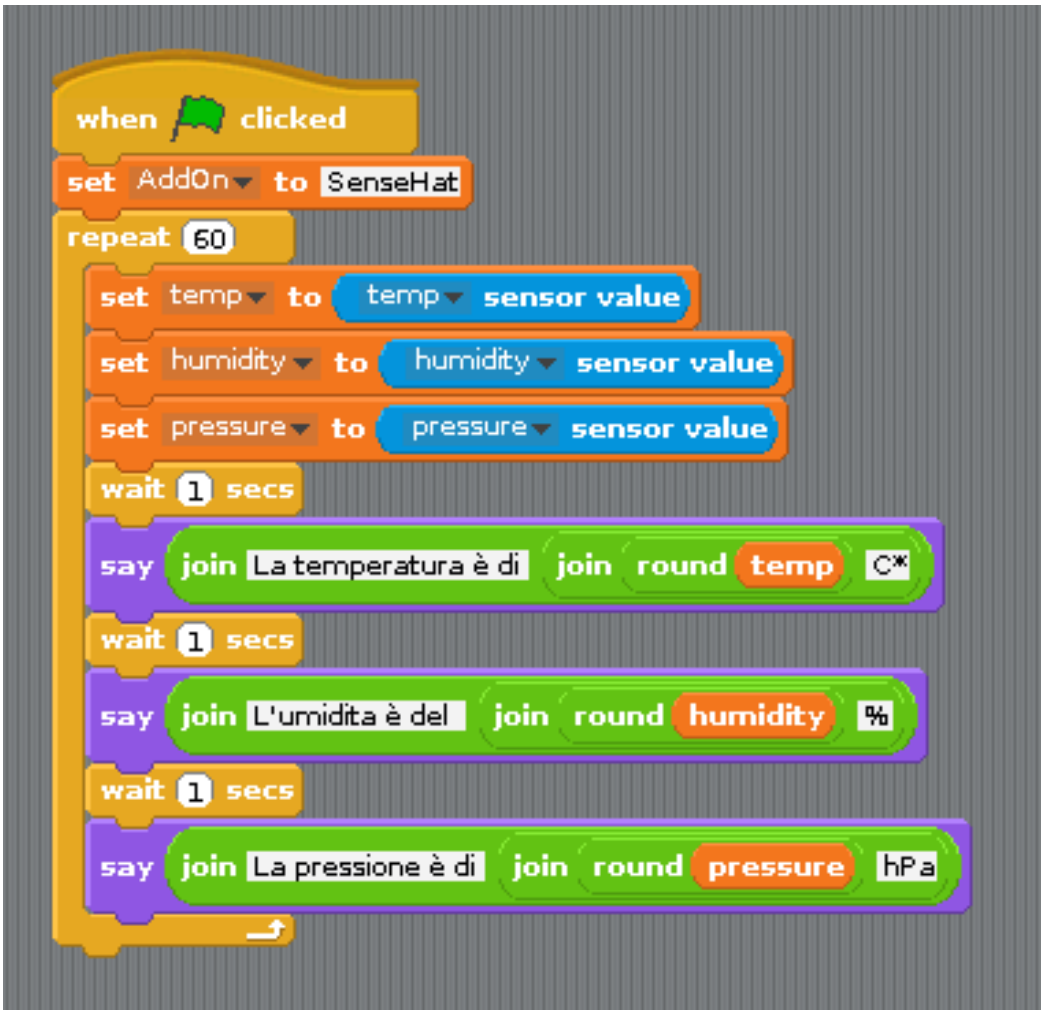

Figura 3.4: Il codice dell'esercizio di stampa rilevazioni

Figura 3.5: L'output dell'esercizio di stampa rilevazioni

3.3.3 Salva i valori di una grandezza su un file

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Concetto di variabile
  - Concetto di arrotondamento
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di lista
  - Concetto di file
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: in questo terzo esercizio l'obiettivo è di produrre un file contenente le rilevazioni di pressione, temperatura ed umidità, una riga per rilevazione, con, prima di ogni terna di rilevazioni, una riga contenente un timestamp. Questo è possibile in Scratch mediante l'uso di una lista, che può essere riempita, mediante la `add`, con ogni rilevazione e che si può poi esportare in txt cliccando sul widget corrispondente nella finestra della sprite. Una possibile variante per mettere alla prova le abilità degli studenti nella

concatenazione delle stringhe può essere chiedergli di produrre una lista che contenga per ogni riga il timestamp ed una rilevazione per ogni grandezza, invece che averle su righe diverse. Si può interpretare questo esercizio come una prima approssimazione della realizzazione di una stazione meteorologica.

Consegna: Realizzare un programma che permetta di salvare in un file di testo una serie di rilevazioni di temperatura, pressione e umidità. Per farlo si consiglia di realizzare una lista ed aggiungere una rilevazione di grandezza per riga, quindi, quando si ritiene di aver un numero sufficiente di rilevazioni, cliccare col destro sul widget ed esportare come file di testo.

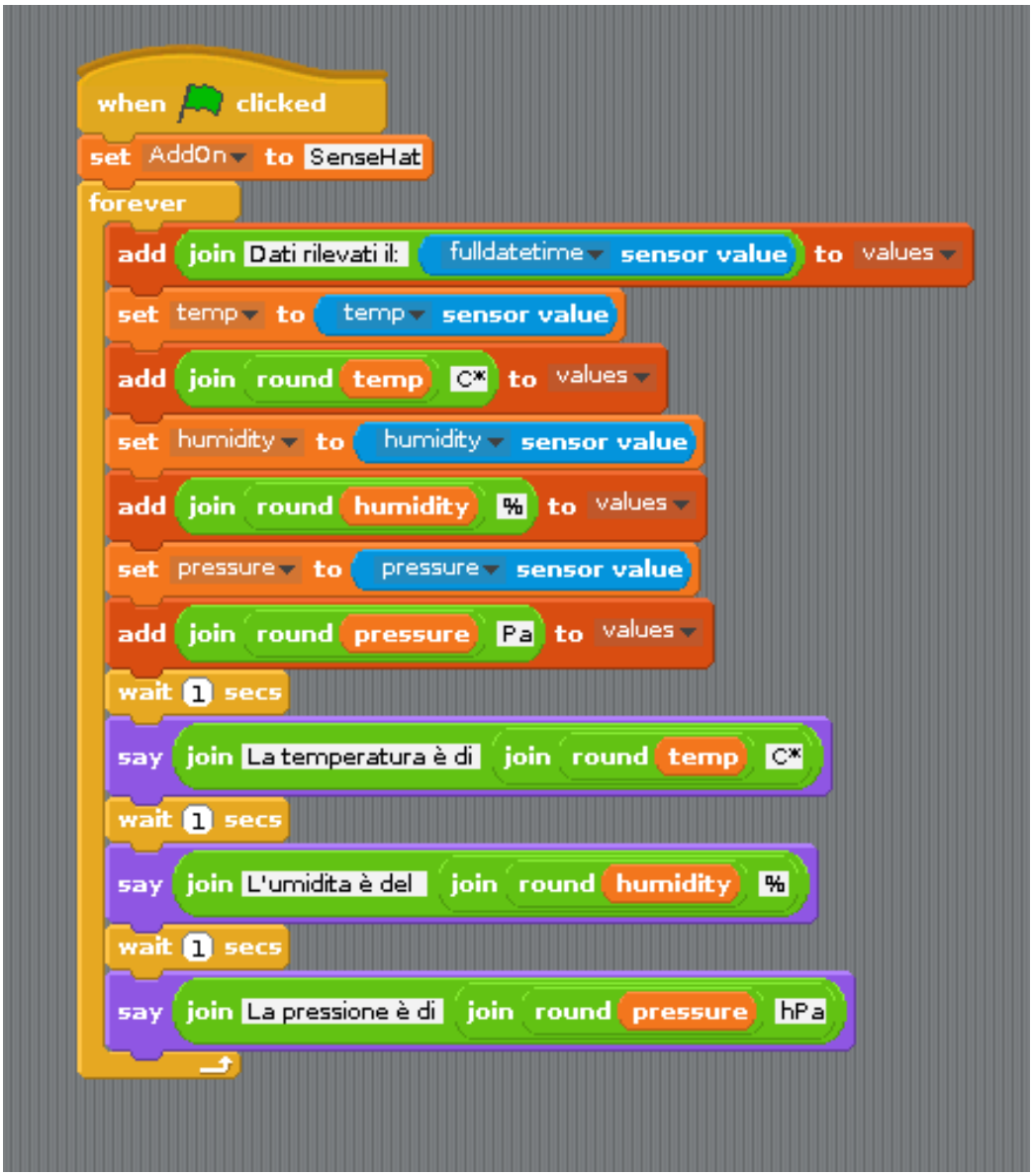

Figura 3.6: Il codice dell'esercizio di salvataggio delle rilevazioni su file

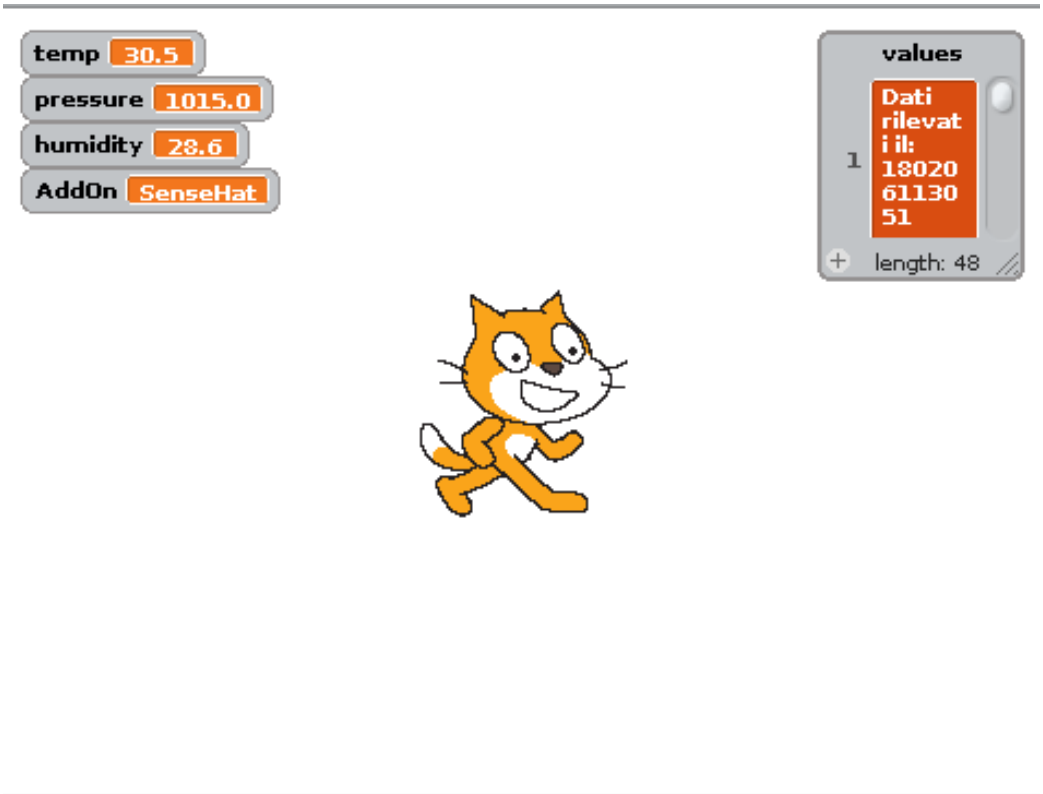

Figura 3.7: L'output dell'esercizio di salvataggio delle rilevazioni su file

3.3.4 Trova i valori massimi di pressione, temperatura e umidità

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Concetto di confronto
  - Concetto di arrotondamento
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di inizializzazione di una variabile
  - Concetto della struttura di controllo IF
  - Algoritmo del massimo
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: in questo quarto esercizio in Scratch, si richiede di dare in output, ad ogni iterazione, per ogni grandezza, oltre al valore rilevato in quel momento, quello che è il valore massimo rilevato fino a quel punto. Appare subito evidente che per realizzare un programma di questo tipo è necessario

l'uso della struttura di controllo if, in cui si entra se si verifica la condizione che il valore rilevato è maggiore del massimo (ad esempio temp > maxtemp), nel qual caso si aggiorna il valore della variabile max rispettiva. Importante è anche inizializzare la variabile a 0, se si vuole che questo meccanismo funzioni, questo può essere un primo modo per introdurre gli studenti al concetto di inizializzazione di una variabile, fondamentale in linguaggi di programmazione più complessi.

Consegna: Realizzare un programma che faccia dire alla sprite, con un fumetto, i valori rilevati di ogni grandezza ed il valore massimo rilevato, fino a quel momento, di ogni grandezza.

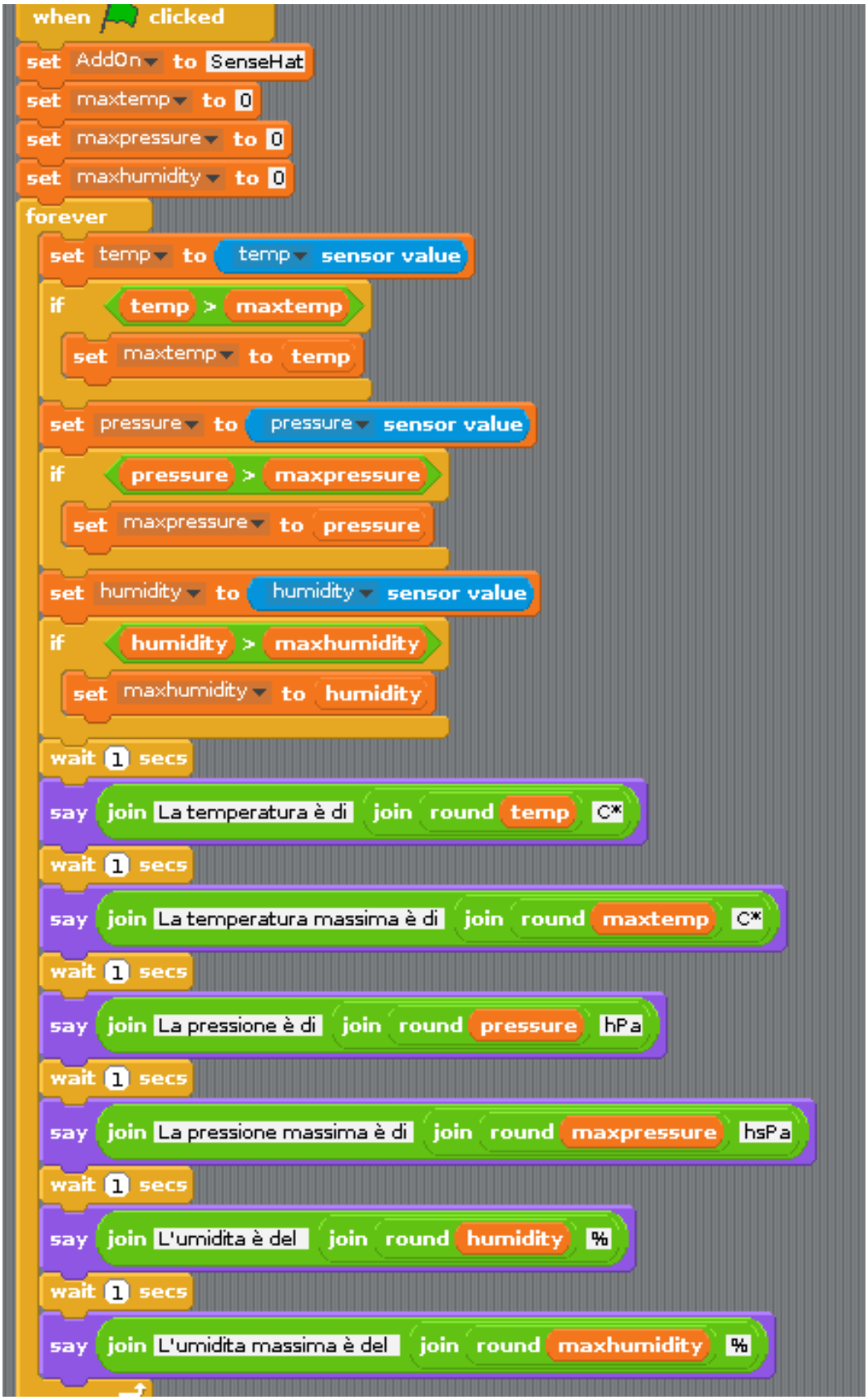

Figura 3.8: Il codice dell'esercizio di stampa dei massimi delle grandezze

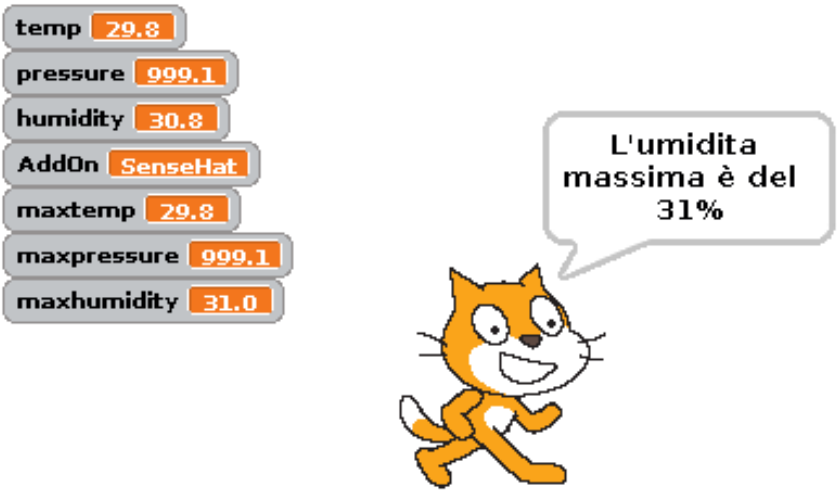

Figura 3.9: L'output dell'esercizio di stampa dei massimi delle grandezze

3.3.5 Gioco della biglia

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Concetto di confronto
  - API per GPIO del Sense Hat
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di numeri pseudocasuali
  - Concetto di operatori booleani
  - Concetto di concatenazione di stringhe
  - Concetto di ciclo loop-until
- Collegamenti interdisciplinari:
  - Fisica: Concetto di accelerazione di gravità
  - Fisica: Modalità di misurazione dell'accelerazione

Descrizione: in questa attività in Scratch si richiede allo studente di creare un gioco con cui poter interagire fisicamente: utilizzando i led del Sense Hat e l'accelerometro dovrà simulare un gioco con la biglia, in cui, data la posizione di partenza, il giocatore vince quando riesce, inclinando il gioco, a portare la biglia dentro un buco. Nel nostro caso la biglia ed il buco sono rappresentatati da led illuminati di colori differenti. La criticità principale di questo esercizio è implementare correttamente i movimenti della biglia, in

modo che ad un'inclinazione in una direzione corrisponda un movimento nella direzione corrispondente. Lo studente verrà guidato nella scelta dei sensori giusti tra quelli presenti e nell'effettuare alcuni test di debug mediante dei blocchi `say`, in modo da comprendere quali valori usare come soglie per considerare l'inclinazione in una direzione. Altre difficoltà possibili derivano dal comprendere come deve essere strutturato il ciclo base del movimento (ciclo con condizione sul raggiungimento della buca, reset matrice, ridisegnamento dei punti, aggiornamento posizione, `wait` finale)

Consegna: Realizzare un gioco in Scratch in cui viene acceso randomicamente un LED "buca", sulla matrice di LED del Sense Hat e bisogna portare un LED "pallina", inclinando il Raspberry Pi, dal centro della matrice a sopra il LED buca. Il sensore da usare è l'accelerometro. Per farlo è necessario includere il blocco giallo broadcast `gpioserveron` all'inizio del programma. Con broadcast `ledpixelx0y0colourwhite` si accende il LED 0,0 di bianco,mentre con `clearleds` si spengono tutti i led. Infine, se si vuole visualizzare un messaggio sulla matrice, il blocco è `broadcast ledscrollstringMessaggio`.

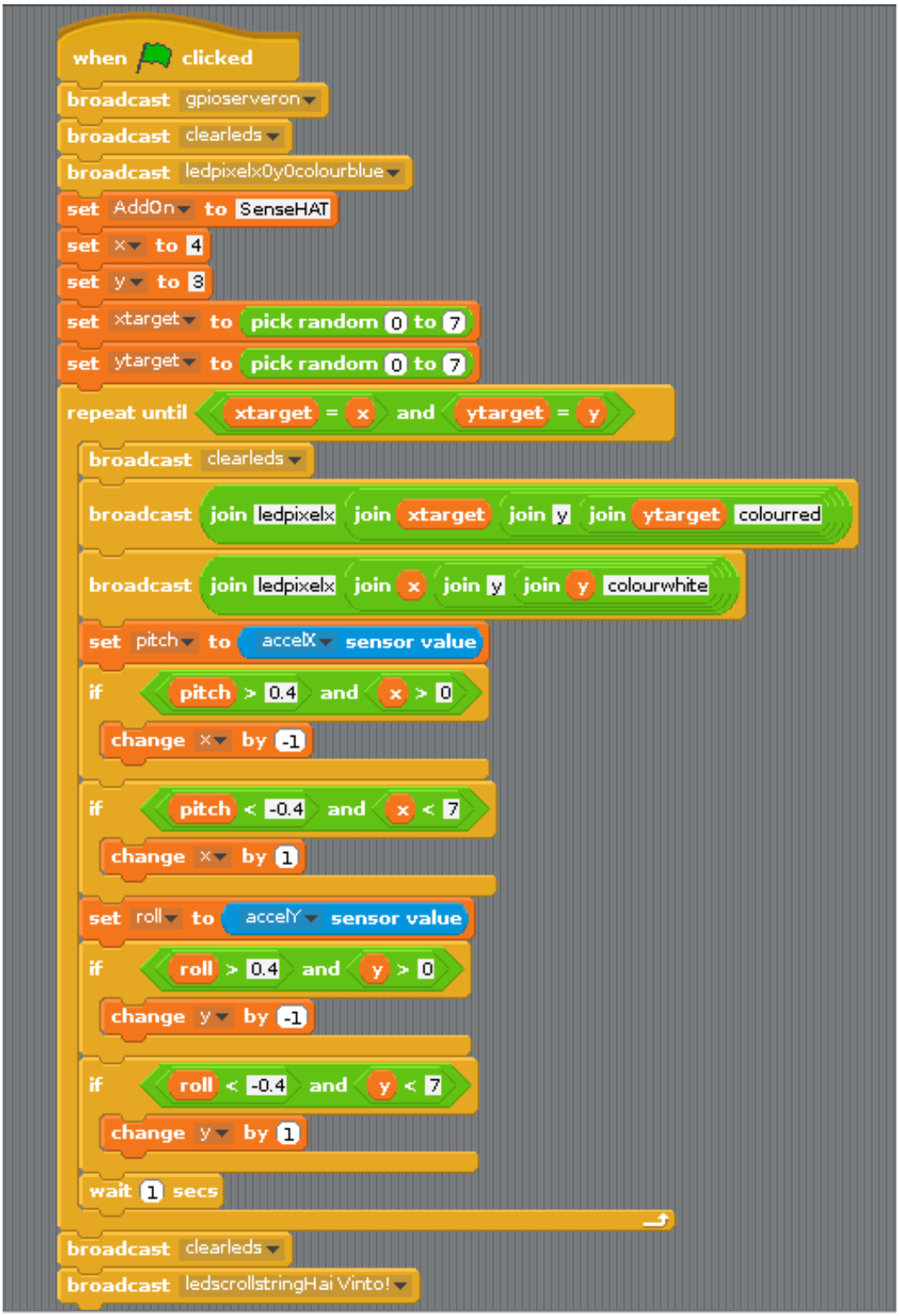

Figura 3.10: Il codice dell'esercizio del gioco della biglia

Figura 3.11: Una foto del gioco della biglia in esecuzione su un Raspberry Pi

3.4 Esercizi in Python

Gli esercizi in Python sono strutturati allo stesso modo di quelli in Scratch, con l'unica differenza che sono presenti parti commentate, negli estratti di codice, che corrispondono alle linee di codice che vengono già fornite inizialmente agli studenti.

3.4.1 Stampa su terminale

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di variabile
  - Concetto di inizializzazione e assegnamento di una variabile
  - Concetto di output
  - Concetto di esecuzione di un file python

Descrizione: Lo studente riceverà il file come mostrato qui sotto e dovrà limitarsi a sostituire la stringa "Nome Cognome" col suo nome ed il suo cognome e ad eseguire il file osservando l'output sul terminale. Serve come un primo esercizio di riscaldamento, di introduzione a Python.

Consegna: Assegnare ad una variabile il proprio nome come valore e stamparlo a video con una `print`.

```
1 messaggio = "Nome Cognome"
2 print(messaggio)
```

3.4.2 Stampa su matrice LED

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di tupla
  - Concetto di oggetto
  - Concetto di ciclo infinito
  - Concetto di indentazione in python
  - Concetto di invocazione di metodo
- Collegamenti interdisciplinari:
  - Fisica: funzionamento di un LED
  - Scienze: modello RGB e motivazioni alla base della sua scelta

Descrizione: In questo esercizio lo studente viene guidato nella realizzazione di un'evoluzione dell'esercizio precedente: la scritta viene ripetutamente stampata sulla matrice di led del Sense Hat. Il docente spiegherà in quale modo importare la libreria per il Sense Hat, così come esempi dell'API usata del Sense Hat. Gli studenti verranno poi incoraggiati a personalizzare l'output anche dal punto di vista estetico, variando colori e velocità di scorrimento, per prendere confidenza con l'invocazione di metodi.

Consegna: Assegnare il proprio nome ad una variabile e visualizzarlo come messaggio a scorrimento sulla matrice LED. Per farlo è necessario invocare la funzione `sense.show_message` passando come parametro il messaggio. Parametri opzionali sono `scroll_speed`, la velocità del testo espressa in numeri decimali, `text_colour` e `back_colour`, rispettivamente il colore del testo e dello sfondo. Per usarli è necessario definire con una tupla (rosso, verde, blu) la variabile colore, per poi passargliela.

```
1  #from sense_hat import SenseHat
2  #sense = SenseHat()
3
4  # Definisco il colore blu
5  blue = (0, 0, 100)
6  #Definisco il colore rosso
7  red = (100, 0, 0)
8
9  while True:
10     message = "Nome Cognome"
11     # Mostro il messaggio a scorrimento
12     sense.show_message(message, scroll_speed=0.05, text_colour=red,
           ↪ back_colour=blue)
```

3.4.3 Stampa su matrice LED pressione, temperatura e umidità

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse

- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di arrotondamento
  - Concetto di invocazione di metodo
  - Concetto di tupla
  - Concetto di ciclo infinito
- Competenze acquisite:
  - Concetto di funzione
  - Concetto di invocazione di funzione
  - Concetto di passaggio di parametri ad una funzione
  - Concetto di struttura di controllo If-then-else
  - Concetto di tipo di dato e casting
  - Concetto di concatenazione di stringhe
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questo esercizio si richiede agli studenti di scrivere un programma che dia in output sulla matrice di led le tre grandezze atmosferiche rilevabili dal sense hat, colorando in modo diverso lo sfondo a seconda del valore della temperatura. Come detto in precedenza, il valore della temperatura è impreciso, quindi viene implementata una funzione per correggerlo tenendo conto della temperatura del processore; questa viene già fornita agli studenti, che ricevono il codice presente nel file seguente fino alle riga 14. Verranno anche mostrati dal docente le API del Sense Hat necessarie per

lo svolgimento dell'esercizio, la sintassi del construtto `if`, gli operatori di casting e di concatenazione stringhe.

Consegna: Scrivere un programma che stampi all'infinito temperatura, pressione ed umidità, con il background rosso se la temperatura è superiore ai 26.7 gradi, verde altrimenti. Per leggere i valori, è necessario invocare il metodo `sense.get_temperature()` o corrispondente per le altri grandezze. Dopo aver letto la temperatura, questa va calibrata invocando la `calibrateTemperature` con il valore appena letto. I valori vanno arrotondati con `round(variabile, numerocifre)` alla prima cifra decimale. Per creare una stringa unica con testo e valori numerici si fa: `stringa = "testo" + str(variabile)`.

```
1  #from sense_hat import SenseHat
2  #import os
3  #sense = SenseHat()
4
5  #def calibrateTemperature(temperatura):
6      #Rilevo la temperatura del processore
7   #   tmp = os.popen('/opt/vc/bin/vcgencmd measure_temp')
8   #   cputemp = tmp.read()
9   #   cputemp = cputemp.replace('temp=','')
10  #   cputemp = cputemp.replace('\'C\n','')
11  #   cputemp = float(cputemp)
12     #correggo la temperatura rilevata
13  #   temperatura = temperatura - (cputemp - temperatura)/1.5
14  #   return temperatura
15
16 # Definisco i colori rosso e verde
17 red = (255, 0, 0)
18 green = (0, 255, 0)
19
20 while True:
```

```
21
22        # Effettuo la lettura delle grandezze da tutti e tre i
              ↪ sensori
23      t = sense.get_temperature()
24      p = sense.get_pressure()
25      h = sense.get_humidity()
26      #Effettuo la calibrazione della temperatura
27      t = calibrateTemperature(t)
28       # Arrotondo i valori alla prima cifra decimale
29      t = round(t, 1)
30      p = round(p, 1)
31      h = round(h, 1)
32      # Creo il messaggio
33      # str() converte i valori in stringhe in modo che possano
            ↪ essere concatenati
34      message = "Temperature: " + str(t) + " Pressure: " + str(p) + "
            ↪  Humidity: " + str(h)
35
36      if t > 18.3 and t < 26.7:
37          bg = green
38      else:
39          bg = red
40      # Mostro il messaggio a scorrimento
41      sense.show_message(message, scroll_speed=0.05, back_colour=bg)
42
43      sense.show_message(message, scroll_speed=0.05, back_colour=bg)
```

3.4.4 Salva su file csv pressione, temperatura e umidità e produci un grafico

- Materiali richiesti:

  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse

- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di arrotondamento
  - Concetto di invocazione di metodo
  - Concetto di ciclo infinito
  - Concetto di passaggio di parametri ad una funzione
  - Concetto di tipo di dato e casting
  - Concetto di concatenazione di stringhe

- Competenze acquisite:
  - Concetto di file csv
  - Formattazione di timestamp
  - Scrittura su file
  - Concetto di funzione di libreria
  - Concetto di invocazione di funzione di libreria

- Collegamenti interdisciplinari:
  - Matematica: Concetto di Diagramma Cartesiano
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questo esercizio agli studenti è richiesto di salvare il rilevazioni delle tre grandezze su un file csv, che poi verrà utilizzato per creare un diagramma cartesiano. Gli studenti possono partire dal codice dell'esercizio precendente, dato che le modifiche necessarie sono poche: ottenere un timestamp e fare output su un file csv. Si può spiegare agli studenti come fare il primo e lasciare che sperimentino variazioni dell'ordine e della formattazione degli elementi. Per il secondo si può fare un semplice esempio e contare sulla similitudine con lo `show_message` dell'esercizio precedente.

Consegna: Scrivere un programma che legge le tre grandezze temperatura, pressione ed umidità e le scrive, precedute da data ed ora, separandole con delle virgole, una serie di rilevazioni per riga, in un file csv. Per creare un file csv aperto in scrittura, bisogna usare `with open('nomefile.csv','w') as file:`. Per salvarsi data ed ora, si usa il comando `time.strftime("%Y-%m-%d %H:%M:%S", time.gmtime())`. Per addormentare il programma per un intervallo di tempo, si usa `time.sleep(1)`. Infine per scrivere sul file, una volta creata una stringa, si usa il comando `file.write(line)`.

```
1  #import time
2  #from sense_hat import SenseHat
3  #sense = SenseHat()
4
5  with open('rilevazioni.csv','w') as file:
6      while True:
7
8            # Effettua le rilevazioni con tutti e tre i sensori
9          t = sense.get_temperature()
10         p = sense.get_pressure()
11         h = sense.get_humidity()
12         # Arrotonda i valori alla prima cifra decimale
13         t = round(t, 1)
14         p = round(p, 1)
15         h = round(h, 1)
```

```
16
17          #Ottengo un timestamp
18        date = time.strftime("%Y-%m-%d %H:%M:%S", time.gmtime())
19
20    # str() converte i valori numerici in stringhe, in modo che
      ↪ possano essere concatenati
21        line =  date + ", "+str(t)+", "+str(p)+", "+str(h)
22        file.write(line)
23        file.write('\n')
24        time.sleep(1)
```

Questo è invece il codice fornito agli studenti per produrre il grafico corrispondente (un diagramma cartesiano) alle rilevazioni delle grandezze nell'arco di tempo selezionato. Per produrre i tre grafici (uno per ogni grandezza) è sufficiente modificare la riga 7, dal momento che la temperatura occupa in ogni riga (quindi nel vettore) la seconda casella, mentre pressione ed umidità rispettivamente la terza e la quarta.

```
1  import matplotlib.pyplot as plt
2  from csv import reader
3
4  with open('rilevazioni.csv', 'r') as f:
5          data = list(reader(f))
6          temp = [i[1] for i in data[1::]]
7          time = [i[0] for i in data[1::]]
8
9          #plt.plot(time, temp)
10         xn = range(len(time))
11         plt.plot(xn, temp)
12         plt.tight_layout()
13         plt.xticks(xn, time)
14         plt.xticks(rotation=90)
15         plt.savefig('test.png')
```

3.4.5 Mostra una barra corrispondente al livello della grandezza selezionata

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di proporzione
  - Concetto di tupla
  - Concetto di funzione
- Competenze acquisite:
  - Concetto di vettore
  - Operatore booleano and
  - Concetto di programmazione ad eventi
- Collegamenti interdisciplinari:
  - Matematica: Concetto di matrice
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questo esercizio si richiede agli studenti di usare la matrice di LED per stampare, quando selezionata con il joystick la grandezza corrispondente, un livello che indichi la dimensione della grandezza. Ad esempio, se i limiti ragionevoli entro cui si può misurare una temperatura sono tra -15 e 45, allora una temperatura di 30 gradi accenderà metà dei led (ovvero metà delle righe), mentre una di 42 li accenederà tutti. I valori dei limiti inferiori e superiori saranno forniti agli studenti. Gli studenti dovranno realizzare innanzitutto le varie matrici da visualizzare, da quella completamente vuota a quella completamente piena. In realtà la funzione delle API del sense hat prende in input un vettore lungo 64, non una matrice 8x8). Quindi dovranno realizzare una funzione che a seconda del valore percentuale della grandezza, accenda il numero di righe corrispondenti. Dovranno poi realizzare una funzione per ogni grandezza, che converta il valore rilevato in un valore percentuale, per poi andare a passarlo alla funzione sopracitata. Infine, seguendo un esempio mostrato alla lavagna, dovranno far sì che le 3 funzioni realizzate vengano invocate al click del joystick in una certa direzione.

Consegna: Scrivere un programma il cui, alla pressione del joystick in una direzione, corrisponda la visualizzazione del livello di una grandezza, usando righe intere della matrice a LED. Se la grandezza va da 0 a 100, ad esempio, ed il valore rilevato è 50, allora andranno accesi metà dei led. Essendo la matrice composta da 8 righe, bisognerà convertire in qualche modo il valore dalla sua scala ad una scala in ottavi. Per accendere in un certo modo la matrice, bisogna usare la `sense.set_pixels`, passandole un vettore di 64 colori, uno per ogni led. Sarà quindi necessario definire staticamente 8 matrici, una per livello. Per far sì che venga eseguita una funzione alla pressione del joystick, bisognerà scrivere `sense.stick.direction_up = funzione`. All'inizio di ogni funzione, per evitare che venga eseguita sia alla pressione, che al rilascio del joystick, bisogna includere la riga `if event.action != ACTION_RELEASED:`.

```
#from sense_hat import SenseHat, ACTION_RELEASED
#sense = SenseHat()

#maxtemp = 45
#mintemp = -15
#fonte wikipedia, minpressure livello del mare con tornado
#maxpressure = 1085
#minpressure = 870
#non necessario, si puo'usare direttamente il valore
#maxhumid = 100
#minhumid = 0

#otto livelli di riempimento della matrice
r = (55, 0, 0)
w = (55, 55, 55)

zero = [
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w
        ]


uno = [
        w,w,w,w,w,w,w,w,
        w,w,w,w,w,w,w,w,
```

```
32          w,w,w,w,w,w,w,w,
33          w,w,w,w,w,w,w,w,
34          w,w,w,w,w,w,w,w,
35          w,w,w,w,w,w,w,w,
36          w,w,w,w,w,w,w,w,
37          r,r,r,r,r,r,r,r
38          ]
39
40  due = [
41          w,w,w,w,w,w,w,w,
42          w,w,w,w,w,w,w,w,
43          w,w,w,w,w,w,w,w,
44          w,w,w,w,w,w,w,w,
45          w,w,w,w,w,w,w,w,
46          w,w,w,w,w,w,w,w,
47          r,r,r,r,r,r,r,r,
48          r,r,r,r,r,r,r,r
49          ]
50
51  tre = [
52          w,w,w,w,w,w,w,w,
53          w,w,w,w,w,w,w,w,
54          w,w,w,w,w,w,w,w,
55          w,w,w,w,w,w,w,w,
56          w,w,w,w,w,w,w,w,
57          r,r,r,r,r,r,r,r,
58          r,r,r,r,r,r,r,r,
59          r,r,r,r,r,r,r,r
60          ]
61
62  quattro = [
```

```
63          w,w,w,w,w,w,w,w,
64          w,w,w,w,w,w,w,w,
65          w,w,w,w,w,w,w,w,
66          w,w,w,w,w,w,w,w,
67          r,r,r,r,r,r,r,r,
68          r,r,r,r,r,r,r,r,
69          r,r,r,r,r,r,r,r,
70          r,r,r,r,r,r,r,r
71          ]
72
73
74  cinque = [
75          w,w,w,w,w,w,w,w,
76          w,w,w,w,w,w,w,w,
77          w,w,w,w,w,w,w,w,
78          r,r,r,r,r,r,r,r,
79          r,r,r,r,r,r,r,r,
80          r,r,r,r,r,r,r,r,
81          r,r,r,r,r,r,r,r,
82          r,r,r,r,r,r,r,r
83          ]
84
85
86  sei = [
87          w,w,w,w,w,w,w,w,
88          w,w,w,w,w,w,w,w,
89          r,r,r,r,r,r,r,r,
90          r,r,r,r,r,r,r,r,
91          r,r,r,r,r,r,r,r,
92          r,r,r,r,r,r,r,r,
93          r,r,r,r,r,r,r,r,
```

```
94          r,r,r,r,r,r,r,r
95          ]
96
97  sette = [
98          w,w,w,w,w,w,w,w,
99          r,r,r,r,r,r,r,r,
100         r,r,r,r,r,r,r,r,
101         r,r,r,r,r,r,r,r,
102         r,r,r,r,r,r,r,r,
103         r,r,r,r,r,r,r,r,
104         r,r,r,r,r,r,r,r,
105         r,r,r,r,r,r,r,r
106         ]
107
108
109 otto = [
110         r,r,r,r,r,r,r,r,
111         r,r,r,r,r,r,r,r,
112         r,r,r,r,r,r,r,r,
113         r,r,r,r,r,r,r,r,
114         r,r,r,r,r,r,r,r,
115         r,r,r,r,r,r,r,r,
116         r,r,r,r,r,r,r,r,
117         r,r,r,r,r,r,r,r
118         ]
119
120 def printLevelOnMatrix(perc):
121     if perc == 0 :
122         sense.set_pixels(zero)
123     if perc > 0 and perc <= 12.5:
124         sense.set_pixels(uno)
```

```
125     if perc > 12.5 and perc <= 25:
126         sense.set_pixels(due)
127     if perc > 25 and perc <= 37.5:
128         sense.set_pixels(tre)
129     if perc > 37.5 and perc <= 50:
130         sense.set_pixels(quattro)
131     if perc > 50 and perc <= 62.5:
132         sense.set_pixels(cinque)
133     if perc > 62.5 and perc <= 75:
134         sense.set_pixels(sei)
135     if perc > 75 and perc <= 87.5:
136         sense.set_pixels(sette)
137     if perc > 87.5 and perc <= 100:
138         sense.set_pixels(otto)
139
140
141 #Definisci le funzioni pressione, temperatura e umidita'
142 def pressure(event):
143   if event.action != ACTION_RELEASED: #altrimenti viene eseguito
        ↪ anche quando il joystick e' rilasciato
144       sense.show_message("pressione")
145       p=sense.get_pressure()
146       print(p)
147       ampiezzaScala = maxpressure - minpressure
148       Pmin = p - minpressure #dovrebbe sempre essere positivo
149       print(Pmin)
150       Pperc = Pmin/ampiezzaScala*100
151       print(Pperc)
152       printLevelOnMatrix(Pperc)
153
154 def temperature(event):
```

```
155     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
            ↪ anche quando il joystick e' rilasciato
156       sense.show_message("temperatura")
157       t=sense.get_temperature()
158       print(t)
159       ampiezzaScala = maxtemp - mintemp
160       Tmin = t - mintemp #dovrebbe sempre essere positivo
161       print(Tmin)
162       Tperc = Tmin/ampiezzaScala*100
163       print(Tperc)
164       printLevelOnMatrix(Tperc)
165
166 def humidity(event):
167     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
            ↪ anche quando il joystick e' rilasciato
168         sense.show_message("umidita")
169         h=sense.get_humidity()
170         print(h)
171         ampiezzaScala = maxhumid - minhumid
172         Hmin = h - minhumid #dovrebbe sempre essere positivo
173         print(Hmin)
174         #superfluo perche' l'umidita e' gia' in percentuale
175         Hperc = Hmin/ampiezzaScala*100
176         print(Hperc)
177         printLevelOnMatrix(Hperc)
178
179 while True:
180    sense.stick.direction_up = pressure
181    sense.stick.direction_left = temperature
182    sense.stick.direction_right = humidity
183    sense.stick.direction_middle = sense.clear
```

3.4.6 Mostra una barra corrispondente al livello della grandezza selezionata con matrice dinamica

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di proporzione
  - Concetto di tupla
  - Concetto di funzione
  - Concetto di vettore
  - Operatore booleano and
- Competenze acquisite:
  - Concetto di matrice come struttura dati
  - Concetto di programmazione ad eventi
- Collegamenti interdisciplinari:
  - Matematica: Concetto di matrice
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questa variante dell'esercizio precendente, si richiede di generare la matrice (in realtà un vettore) da passare alla funzione che accende i LED in maniera dinamica, invece che definire tutte le possibili matrici staticamente. Essendo più complicato generare direttamente un vettore, verrà suggerito agli studenti di operare su di una matrice, per poi convertirla in un vettore mediante una apposita funzione (che in caso di difficoltà può essere fornita direttamente agli studenti). Occorrerà quindi una terza funzione in più, che generi la matrice, la converta in vettore e poi accenda i led. Questa verrà invocata nella `printLevelOnMatrix`, al posto della `set_pixels` dell'esercizio precedente.

Consegna: Partendo dal risultato dell'esercizio precendente, scrivere un programma che svolge un compito analogo, ma, invece che usare per la `sense.set_pixels` matrici generate manualmente, scrivere una funzione che dato il numero di righe piene che deve avere una matrice, genera la matrice corrispondente. Dato che in realtà `sense.set_pixels` lavora con vettori, ma usare le matrici è più comodo, vi è fornita una funzione che converte una matrice in vettore.

```
1  #from sense_hat import SenseHat, ACTION_RELEASED
2  #sense = SenseHat()
3
4  #maxtemp = 55
5  #mintemp = -15
6  #fonte wikipedia, minpressure livello del mare con tornado
7  #maxpressure = 1085
8  #minpressure = 870
9  #non necessario, si puo'usare direttamente il valore
10 #maxhumid = 100
11 #minhumid = 0
12
13 r = (55, 0, 0)
14 w = (55, 55, 55)
```

```
15
16 def generateMatrix(numeroBarre):
17     mat = []
18     #faccio le barre vuote
19     for i in range(0,8-numeroBarre):
20         line = []
21         for j in range(0,8):
22             line.append(w)
23         mat.append(line)
24     #faccio le barre piene
25     for i in range(8-numeroBarre+1,9):
26         line = []
27         for j in range(0,8):
28             line.append(r)
29         mat.append(line)
30     return mat
31
32 #def matrixToList(matrice):
33 #    lst = []
34 #    for i in range(0,8):
35 #        for j in range(0,8):
36 #            lst.append(matrice[i][j])
37 #    return lst
38
39 def setMatrixLed(numeroBarre):
40     mat = generateMatrix(numeroBarre)
41     lst = matrixToList(mat)
42     sense.set_pixels(lst)
43
44 def printLevelOnMatrix(perc):
45     if perc == 0 :
```

```
46          setMatrixLed(0)
47      if perc > 0 and perc <= 12.5:
48          setMatrixLed(1)
49      if perc > 12.5 and perc <= 25:
50          setMatrixLed(2)
51      if perc > 25 and perc <= 37.5:
52          setMatrixLed(3)
53      if perc > 37.5 and perc <= 50:
54          setMatrixLed(4)
55      if perc > 50 and perc <= 62.5:
56          setMatrixLed(5)
57      if perc > 62.5 and perc <= 75:
58          setMatrixLed(6)
59      if perc > 75 and perc <= 87.5:
60          setMatrixLed(7)
61      if perc > 87.5 and perc <= 100:
62          setMatrixLed(8)
63
64  #Definisci le funzioni pressione, temperatura e umidita'
65  def pressure(event):
66    if event.action != ACTION_RELEASED: #altrimenti viene eseguito
        ↪ anche quando il joystick e' rilasciato
67        sense.show_message("pressione")
68        p=sense.get_pressure()
69        print(p)
70        ampiezzaScala = maxpressure - minpressure
71        Pmin = p - minpressure #dovrebbe sempre essere positivo
72        print(Pmin)
73        Pperc = Pmin/ampiezzaScala*100
74        print(Pperc)
75        printLevelOnMatrix(Pperc)
```

```
76
77 def temperature(event):
78     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
       ↪ anche quando il joystick e' rilasciato
79       sense.show_message("temperatura")
80       t=sense.get_temperature()
81       print(t)
82       ampiezzaScala = maxtemp - mintemp
83       Tmin = t - mintemp #dovrebbe sempre essere positivo
84       print(Tmin)
85       Tperc = Tmin/ampiezzaScala*100
86       print(Tperc)
87       printLevelOnMatrix(Tperc)
88
89 def humidity(event):
90     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
       ↪ anche quando il joystick e' rilasciato
91         sense.show_message("umidita")
92         h=sense.get_humidity()
93         print(h)
94         ampiezzaScala = maxhumid - minhumid
95         Hmin = h - minhumid #dovrebbe sempre essere positivo
96         print(Hmin)
97         #superfluo perche' l'umidita e' gia' in percentuale
98         Hperc = Hmin/ampiezzaScala*100
99         print(Hperc)
100        printLevelOnMatrix(Hperc)
101
102 while True:
103    sense.stick.direction_up = pressure
104    sense.stick.direction_left = temperature
```

```
105    sense.stick.direction_right = humidity
106    sense.stick.direction_middle = sense.clear
```

3.4.7 Mostra tre barre corrispondenti al livello di pressione, temperatura e umidità

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di proporzione
  - Concetto di tupla
  - Concetto di funzione
  - Concetto di vettore
- Competenze acquisite:
  - Operatore booleano and
  - Concetto di matrice come struttura dati
  - Concetto di programmazione ad eventi
- Collegamenti interdisciplinari:
  - Matematica: Concetto di matrice
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità

– Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questa ulteriore variante dell'esercizio precedente è richiesto di usare la matrice di LED per visualizzare contemporaneamente, su tre colonne di LED, i livelli delle tre grandezze. Avendo completato l'esercizio con la matrice dinamica, lo svolgimento di questo è relativamente lineare. Invece di avere un'unica funzione che genera la matrice, se ne avranno tre, ognuna che va a sovrascrivere con colori diversi una diversa area della matrice di LED. Essendo differenti queste funzioni, non si potrà usare `setMatrixLed` che aggreghi la generazione della matrice, la conversione in vettore e l'invocazione dell'API del Sense Hat.

Consegna: Partendo dal risultato dell'esercizio precedente, realizzare un programma che mostri tre colonne di colori diversi sulla matrice LED, una per ogni diversa grandezza. In questo caso la pressione del joystick non farà cambiare grandezza, ma si limiterà ad aggiornare il livello della grandezza corrispondente.

```
1  #from sense_hat import SenseHat, ACTION_RELEASED
2  #sense = SenseHat()
3
4  #maxtemp = 55
5  #mintemp = -15
6  #fonte wikipedia, minpressure livello del mare con tornado
7  #maxpressure = 1085
8  #minpressure = 870
9  #non necessario, si puo'usare direttamente il valore
10 #maxhumid = 100
11 #minhumid = 0
12
13 #otto livelli di riempimento della matrice
14 r = (55, 0, 0)
15 g = (0, 55, 0)
```

```
16 b = (0, 0, 55)
17 w = (55, 55, 55)
18 mat = []
19
20 #creo la matrice con tutti i led spenti
21 for i in range(8):
22     line = []
23     for j in range(8):
24         line.append(w)
25     mat.append(line)
26
27
28 def generateMatrixP(numeroBarreP):
29     #faccio le barre vuote per la pressione
30     for i in range(0,8-numeroBarreP):
31         for j in range(0,2):
32             mat[i][j]=w
33     #faccio le barre piene per la pressione
34     for i in range(8-numeroBarreP+1,8):
35         for j in range(0,2):
36             mat[i][j]=r
37     return mat
38
39 def generateMatrixT(numeroBarreT):
40      #faccio le barre vuote per la temperatura
41     for i in range(0,8-numeroBarreT):
42         for j in range(2,5):
43             mat[i][j]=w
44     #faccio le barre piene per la temperatura
45     for i in range(8-numeroBarreT+1,8):
46         for j in range(2,5):
```

```
47              mat[i][j]=g
48      return mat
49
50  def generateMatrixU(numeroBarreU):
51       #faccio le barre vuote per l'umidita
52      for i in range(0,8-numeroBarreU):
53          for j in range(5,8):
54              mat[i][j]=w
55      #faccio le barre piene per l'umidita
56      for i in range(8-numeroBarreU+1,8):
57          for j in range(5,8):
58              mat[i][j]=b
59      return mat
60
61  def matrixToList(matrice):
62      lst = []
63      for i in range(0,8):
64          for j in range(0,8):
65              lst.append(matrice[i][j])
66      return lst
67
68  def percToOctave(perc):
69      if perc == 0:
70          return 0
71      if perc > 0 and perc <= 12.5:
72          return 1
73      if perc > 12.5 and perc <= 25:
74          return 2
75      if perc > 25 and perc <= 37.5:
76          return 3
77      if perc > 37.5 and perc <= 50:
```

```
78          return 4
79      if perc > 50 and perc <= 62.5:
80          return 5
81      if perc > 62.5 and perc <= 75:
82          return 6
83      if perc > 75 and perc <= 87.5:
84          return 7
85      if perc > 87.5 and perc <= 100:
86          return 8
87
88  #Definisci le funzioni pressione, temperatura e umidita'
89  def pressure(event):
90    if event.action != ACTION_RELEASED: #altrimenti viene eseguito
          ↪ anche quando il joystick e' rilasciato
91        sense.show_message("pressione", text_colour=r, scroll_speed
              ↪ =0.01)
92        p=sense.get_pressure()
93        print(p)
94        ampiezzaScala = maxpressure - minpressure
95        Pmin = p - minpressure #dovrebbe sempre essere positivo
96        print(Pmin)
97        Pperc = Pmin/ampiezzaScala*100
98        print(Pperc)
99        Poctave = percToOctave(Pperc)
100       mat = generateMatrixP(Poctave)
101       lst = matrixToList(mat)
102       sense.set_pixels(lst)
103
104 def temperature(event):
105     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
            ↪ anche quando il joystick e' rilasciato
```

```
106         sense.show_message("temperatura", text_colour=g, scroll_speed
                ↪ =0.01)
107         t=sense.get_temperature()
108         print(t)
109         ampiezzaScala = maxtemp - mintemp
110         Tmin = t - mintemp #dovrebbe sempre essere positivo
111         print(Tmin)
112         Tperc = Tmin/ampiezzaScala*100
113         print(Tperc)
114         Toctave = percToOctave(Tperc)
115         mat = generateMatrixT(Toctave)
116         lst = matrixToList(mat)
117         sense.set_pixels(lst)
118
119 def humidity(event):
120     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
            ↪ anche quando il joystick e' rilasciato
121         sense.show_message("umidita", text_colour=b, scroll_speed
                ↪ =0.01)
122         h=sense.get_humidity()
123         print(h)
124         ampiezzaScala = maxhumid - minhumid
125         Hmin = h - minhumid #dovrebbe sempre essere positivo
126         print(Hmin)
127         #superfluo perche' l'umidita e' gia' in percentuale
128         Hperc = Hmin/ampiezzaScala*100
129         print(Hperc)
130         Hoctave = percToOctave(Hperc)
131         mat = generateMatrixU(Hoctave)
132         lst = matrixToList(mat)
133         sense.set_pixels(lst)
```

```
134
135 while True:
136     sense.stick.direction_up = pressure
137     sense.stick.direction_left = temperature
138     sense.stick.direction_right = humidity
139     sense.stick.direction_middle = sense.clear
```

3.4.8 Mostra sulla matrice LED due cifre corrispondenti alla grandezza selezionata

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di tupla
  - Concetto di funzione
  - Concetto di vettore
- Competenze acquisite:
  - Concetto di bitmap
  - Concetto di font
  - Concetto di programmazione ad eventi
- Collegamenti interdisciplinari:
  - Matematica: Concetto di matrice

- Fisica: Concetti di pressione, temperatura, umidità
- Fisica: Unità di misura di pressione, temperatura, umidità
- Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questo esercizio viene richiesto agli studenti di visualizzare sulla matrice di LED due cifre che indichino il valore della grandezza desiderata, alla pressione del joystick nella dimensione corrispondente. Gli studenti dovranno quindi cimentarsi nel disegnare i font per ognuna delle cifre, quindi dovranno creare una funzione, di fatto il punto critico dell'esercizio, che, dato un numero, ricopia, su di una matrice con tutti i led spenti, la cifra delle decine sulla parte sinistra della matrice e la parte delle unità sulla parte destra. Questo è molto complesso da realizzare usando i vettori, come mostra il codice qua sotto, quindi se uno studente si trova in difficoltà, gli si può suggerire di fare come nell'esercizio precedente: agire su di una matrice vera e propria, per poi convertirla alla fine in un vettore. Questo intuitivamente non è fattibile per la pressione, che ha quattro cifre ma una possibile variante è richiedere di farlo anche per essa, tenendo magari per qualche secondo le cifre delle centinaia e delle migliaia, per poi passare alle decine e unità.

Consegna: Partendo dal programma precedente, scrivere un programma che visualizzi, con due cifre, sulla matrice LED il valore della grandezza selezionata col joystick. Sarà quindi necessario definire un vettore font per ogni cifra e poi incollare due cifre per fare una matrice, da visualizzare con la solita `set_pixels`.

```
1 #from sense_hat import SenseHat, ACTION_RELEASED
2 #sense = SenseHat()
3
4 r = (55, 0, 0)
5 w = (55, 55, 55)
6
7 #i vari font delle diverse cifre sulla matrice di led 3x8 (per
    ↪ lasciare spazio)
```

```
8  empty = [
9          w,w,w,w,w,w,w,w,
10         w,w,w,w,w,w,w,w,
11         w,w,w,w,w,w,w,w,
12         w,w,w,w,w,w,w,w,
13         w,w,w,w,w,w,w,w,
14         w,w,w,w,w,w,w,w,
15         w,w,w,w,w,w,w,w,
16         w,w,w,w,w,w,w,w
               ↪
               ↪ ]
17
18 zero = [
19         r,r,r,w,
20         r,w,r,w,
21         r,w,r,w,
22         r,w,r,w,
23         r,w,r,w,
24         r,w,r,w,
25         r,w,r,w,
26         r,r,r,w
27         ]
28
29
30 uno = [
31         r,r,w,w,
32         w,r,w,w,
33         w,r,w,w,
34         w,r,w,w,
35         w,r,w,w,
36         w,r,w,w,
```

```
37          w,r,w,w,
38          r,r,r,w
39          ]
40
41  due = [
42          r,r,r,w,
43          w,w,r,w,
44          w,w,r,w,
45          w,w,r,w,
46          r,r,r,w,
47          r,w,w,w,
48          r,w,w,w,
49          r,r,r,w
50          ]
51
52  tre = [
53          r,r,r,w,
54          w,w,r,w,
55          w,w,r,w,
56          r,r,r,w,
57          w,w,r,w,
58          w,w,r,w,
59          w,w,r,w,
60          r,r,r,w
61          ]
62
63  quattro = [
64          w,w,r,w,
65          w,r,w,w,
66          r,w,w,w,
67          r,w,r,w,
```

```
68          r,r,r,w,
69          w,w,r,w,
70          w,w,r,w,
71          w,w,r,w
72          ]
73
74
75  cinque = [
76          r,r,r,w,
77          r,w,w,w,
78          r,w,w,w,
79          r,r,r,w,
80          w,w,r,w,
81          w,w,r,w,
82          w,w,r,w,
83          r,r,r,w
84          ]
85
86
87  sei = [
88          r,r,r,w,
89          r,w,w,w,
90          r,w,w,w,
91          r,r,r,w,
92          r,w,r,w,
93          r,w,r,w,
94          r,w,r,w,
95          r,r,r,w
96          ]
97
98
```

```
 99 sette = [
100          r,r,r,w,
101          w,w,r,w,
102          w,w,r,w,
103          w,r,w,w,
104          r,w,w,w,
105          r,w,w,w,
106          r,w,w,w,
107          r,w,w,w
108          ]
109
110
111 otto = [
112          r,r,r,w,
113          r,w,r,w,
114          r,w,r,w,
115          r,r,r,w,
116          r,w,r,w,
117          r,w,r,w,
118          r,w,r,w,
119          r,r,r,w
120          ]
121
122 nove = [
123          r,r,r,w,
124          r,w,r,w,
125          r,w,r,w,
126          r,r,r,w,
127          w,w,r,w,
128          w,w,r,w,
129          w,w,r,w,
```

```
130         r,r,r,w
131         ]
132 
133 def displayDigits(decimal, unity):
134     output = empty
135     for i in range(0,8):
136         for j in range(0, 4):
137             if decimal == 0:
138                 output[i*8+j]= zero[i*4+j]
139             if decimal == 1:
140                 output[i*8+j]= uno[i*4+j]
141             if decimal == 2:
142                 output[i*8+j]= due[i*4+j]
143             if decimal == 3:
144                 output[i*8+j]= tre[i*4+j]
145             if decimal == 4:
146                 output[i*8+j]= quattro[i*4+j]
147             if decimal == 5:
148                 output[i*8+j]= cinque[i*4+j]
149             if decimal == 6:
150                 output[i*8+j]= sei[i*4+j]
151             if decimal == 7:
152                 output[i*8+j]= sette[i*4+j]
153             if decimal == 8:
154                 output[i*8+j]= otto[i*4+j]
155             if decimal == 9:
156                 output[i*8+j]= nove[i*4+j]
157     for i in range(0,8):
158         for j in range(4, 8):
159             if unity == 0:
160                 output[i*8+j]= zero[i*4+j-4]
```

```
161             if unity == 1:
162                 output[i*8+j]= uno[i*4+j-4]
163             if unity == 2:
164                 output[i*8+j]= due[i*4+j-4]
165             if unity == 3:
166                 output[i*8+j]= tre[i*4+j-4]
167             if unity == 4:
168                 output[i*8+j]= quattro[i*4+j-4]
169             if unity == 5:
170                 output[i*8+j]= cinque[i*4+j-4]
171             if unity == 6:
172                 output[i*8+j]= sei[i*4+j-4]
173             if unity == 7:
174                 output[i*8+j]= sette[i*4+j-4]
175             if unity == 8:
176                 output[i*8+j]= otto[i*4+j-4]
177             if unity == 9:
178                 output[i*8+j]= nove[i*4+j-4]
179
180     sense.set_pixels(output)
181
182 #Definisci le funzioni pressione, temperatura e umidita'
183 def pressure(event):
184   if event.action != ACTION_RELEASED: #altrimenti viene eseguito
        ↪ anche quando il joystick e' rilasciato
185       sense.show_message("pressione", scroll_speed=0.03)
186       p=sense.get_pressure()
187       print(p)
188       p = round(p)
189       sense.show_message(str(p), text_colour = r)
190
```

```
191 def temperature(event):
192     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
            ↪ anche quando il joystick e' rilasciato
193       sense.show_message("temperatura", scroll_speed=0.03)
194       t=sense.get_temperature()
195       t = round(t)
196       decimal = round((t-t%10)/10)
197       unity = round(t%10)
198       print(t)
199       displayDigits(decimal, unity)
200
201 def humidity(event):
202     if event.action != ACTION_RELEASED: #altrimenti viene eseguito
            ↪ anche quando il joystick e' rilasciato
203         sense.show_message("umidita", scroll_speed = 0.03)
204         h=sense.get_humidity()
205         h = round(h)
206         decimal = round((h-h%10)/10)
207         unity = round(h%10)
208         print(h)
209         displayDigits(decimal, unity)
210
211 while True:
212    sense.stick.direction_up = pressure
213    sense.stick.direction_left = temperature
214    sense.stick.direction_right = humidity
215    sense.stick.direction_middle = sense.clear
```

3.4.9 La bussola

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb
  - Sense Hat
  - Schermo, Tastiera e Mouse
- Competenze richieste:
  - Uso di tastiera e mouse
  - Concetto di tupla
  - Concetto di funzione
  - Concetto di vettore
  - Concetti di angolo e gradi
- Competenze acquisite:
  - Concetto di polling
- Collegamenti interdisciplinari:
  - Matematica: Concetto di matrice
  - Fisica: Concetto di magnetismo terrestre
  - Fisica: Funzionamento del magnetometro

Descrizione: In questo esercizio si richiede allo studente di realizzare una bussola usando i sensori del sense hat e la matrice di LED. Ci viene in aiuto una API del sense hat, che restituisce l'angolo in gradi a cui si trova il nord rispetto all'orientamento attuale del Sense Hat (e quindi del Raspberry). Quindi lo studente deve semplicemente fare una funzione che, a seconda dell'angolo passato, disegni sulla matrice LED una retta con un estremo blu ed uno rosso (come le punte dell'ago della bussola), avente il rosso che

punta all'angolo passato alla funzione, per poi andarla ad invocare in un ciclo infinito ogni tot secondi. Nell'esempio che segue si usano otto configurazioni diverse della matrice, quindi la bussola non è molto sensibile alle piccole variazioni, di conseguenza una variante può essere quella di arrivare ad un numero maggiore di configurazioni.

Descrizione: Scrivere un programma che utilizzi il magnetometro integrato nel Sense Hat, per realizzare una bussola. Si tratta di avere una retta sulla matrice LED che punti sempre a nord con l'estremità rossa ed a sud con quella blu. Il metodo per leggere il magnetometro è `sense.get_compass()`

```
#from sense_hat import SenseHat
#import time

sense = SenseHat()
def disegnaRetta(angolo):
    r = (255, 0, 0)
    b = (0, 0, 255)
    w = (0, 0, 0)
    if angolo < 45:
        output=[w,w,w,w,w,w,w,w,
                w,w,w,w,w,w,w,w,
                w,w,w,w,w,w,w,w,
                b,b,b,b,r,r,r,r,
                w,w,w,w,w,w,w,w,
                w,w,w,w,w,w,w,w,
                w,w,w,w,w,w,w,w,
                w,w,w,w,w,w,w,w
                ]
        sense.set_pixels(output)
    else:
        if angolo < 90:
             output=[w,w,w,w,w,w,w,r,
```

```
23                  w,w,w,w,w,w,r,w,
24                  w,w,w,w,w,r,w,w,
25                  w,w,w,w,r,w,w,w,
26                  w,w,w,b,w,w,w,w,
27                  w,w,b,w,w,w,w,w,
28                  w,b,w,w,w,w,w,w,
29                  b,w,w,w,w,w,w,w
30                  ]
31               sense.set_pixels(output)
32          else:
33              if angolo < 135:
34                  output=[w,w,w,r,w,w,w,w,
35                      w,w,w,r,w,w,w,w,
36                      w,w,w,r,w,w,w,w,
37                      w,w,w,r,w,w,w,w,
38                      w,w,w,b,w,w,w,w,
39                      w,w,w,b,w,w,w,w,
40                      w,w,w,b,w,w,w,w,
41                      w,w,w,b,w,w,w,w
42                      ]
43                  sense.set_pixels(output)
44              else:
45                  if angolo < 180:
46                      output=[r,w,w,w,w,w,w,w,
47                      w,r,w,w,w,w,w,w,
48                      w,w,r,w,w,w,w,w,
49                      w,w,w,r,w,w,w,w,
50                      w,w,w,w,b,w,w,w,
51                      w,w,w,w,w,b,w,w,
52                      w,w,w,w,w,w,b,w,
53                      w,w,w,w,w,w,w,b
```

```
54                      ]
55                      sense.set_pixels(output)
56                  else:
57                      if angolo < 225:
58                          output=[w,w,w,w,w,w,w,w,
59                                  w,w,w,w,w,w,w,w,
60                                  w,w,w,w,w,w,w,w,
61                                  r,r,r,r,b,b,b,b,
62                                  w,w,w,w,w,w,w,w,
63                                  w,w,w,w,w,w,w,w,
64                                  w,w,w,w,w,w,w,w,
65                                  w,w,w,w,w,w,w,w
66                                  ]
67                          sense.set_pixels(output)
68                      else:
69                          if angolo < 270:
70                              output=[w,w,w,w,w,w,w,b,
71                              w,w,w,w,w,w,b,w,
72                              w,w,w,w,w,b,w,w,
73                              w,w,w,w,b,w,w,w,
74                              w,w,w,r,w,w,w,w,
75                              w,w,r,w,w,w,w,w,
76                              w,r,w,w,w,w,w,w,
77                              r,w,w,w,w,w,w,w
78                              ]
79                              sense.set_pixels(output)
80                          else:
81                              if angolo < 315:
82                                  output=[w,w,w,b,w,w,w,w,
83                                  w,w,w,b,w,w,w,w,
84                                  w,w,w,b,w,w,w,w,
```

```
85                                  w,w,w,b,w,w,w,w,
86                                  w,w,w,r,w,w,w,w,
87                                  w,w,w,r,w,w,w,w,
88                                  w,w,w,r,w,w,w,w,
89                                  w,w,w,r,w,w,w,w
90                                  ]
91                                  sense.set_pixels(output)
92                              else:
93                                  if angolo < 360:
94                                      output=[b,w,w,w,w,w,w,w,
95                                              w,b,w,w,w,w,w,w,
96                                              w,w,b,w,w,w,w,w,
97                                              w,w,w,b,w,w,w,w,
98                                              w,w,w,w,r,w,w,w,
99                                              w,w,w,w,w,r,w,w,
100                                             w,w,w,w,w,w,r,w,
101                                             w,w,w,w,w,w,w,r
102                                             ]
103                                     sense.set_pixels(output)
104
105 while (1):
106     compass = sense.get_compass()
107     print ("Nord ", compass)
108     disegnaRetta(compass)
109     time.sleep(0.1)
```

3.4.10 Gioco della biglia

- Materiali richiesti:
  - Raspberry Pi (qualunque modello a 40 pin)
  - Alimentatore e Cavo MicroUsb

  - Sense Hat
  - Schermo, Tastiera e Mouse

- Competenze richieste:
  - Concetto di confronto
  - Uso di tastiera e mouse
- Competenze acquisite:
  - Concetto di numeri pseudocasuali
  - Concetto di operatori booleani
  - Concetto di dizionario
- Collegamenti interdisciplinari:
  - Fisica: Concetto di accelerazione di gravità
  - Fisica: Modalità di misurazione dell'accelerazione

Descrizione: Questo esercizio ripropone il gioco della biglia realizzato precedentemente in Scratch, ma da implementare questa volta in Python. Le competenze necessarie e le criticità sono grossomodo le stesse, a parte che qui la lettura dell'accelerometro viene fatta su di un oggetto dizionario, quindi bisogna spiegare agli studenti di cosa si tratta e come usarlo. La criticità maggiore rimane capire quali valori restituisce l'accelerometro a seconda delle inclinazioni sui vari assi e, quindi, quali valori usare come soglie per determinare il movimento. Se lo studente non ha svolto o non si ricorda l'esercizio omologo in Scratch, gli si suggerirà di procedere con alcune stampe dei valori mediante `print`.

Consegna: Realizzare un gioco in Python in cui viene acceso randomicamente un LED "buca", sulla matrice di LED del Sense Hat e bisogna portare un LED "biglia", inclinando il Raspberry Pi, dal centro della matrice a sopra il LED buca. Il sensore da usare è l'accelerometro. Per farlo è necessario leggere l'accelerometro, con la funzione `sense.get_accelerometer_raw()`, che

restituisce una struttura dati dizionario, con i valori sui singoli assi. Per ottenere un singolo valore, ad esempio la inclinazione sulla x, bisogna fare `x = accelerometro["x"]`.

```
#from sense_hat import SenseHat
#import random
#import time
#sense = SenseHat()

red = (255, 0, 0)
white = (255,255,255)

#il gioco continua all'infinito
while(True):
    x=4
    y=3
    # genero casualmente le coordinate del pixel corrispondente alla buca
    xtarget = random.randint(0,7)
    ytarget = random.randint(0,7)
    #prevengo che la buca venga generata sotto la pallina
    while (xtarget==x and ytarget==y):
        xtarget = random.randint(0,7)
        ytarget = random.randint(0,7)
        #concetto while: finche' non ho raggiunto l'obiettivo
    while (x!=xtarget or y!=ytarget):
        sense.clear()
        #setto il pixel corrispondente alla buca
        sense.set_pixel(xtarget, ytarget, red)
        # setto il pixel corrispondente alla biglia
        sense.set_pixel(x, y, white)
        o = sense.get_accelerometer_raw()
```

```
28          pitch = o["x"]
29          roll = o["y"]
30          if (pitch > 0.4 and x < 7):
31              x = x+1
32          if (pitch < -0.4 and x > 0):
33              x = x-1
34          if (roll > 0.4 and y < 7):
35              y = y+1
36          if (roll < -0.4 and y > 0):
37              y = y-1
38          time.sleep(0.1)
39      sense.show_message("Hai Vinto!")
```

Capitolo 4

Making alle Scuole Superiori

4.1 Raspberry Pi, Arduino ed i sensori usati

Per gli esercizi di making alle scuole superiori, ho deciso di proporre l'utilizzo di Raspberry Pi, il popolare ed economico single board computer, Arduino, il versatilissimo microcontrollore di progettazione Italiana, e di una serie di sensori selezionati sia perché insieme costituiscono una buona base per realizzare una stazione meteorologica pienamente funzionale, che può quindi essere il fine concreto di questa serie di esercizi, sia perché avendo diversi modi con cui ci si può interfacciare al sensore, è utile e formativo averci a che fare.

Inoltre, a differenza del Sense Hat, per utilizzare questo tipo di sensori è necessario imparare anche rudimenti di elettronica, di circuitistica e di saldatura (nonostante per iniziare suggerisca di usare breadboard, invece che saldare, questa attività può rendersi necessaria per l'uso di alcuni sensori).

In particolare si è optato per il sensore DS18B20 per la rilevazione della temperatura, il quale scrive il valore rilevato su di un file, perciò verrà suggerito allo studenti di implementare una classe per leggerlo in una maniera più ad alto livello; per il sensore DHT22 per l'umidità invece bisogna utilizzare una libreria ed accedere al secondo elemento della tupla ricevuta in output dall'invocazione della funzione per la lettura del sensore; per il sen-

sore BMP180 per la pressione bisogna innanzitutto saldare il sensore con i piedini, quindi per la lettura delle grandezze (include infatti anche un sensore di umidità e di temperatura, oltre a quello di pressione) bisogna importare una libreria, istanziare un oggetto ed utilizzare i rispettivi metodi; infine per un sensore YL69 per la rilevazione dell'umidità del suolo, questo, producendo un output di tipo analogico (invece che digitale come i precedenti sensori), non può essere utilizzato collegandolo direttamente al Raspberry Pi, ma è necessario collegarlo ad un Arduino, che poi trasmetta a sua volta via porta seriale al Raspberry Pi i valori rilevati.

Tutti questi sensori sono facilmente reperibili online, con prezzi inferiori ai 10 euro, quindi se una scuola volesse dotarsene per realizzare un laboratorio dedicato a questo (o una stazione meteorologica) la spesa sarebbe decisamente contenuta.

4.2 Esercizi in Python

Come per la sezione riguardante gli esercizi alle medie, per ogni esercizio verrà fornita la lista dei materiali richiesti, delle competenze richieste, delle competenze acquisite, dei collegamenti interdisciplinari, oltre ad una descrizione dell'esercizio ed alla consegna dell'esercizio. Gli estratti di codice forniti consistono nell'esito atteso degli esercizi, chiaramente altre soluzioni sono possibili. Le parti commentate, negli estratti di codice, sono quelle che vengono già fornite inizialmente agli studenti.

4.2.1 Accendi LED

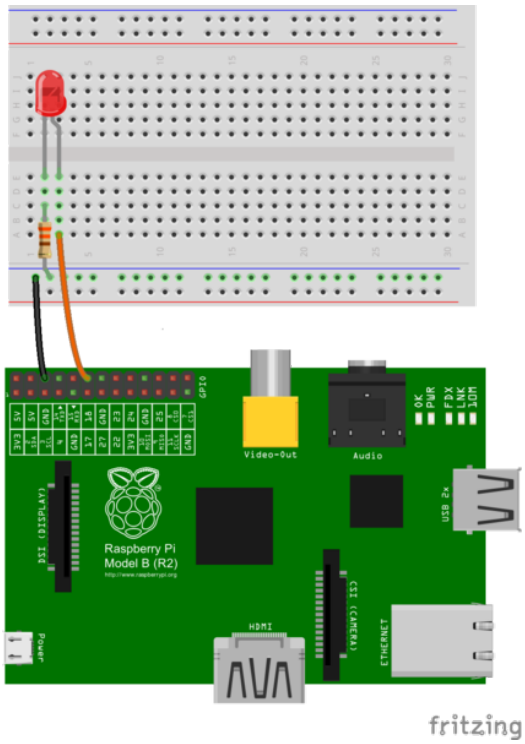

Figura 4.1: Lo schema elettrico dell'esercizio con il LED

- Materiali richiesti:
  - Raspberry Pi
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - LED
- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
- Competenze acquisite:
  - Concetto di libreria

  - Invocazione di funzione di libreria
  - Concetto di inizializzazione di variabile
  - Concetto di stampa in output

- Collegamenti interdisciplinari:
  - Fisica: funzionamento di un LED

Descrizione: In questo primo esercizio, gli studenti vengono guidati nella realizzazione di un programma che faccia accendere un LED per cinque secondi e poi lo spenga. Le librerie necessarie, con le rispettive funzioni da invocare, verranno spiegate alla lavagna. Verrà anche fatta una breve introduzione sulle modalità di utilizzo della breadboard. Gli studenti si limiteranno a realizzare il semplice schema elettrico sulla breadboard ed a mettere tutti i frammenti di codice insieme nell'ordine corretto.

Consegna: Scrivere in Python un programma che faccia accendere un LED per 5 secondi e poi lo spenga. E' necessario creare una variabile numerica corrispondente alla GPIO a cui si è collegato il LED (es. se è collegato alla GPIO 18, LED = 18), quindi lo si configura come output con `GPIO.setup(LED, GPIO.OUT)`, lo si accende con `GPIO.output(LED,True)` e equivalentemente lo si spegne.

```
1  # Importa le librerie necessarie
2  #import RPi.GPIO as GPIO
3  #import time
4
5  # Seleziona la modalita' di GPIO
6  #GPIO.setmode(GPIO.BCM)
7
8  #Disabilita i warnings
9  #GPIO.setwarnings(False)
10
11 # Inserisci il numero della GPIO a cui il LED e' collegato
```

```
12 LED = 18
13
14 # Seleziona il pin di GPIO del LED come un output
15 GPIO.setup(LED, GPIO.OUT)
16
17 print("accendo il LED")
18 # Accendi il pin di GPIO
19 GPIO.output(LED,True)
20
21 # Aspetta 5 secondi
22 time.sleep(5)
23
24 print("spengo il LED")
25 # Spegni il pin di GPIO
26 GPIO.output(LED,False)
```

4.2.2 Funzione per far lampeggiare il LED

- Materiali richiesti:
  - Raspberry Pi
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - LED
- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
- Competenze acquisite:

  - Concetto di libreria
  - Invocazione di funzione di libreria
  - Concetto di inizializzazione di variabile
  - Concetto di stampa in output
  - Concetto di ciclo limitato
  - Concetto di funzione
  - Concetto di input
  - Concetto di casting
- Collegamenti interdisciplinari:
  - Fisica: funzionamento di un LED

Descrizione: In questo esercizio si richiede di superare l'esercizio precedente, pur mantenendo lo stesso schema elettrico, realizzando una funzione che mantenga acceso per un dato intervallo di tempo, per un certo numero di iterazioni, il LED sullo specifico pin, questi tre parametri le saranno passati in input, dopo che sono stati richiesti all'utente dal terminale, con la semplice funzione `input`. Le operazioni elementari sono le medesime, ma è necessario che il docente fornisca spiegazioni sul concetto di ciclo e sul concetto di funzione, fondamentali per lo svolgimento dell'esercizio.

Consegna: Scrivere in Python una funzione che prenda in input il pin sul quale è collegato il LED, il numero di volte in cui bisogna farlo lampeggiare e la durata di ogni lampeggio in secondi, e che faccia lampeggiare il corrispondente LED per il numero corrispondente di volte per tot secondi. Per prendere in input i valori, bisogna usare la funzione `input("Messaggio")`, che restituisce il valore digitato dall'utente.

```
1 #import RPi.GPIO as GPIO ## Importo la libreria GPIO
2 #import time ## Importo la libreria time, permette l'uso di 'sleep'
3
4 #Usa la numerazione della board per i pin
```

```
5  #GPIO.setmode(GPIO.BCM)
6
7  #Definisce una funzione chiamata Lampeggio
8  def Lampeggio(pin,numVolte,intervallo):
9      GPIO.setup(pin, GPIO.OUT)
10     #Esegue il ciclo numVolte volte
11     for i in range(0,numVolte):
12         print("Iterazione " + str(i+1))
13         #Accende il pin desiderato
14         GPIO.output(pin,True)
15         #Attesa di intervallo secondi
16         time.sleep(intervallo)
17         #Spegne il pin desiderato
18         GPIO.output(pin,False)
19         #Attesa di intervallo secondi
20         time.sleep(intervallo)
21     print("Fatto")
22     GPIO.cleanup()
23
24 iterazioni = input("Inserisci il numero di lampeggi: ")
25 durata = input("Inserisci la durata in secondi di ogni lampeggio: "
       ↪ )
26 pin = input("inserisci il numero del pin al quale hai collegato il
       ↪ led: ")
27
28 Lampeggio(int(pin),int(iterazioni),float(durata))
```

4.2.3 Classe LED

- Materiali richiesti:
  - Raspberry Pi
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - LED
- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
- Competenze acquisite:
  - Concetto di libreria
  - Invocazione di funzione di libreria
  - Concetto di stampa in output
  - Concetto di programmazione orientata agli oggetti
  - Concetto di costruttore
  - Concetto di metodo
  - Concetto di campo
- Collegamenti interdisciplinari:
  - Fisica: funzionamento di un LED

Descrizione: Con questo esercizio si richiede un ulteriore passo avanti, chiedendo agli studenti di implementare una classe LED, che permetta di ottenere e settare la durata predefinita dell'intervallo di accensione, ottenere e settare il pin su cui si va ad operare, di accendere o spegnere il led, di lampeggiare per

la durata predefinita o per una durata passata in input. Tutte queste funzionalità sono realizzate mediante l'implementazione di metodi della classe. Si rende necessaria quindi un'introduzione teorica alla programmazione orientata agli oggetti, con particolare enfasi sui concetti di metodo, costruttore e campo. Lo schema elettrico rimane invece quello degli esercizi precedenti.

Consegna: Implementare una classe LED in Python, che abbia un costruttore che prende in input il pin su cui è montato il LED e la durata dell'intervallo di lampeggiamento. Inoltre la classe deve avere un metodo che stampi a terminale il pin attuale ed uno che stampi l'intervallo attuale e metodi che permettano di modificare i due campi. Infine, deve avere metodi che permettano di accendere il LED, spegnerlo, lampeggiare della durata predefinita di secondi e lampeggiare per un numero di secondi passati come parametro.

```
1  #import time
2  #import RPi.GPIO as GPIO
3  #class LED:
4      def __init__(self, x, y):
5          self.GPIO = x
6          self.intervallo = y
7
8      def dammiGPIOAttuale(self):
9          print(self.GPIO)
10
11     def dammiIntervalloAttuale(self):
12         print(self.intervallo)
13
14     def settaGPIO(self, x):
15         self.GPIO = x
16
17     def settaIntervallo(self, y):
18         self.intervallo = y
```

```
19
20      def accendi(self):
21          GPIO.setmode(GPIO.BCM)
22          GPIO.setwarnings(False)
23          GPIO.setup(self.GPIO, GPIO.OUT)
24          print("accendo il LED")
25          GPIO.output(self.GPIO,True)
26
27      def spegni(self):
28          GPIO.setmode(GPIO.BCM)
29          GPIO.setwarnings(False)
30          GPIO.setup(self.GPIO, GPIO.OUT)
31          print("spengo il LED")
32          GPIO.output(self.GPIO,False)
33
34      def lampeggia(self):
35          GPIO.setmode(GPIO.BCM)
36          GPIO.setwarnings(False)
37          GPIO.setup(self.GPIO, GPIO.OUT)
38          print("accendo il LED")
39          GPIO.output(self.GPIO,True)
40          time.sleep(self.intervallo)
41          print("spengo il LED")
42          GPIO.output(self.GPIO, False)
43
44      def lampeggiaPerSecondi(self, durata):
45          GPIO.setmode(GPIO.BCM)
46          GPIO.setwarnings(False)
47          GPIO.setup(self.GPIO, GPIO.OUT)
48          print("accendo il LED")
49          GPIO.output(self.GPIO,True)
```

```
50          time.sleep(durata)
51          print("spengo il LED")
52          GPIO.output(self.GPIO, False)
```

In questo file di prova invece si fa uso dei metodi della classe appena implementati, come dimostrazione di quanto sia più semplice gestire il LED ad alto livello.

```
1  from diodoLED import LED
2  lucetta = LED(18, 3)
3  lucetta.dammiGPIOAttuale()
4  lucetta.dammiIntervalloAttuale()
5  lucetta.settaIntervallo(10)
6  lucetta.lampeggia()
7  lucetta.lampeggiaPerSecondi(4)
```

4.2.4 Classe termometro

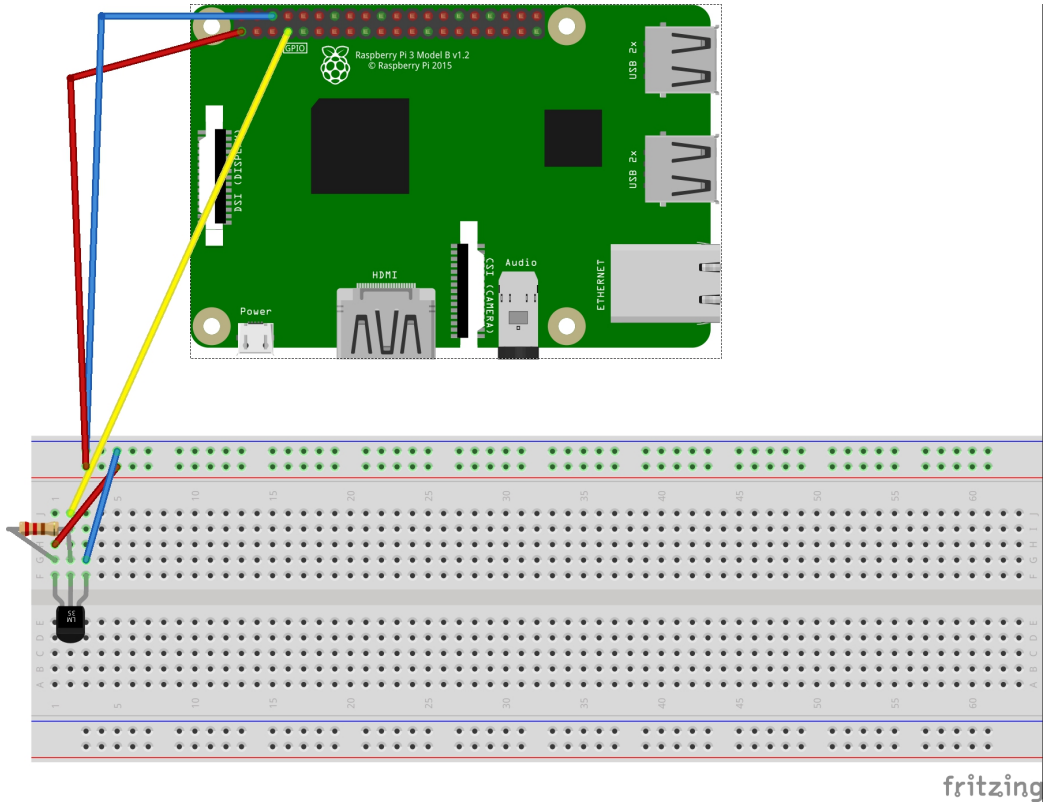

Figura 4.2: Lo schema elettrico dell'esercizio con il sensore DS18B20

- Materiali richiesti:
  - Raspberry Pi
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - Sensore DS18B20
- Competenze richieste:
  - Uso di tastiera e mouse

  - Conoscenze base su realizzazione di circuiti con breadboard

- Competenze acquisite:
  - Concetto di programmazione orientata agli oggetti
  - Concetto di metodo
  - Concetto di campo
  - Concetto di input da file
  - Concetto di casting

- Collegamenti interdisciplinari:
  - Fisica: Concetto di temperatura
  - Fisica: Unità di misura di temperatura
  - Fisica: Modalità di misurazione di temperatura

Descrizione: Dopo aver guidato gli studenti alla realizzazione del circuito ed aver configurato ogni Raspberry in modo da supportare il sensore, si mostra agli studenti che in un certo file, ad un dato path, si può leggere la misura di temperatura. Quindi l'obiettivo dell'esercizio sarà quello di realizzare una classe che implementi un termometro ad alto livello, che comprenda i metodi di ottenimento di minimo e massimo, aggiornamento del path a cui si trova il file da leggere e, chiaramente, di rilevazione della temperatura. Quest'ultima si basa proprio sulla lettura del file mostrato in precedenza, che deve essere aperto con una with, ne si deve leggere la seconda riga, ed estrarre gli ultimi caratteri (contenenti le cifre corrispondenti alla temperatura). Su queste deve essere effettuato poi il casting a `float`, quindi devono essere divise per 1000 per ottenere una temperatura in gradi Celsius.

Consegna: Creare una classe Termometro in Python, che abbia i campi path, maxtemp e mintemp. Deve avere metodi per restituire a la massima e la minima temperatura misurata fino a quel momento. Inoltre deve poter permettere di aggiornare il path del file in cui leggere il dato della temperatura. Infine deve esserci una metodo che restituisca la temperatura, dopo

averla letta dal file indicato in path, e che aggiorni il max ed il min. Per aprire un file usare `with open(file) as var`, per leggere una riga del file usare la funzione `var.readlines()`, che restituisce un vettore di righe (da cui quindi sarà necessario estrarre quella che ci interessa), per splittare un vettore di caratteri usare `var[indiceinizio:indicefine]`.

```
#class Termometro:
#  __path = '/sys/bus/w1/devices/28-03742e126461/w1_slave'
#  maxtemp = 0.0
#  mintemp = 100.0

  def dammiMax(self):
      return self.maxtemp

  def dammiMin(self):
      return self.mintemp

  def aggiornaPath(self, nuovoPath):
          self.__path = nuovoPath
#errore tipico: dimenticarsi di usare self (python non da errore,
    ↪ perche lo prende per inizializzazione variabile)
  def dammiTemperatura(self):
      with open(self.__path) as f:
          content = f.readlines()[1]
          temp = float(content[len(content)-6:len(content)-1])
          temp = temp /1000
          #print(temp)
          if (temp > self.maxtemp):
              self.maxtemp = temp
          if (temp < self.mintemp):
              self.mintemp = temp
          return temp
```

4.2.5 Stampa misura qualitativa pressione, temperatura e umidità

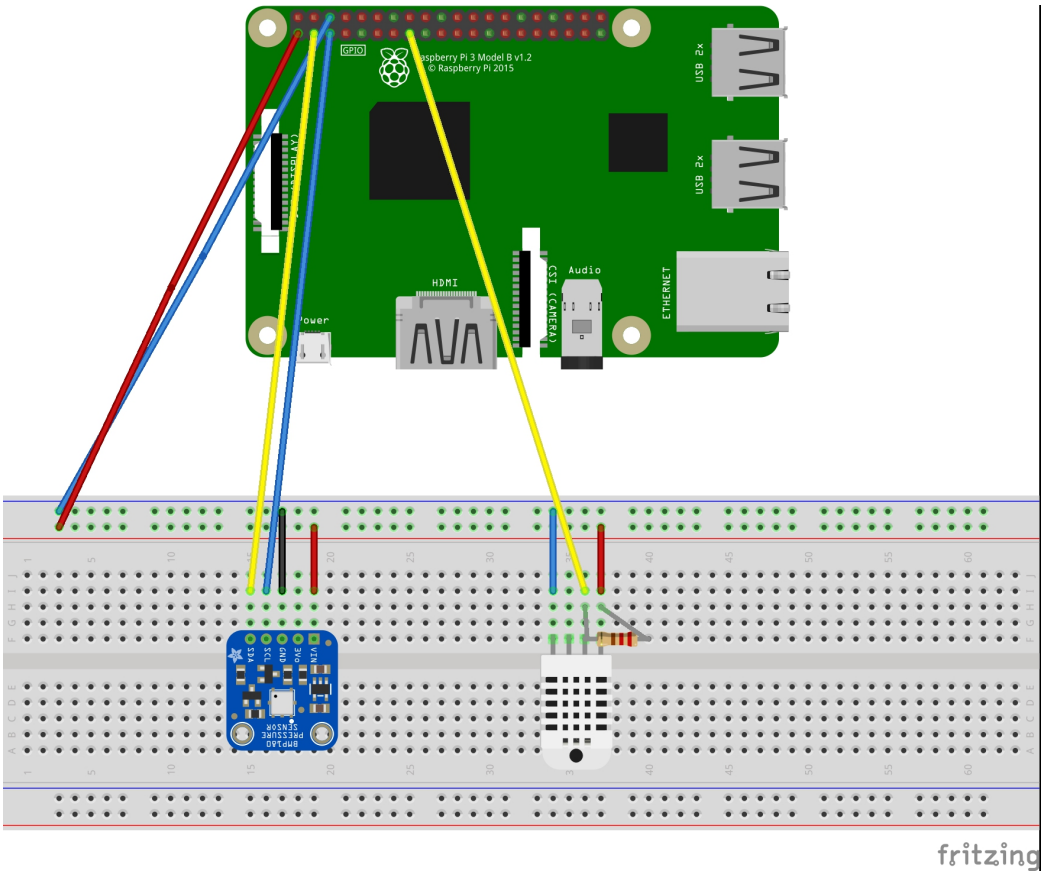

Figura 4.3: Lo schema elettrico dell'esercizio con i sensori DHT22 e BMP180

- Materiali richiesti:
  - Raspberry Pi
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - Sensore DHT22

  - Sensore BMP180

- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
- Competenze acquisite:
  - Concetto della struttura di controllo IF
  - Concetto di libreria
  - Invocazione di funzione di libreria
  - Concetto di programmazione orientata agli oggetti
  - Concetto di metodo
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: In questo esercizio si richiede agli studenti di realizzare un programma che dia informazioni qualitative sulle grandezze temperatura, pressione e umidità (del tipo "è caldo" o "è freddo"). La realizzazione di questo esercizio è sicuramente più complessa dal punto di vista della preparazione (specialmente sotto l'aspetto elettronico), mentre è abbastanza lineare dal punto di vista del codice. E' necessario infatti saldare il sensore BMP180 ai suoi piedini, inoltre lo schema elettrico è più complesso dei casi precedenti. Vanno poi installate e configurate le librerie necessarie per gestire i sensori. Una volta fornite agli studenti le sintassi delle chiamate alle funzioni (o ai metodi, nel caso del BMP180, che fa instanziare un oggetto) di libreria, gli studenti dovrebbero essere in grado di completare facilmente l'esercizio.

Consegna: Scrivere un programma in Python che stampi, a seconda dei valori rilevati di temperatura, pressione ed umidità, rilevati dai sensori DHT22 e BMP180, se è umido o secco, se fa caldo o freddo e se c'è alta o bassa pressione. Per leggere dal DHT22, effettuare una chiamata alla libreria del tipo `Adafruit_DHT.read_retry(NumGPIO,NumGPIO)`, che restituisce una tupla in cui il primo valore è l'umidità ed il secondo la temperatura. Per leggere dal BMP180 basta una invocazione al metodo `read_pressure` (o `read_temperature`) sull'oggetto istanziato.

```
1  #import Adafruit_DHT
2  #import Adafruit_BMP.BMP085 as BMP085
3
4  #sensor = BMP085.BMP085()
5
6  umidita = Adafruit_DHT.read_retry(22,22)[0]
7  temperatura1 = Adafruit_DHT.read_retry(22,22)[1]
8  temperatura2 = sensor.read_temperature()
9  pressione = sensor.read_pressure()
10 if umidita > 50:
11    print("e' umido")
12 else:
13    print("e' secco")
14 if temperatura1 > 25:
15    print("e' caldo")
16 else:
17    print("e' freddo")
18 if pressione > 1000:
19    print("c'e' alta pressione")
20 else:
21    print("c'e' bassa pressione")
```

4.2.6 Ordinamento timeseries

- Materiali richiesti:
  - Raspberry Pi
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - Sensore DHT22
  - Sensore BMP180
- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
  - Concetto di ciclo limitato e illimitato
  - Concetto di vettore
- Competenze acquisite:
  - Concetto della struttura di controllo IF
  - Concetto di ciclo annidato
  - Concetto di libreria
  - Invocazione di funzione di libreria
  - Concetto di programmazione orientata agli oggetti
  - Concetto di metodo
  - Algoritmi di sorting bubble sort e insert sort
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità
  - Fisica: Unità di misura di pressione, temperatura, umidità

– Fisica: Modalità di misurazione di pressione, temperatura, umidità

Descrizione: Questo esercizio si compone di due parti, sempre basandosi sullo stesso schema elettrico dell'esercizio precedente: un primo programma in cui si definisce una funzione che restituisce una tupla contenente le rilevazioni delle quattro grandezze (due misurazioni di temperatura, umidità e pressione)

Consegna - parte 1: Scrivere un programma in Python in cui si definiscono due funzioni: la prima che stampa ogni grandezza rilevata con la rispettiva unità di misura, la seconda che restituisce una tupla con le quattro rilevazioni effettuate alla chiamata della funzione.

```
1  #import sys
2  #import time
3
4  #import Adafruit_DHT
5  #import Adafruit_BMP.BMP085 as BMP085
6
7  #sensor = BMP085.BMP085()
8
9  def printDataForSeconds(sec):
10     for i in range(0,sec):
11         umidita = Adafruit_DHT.read_retry(22,22)[0]
12         temperatura1 = Adafruit_DHT.read_retry(22,22)[1]
13         temperatura2 = sensor.read_temperature()
14         pressione = sensor.read_pressure()
15         print('Temperatura1 = {0:0.01f} C*, Temperatura2 = {1:0.01f
               ↪ } C*, Pressione = {2:0.01f} Pa, Umidita = {3:0.01f}
               ↪ %'.format(temperatura1, temperatura2, pressione,
               ↪ umidita))
16
17 def returnDataAllSensors():
```

```
18          temperatura1 = Adafruit_DHT.read_retry(22,22)[1]
19          temperatura2 = sensor.read_temperature()
20          umidita = Adafruit_DHT.read_retry(22,22)[0]
21          pressione = sensor.read_pressure()
22          return (temperatura1, temperatura2, umidita, float(
                ↪ pressione)/100)
```

Descrizione: Ed un secondo programma in cui si utilizza la funzione precentemente definita per ottenere due vettori di rilevazioni della stessa grandezza (quindi due timeseries), che verranno poi ordinate in ordine crescente mediante due differenti algoritmi di sorting, bubble sort e insertion sort, la cui implementazione verrà lasciata agli studenti dopo averne mostrato alla lavagna una in pseudocodice. L'uso della libreria `numpy` è limitato alla conversione della lista di rilevazioni, in un vettore vero e proprio, in modo da poter procedere con gli algoritmi di ordinamento.

Consegna - parte 2: Trasformare lo pseudocodice fornito per l'insertion sort ed il bubble sort in codice Python. Quindi, usando la funzione definita nel precedente punto, creare una matrice che abbia per colonne una serie di rilevazioni per ogni grandezza. Infine, estrarre una colonna dalla matrice ed ordinarla con uno od entrambi gli algoritmi di sorting. Per creare una lista vuota usare `lista = []`, per farla diventare una lista di liste basta riempirlo di liste con `lista.append(riga)`, quindi convertirlo in matrice tramite `np.array(listadiliste)`. Per selezionare, ad esempio la prima colonna, usare `matrice[:,0]` e poi `.toList` per riconvertirlo in lista.

```
1  #from es5stampaLoopPTU import returnDataAllSensors
2  #import time
3  #import numpy as np
4
5  #Implementazione del bubble sort
6  def bubble_sort(items):
7      for i in range(len(items)):
8          for j in range(len(items)-1-i):
```

```
9             if items[j] > items[j+1]:
10                items[j], items[j+1] = items[j+1], items[j]     #
                      ↪ Swap!
11    return items
12
13 #Implementazione dell'insertion sort
14 def insertion_sort(items):
15    for i in range(1, len(items)):
16        j = i
17        while j > 0 and items[j] < items[j-1]:
18            items[j], items[j-1] = items[j-1], items[j]
19            j -= 1
20    return items
21
22 timeseries = []
23
24 for i in range(0, 10):
25    row = returnDataAllSensors()
26    timeseries.append(row)
27    time.sleep(1)
28
29 arrayTimeseries = np.array(timeseries)
30 #la temperatura e' la seconda colonna della matrice timeseries
31 temperature = arrayTimeseries[:,1].tolist()
32 print(temperature)
33 #l'umidita' e' la terza colonna della matrice timeseries
34 umidita = arrayTimeseries[:,2].tolist()
35 print(umidita)
36 print(bubble_sort(temperature))
37 print(insertion_sort(umidita))
```

4.2.7 Stampa umidità suolo da seriale con Arduino

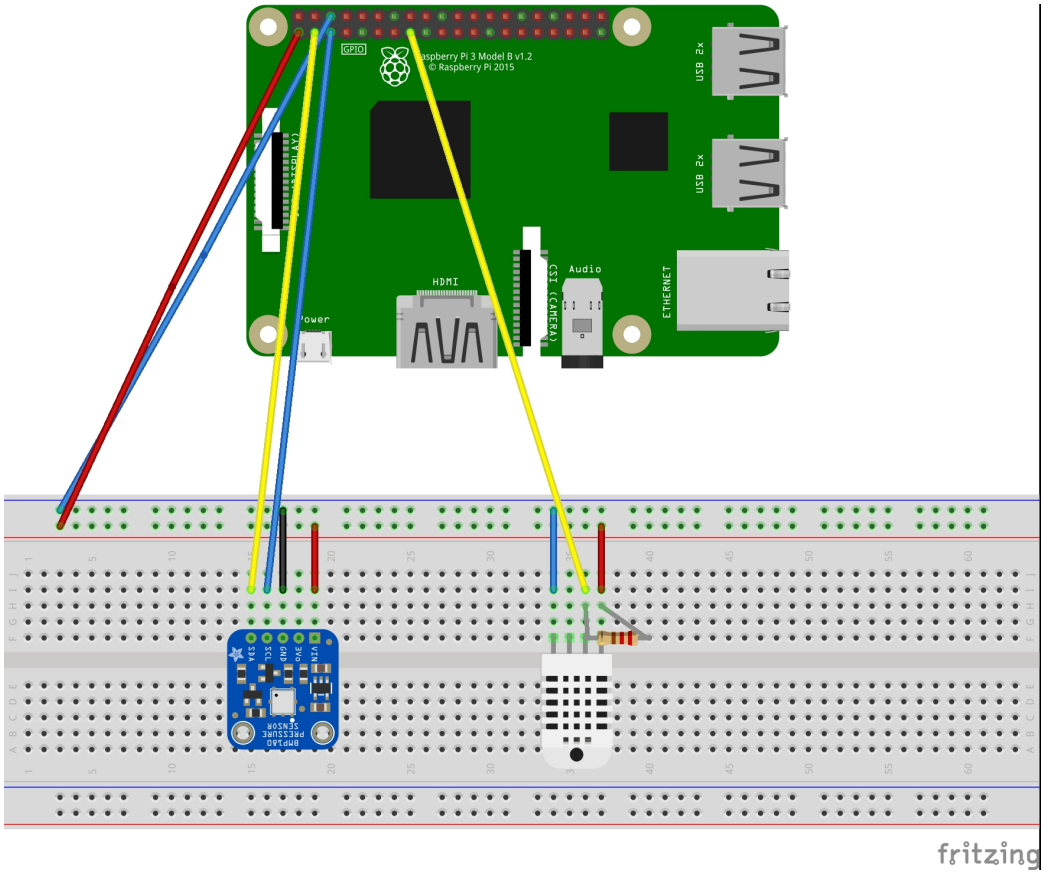

Figura 4.4: Lo schema elettrico dell'esercizio con il sensore YL69 e l'Arduino

- Materiali richiesti:
  - Raspberry Pi
  - Arduino Uno o Nano
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - Sensore YL69

- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
- Competenze acquisite:
  - Concetto di programmazione orientata agli oggetti
  - Concetto di metodo
  - Concetto di campo
  - Concetto di porta seriale
- Collegamenti interdisciplinari:
  - Fisica: Concetto di umidità del terreno
  - Fisica: Unità di misura dell'umidità del terreno
  - Fisica: Modalità di misurazione dell'umidità del terreno

Descrizione: Questo esercizio si compone di tre parti, la prima prevede la realizzazione di un programma per l'arduino. Ovvero di un file .ino (di fatto un programma in C) che prima inizializzi la seriale su 9600 baud e definisca il pin analogico A0 come pin di input, quindi definisca un loop in cui si legga il valore dal sopracitato pin, lo si stampi sulla seriale e si attenda mezzo secondo prima della iterazione successiva. La seconda parte invece consiste nel leggere la seriale dal Raspberry. Le uniche criticità consistono nello scegliere il path giusto della seriale e gli stessi Baud impostati nel programma dell'Arduino, come è necessario decodificare in Ascii i caratteri, che altrimenti mostrerebbero anche l'intestazione. Nella terza parte dell'esercizio è invece richiesto, come già fatto per gli esercizi precedenti, di realizzare una classe "UmidometroTerreno", che permetta di accedere, mediante metodi ad alto livello, ai valori di umidità percentuale rilevata dal sensore analogico connesso all'Arduino. Il metodo chiave, `dammiUmidita`, procede sulla falsariga di ciò che è stato fatto nella seconda parte di questo esercizio, con l'aggiunta che il

valore rilevato è convertito in intero, scalato in grandezza percentuale (invece che in valore da 0 a 1024, con 0 ad indicare la massima umidità e 1024 la minima). Il metodo inoltre si deve occupare di aggiornare i campi maxumid e minumid.

Consegna - parte 1: Scrivere un programma per Arduino che stampi sulla seriale il valore rilevato dal sensore di umidità del suolo. Per aprire la scrittura sulla seriale usare Serial.begin(numeroDiBaud). Per scegliere il pin analogico usare `pinMode(nomePin, INPUT)`, per effettuare una lettura su di esso usare invece `analogRead(nomePin)`. Per stampare sulla seriale usare `Serial.println(var)`. Infine può essere utile usare `delay(numMillisecondi)`, tra una lettura e l'altra.

```
1  void setup()
2  {
3    Serial.begin(9600);
4    pinMode(A0, INPUT);
5  }
6
7  void loop()
8  {
9    int s0 = analogRead(A0);
10   Serial.println(s0);
11   delay(500);
12 }
```

Consegna - parte 2: Scrivere un programma in Python che legga dalla seriale, decodifichi il messaggio in ASCII e lo stampi sul terminale. Per prepararsi alla lettura della seriale usare `serial.Serial('pathArduino',numBaud)`, solitamente il path è `/dev/ttyACM0`. Per leggere una riga usare il metodo readline, è necessario anche il metodo `decode('ascii')` per decodificare dall'ASCII.

```
1  import serial
```

```
2
3 ser = serial.Serial('/dev/ttyACM0',9600)
4 while True:
5         read_serial=ser.readline()
6         msg = read_serial.decode('ascii')
7         #rimuovo l'ultimo carattere, che e' un \n per evitare la
              ↪ doppia andata a capo
8         msg = msg[0:len(msg)-1]
9         print(msg)
```

Consegna - parte 3: Scrivere una classe UmidometroTerreno in Python, che abbia come campi il path dell'Arduino, i Baud, l'oggetto seriale, la minima e la massima umidità. Come metodi deve averne due che ritornano massima e minima umidità rilevata, uno per aggiornare il path ed uno che restituisca l'umidità in percentuale (e che aggiorni massimo e minimo). Bisogna tenere conto che il sensore restituisce 1024 per umidità al 0% e 0 per umidità al 100%.

```
1 #class UmidometroTerreno:
2 #  import serial
3 #  __path = '/dev/ttyACM0'
4 #  __baud = 9600
5 #  ser = serial.Serial(__path,__baud)
6 #  maxumid = 0.0
7 #  minumid = 100.0
8
9   def dammiMax(self):
10      return self.maxumid
11
12  def dammiMin(self):
13      return self.minumid
14
15  def aggiornaPath(self, nuovoPath):
```

```
16            self.__path = nuovoPath
17            self.ser = serial.Serial(__path,__baud)
18  #errore tipico: dimenticarsi di usare self (python non da errore,
        ↪ perche lo prende per inizializzazione variabile)
19    def dammiUmidita(self):
20          read_serial=self.ser.readline()
21          msg = read_serial.decode('ascii')
22          msg = msg[0:len(msg)-1]
23          umid = int(msg)
24          percumid = (1024-umid)/1024*100
25          if (percumid > self.maxumid):
26              self.maxumid = percumid
27          if (percumid < self.minumid):
28              self.minumid = percumid
29          return percumid
```

4.2.8 Stampa pressione, temperatura, umidità e umidità suolo

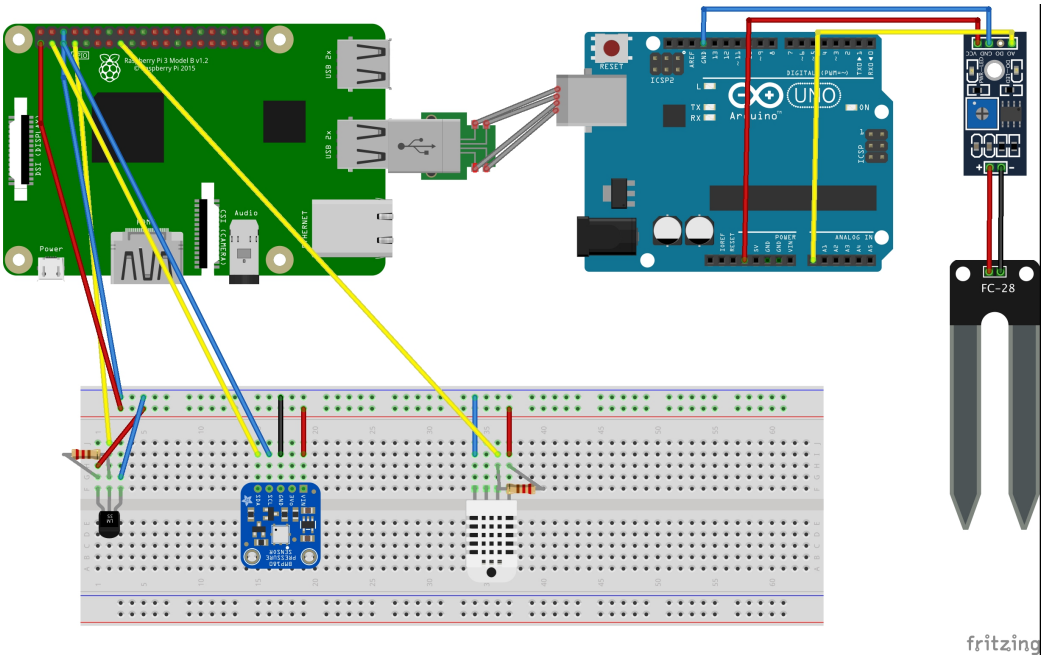

Figura 4.5: Lo schema elettrico dell'esercizio con tutti i sensori

- Materiali richiesti:
  - Raspberry Pi
  - Arduino Uno o Nano
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - Sensore DS18B20
  - Sensore DHT22
  - Sensore BMP180
  - Sensore YL69

- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
- Competenze acquisite:
  - Concetto di libreria
  - Invocazione di funzione di libreria
  - Concetto di programmazione orientata agli oggetti
  - Concetto di metodo
- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità, umidità del terreno
  - Fisica: Unità di misura di pressione, temperatura, umidità, umidità del terreno
  - Fisica: Modalità di misurazione di pressione, temperatura, umidità, umidità del terreno

Descrizione: Il compito di questo esercizio è stampare per un minuto ogni grandezza rilevata, dato che ci sono molteplici sensori di temperatura, si stamperanno molteplici rilevazioni della stessa grandezza. Un'aggiunta opzionale può essere quella di ottenere un valore ponderato facendo la media dei tre sensori. Una volta svolti gli esercizi precedenti, questo esercizio è relativamente semplice, perché sia dal punto di vista elettronico che da quello informatico, consiste semplicemente nel mettere tutto insieme.

Consegna: Scrivere un programma Python che metta insieme tutto quanto è stato prodotto negli esercizi precedenti, ovvero che stampi a terminale, per un minuto, tutte le grandezze che è possibile rilevare coi sensori e le relative unità di misura.

```
#import sys
#import time
#import Adafruit_DHT
#import Adafruit_BMP.BMP085 as BMP085
from es3termometroDS import Termometro
from es6umidterr import UmidometroTerreno

sensor = BMP085.BMP085()
term = Termometro()
umid = UmidometroTerreno()

for i in range(0,60):
        umidita = Adafruit_DHT.read_retry(22,22)[0]
        temperatura1 = Adafruit_DHT.read_retry(22,22)[1]
        temperatura2 = sensor.read_temperature()
        temperatura3 = term.dammiTemperatura()
        pressione = sensor.read_pressure()/100
        umidterreno = umid.dammiUmidita()
        print('Temperatura1 = {0:0.01f} C*, Temperatura2 = {1:0.01f
            ↪ } C*, Temperatura3 = {2:0.0001f} C*, Pressione =
            ↪ {3:0.01f} hPa, Umidita = {4:0.01f} %, Umidita
            ↪ Terreno = {5:0.01f} %'.format(temperatura1,
            ↪ temperatura2, temperatura3, pressione, umidita,
            ↪ umidterreno))
        time.sleep(1)
```

4.2.9 Salva rilevazioni grandezze su DB MySQL

- Materiali richiesti:
  - Raspberry Pi

  - Arduino Uno o Nano
  - Alimentatore e Cavo MicroUsb
  - Schermo, Tastiera e Mouse
  - Breadboard e ponticelli
  - Sensore DS18B20
  - Sensore DHT22
  - Sensore BMP180
  - Sensore YL69

- Competenze richieste:
  - Uso di tastiera e mouse
  - Conoscenze base su realizzazione di circuiti con breadboard
  - Concetto di database relazionale

- Competenze acquisite:
  - Concetto di libreria
  - Invocazione di funzione di libreria
  - Concetto di programmazione orientata agli oggetti
  - Concetto di metodo
  - Costrutto try
  - Costrutto with
  - Operazione insert su DB

- Collegamenti interdisciplinari:
  - Fisica: Concetti di pressione, temperatura, umidità, umidità del terreno
  - Fisica: Unità di misura di pressione, temperatura, umidità, umidità del terreno

– Fisica: Modalità di misurazione di pressione, temperatura, umidità, umidità del terreno

Descrizione: Sostanzialmente si tratta dello stesso esercizio precedente, con la fondamentale differenza che le rilevazioni vanno inserite in un database relazionale mysql, sulla quale gli studenti devono essere edotti prima dello svolgimento dell'esercizio. Prima di svolgere l'esercizio deve essere creato il database, il cui nome sarà WordpressDB, per motivi che saranno spiegati nella prossima sezione. Fa parte degli obiettivi dell'esercizio, creare la tabella, se non presente, usando un file .sql che verrà già fornito (e spiegato) agli studenti; dopo questa operazione il file dovrà essere rinominato in .bkp in modo da essere ancora accessibile, ma non lanciato in caso di invocazioni successive del programma. Per fare questo è caldamente consigliato utilizzare il costrutto try, mentre è da usare il construtto with, per verificare, prima di procedere, che tutto sia andato a buon fine nella connessione al database.

Consegna: Scrivere un programma in Python che inserisca le rilevazioni delle grandezze all'interno di un database mysql. E' necessario definire una variabile per l'username, la password ed il nome del DB. Nella funzione che salverà i dati sul DB, bisogna innanzitutto connettersi al DB con `mdb.connect("indirizzoIp", username, password, nomeDB)`, ottenere data ed ora da inserie come timestamp, e, dentro una with sulla connessione, invocare `cursore=connessione.cursor()` e `cursore.execute("query mysql")`. Inoltre bisogna controllare se la tabella esiste, con `try` creandone una nel caso negativo.

```
1 #import os
2 #import MySQLdb as mdb
3 #import datetime
4 #import Adafruit_DHT
5 #import Adafruit_BMP.BMP085 as BMP085
6 from es3termometroDS import Termometro
7 from es6umidterr import UmidometroTerreno
8
```

```
9  sensor = BMP085.BMP085()
10 term = Termometro()
11 umid = UmidometroTerreno()
12
13
14 databaseUsername="username" #L'username di mysql, solitamente root
15 databasePassword="password" #La password di mysql
16 databaseName="WordpressDB" #Il nome del database
17
18 def saveToDatabase(temperature, pressure, humidity, humidsoil):
19         con=mdb.connect("localhost", databaseUsername,
               ↪ databasePassword, databaseName)
20         currentDate=datetime.datetime.now().date()
21         hour = datetime.datetime.now().time()
22         with con:
23             cur=con.cursor()
24             cur.execute("INSERT INTO relevations (temperature,
                  ↪ pressure, humidity, humidsoil,dateMeasured,
                  ↪ hourMeasured) VALUES (%s,%s,%s,%s, %s, %s)",(
                  ↪ temperature,pressure, humidity, humidsoil,
                  ↪ currentDate, hour))
25             print("Rilevazioni salvate sul database")
26             return "true"
27
28
29 #controlla se la tabella gia' esiste o se dobbiamo crearne una
30 #try:
31         #queryFile=open("wordpress/createTable.sql","r")
32
33         #con=mdb.connect("localhost", databaseUsername,
               ↪ databasePassword,databaseName)
```

```
34          #currentDate=datetime.datetime.now().date()
35
36          #with con:
37              #line=queryFile.readline()
38              #query=""
39              #while(line!=""):
40                 # query+=line
41                #  line=queryFile.readline()
42
43              #cur=con.cursor()
44              #cur.execute(query)
45
46              #Rinomina il file, perche' non abbiamo bisogno di
                    ↪ ricreare la tabella ogni volta che lo script
                    ↪ viene eseguito
47              #queryFile.close()
48             # os.rename("wordpress/createTable.sql","wordpress/
                    ↪ createTable.sql.bkp")
49
50
51  #except IOError:
52     #pass #table has already been created
53
54
55  humidity = Adafruit_DHT.read_retry(22,22)[0]
56  temperature = term.dammiTemperatura()
57  pressure = sensor.read_pressure()/100
58  humidsoil = umid.dammiUmidita()
59  saveToDatabase(temperature, pressure, humidity, humidsoil)
```

4.3 Realizzazione del sito web

Fondamentale per la realizzazione di una stazione meteorologica pienamente funzionante è la realizzazione di un sito web in cui poter consultare i valori, per le quattro grandezze misurate, rilevati nelle precedenti ore. Inizialmente avevo pensato di far realizzare anche questa parte agli studenti, in modo da dargli nozioni anche di tecnologie web, ma nel corso della realizzazione di un proof of concept ho compreso che gran parte di questo processo è noioso, complicato e poco istruttivo.

Nella mia visione è quindi il docente a doversi occupare di questa parte.

Consiglio di seguire il valido tutorial per la realizzazione di un sito in wordpress con un database mysql (che verrà popolato, questo sì, dai programmi scritti dagli studenti nell'ultimo esercizio) che si può trovare al seguente indirizzo `https://www.raspberryweather.com/`.

Per visualizzare, come visibile in 4.6 ed in 4.7, le quattro grandezze è stato necessario mettere mano al plugin di wordpress realizzato dall'autore del tutorial.

Il codice risultante si può trovare, in licenza libera, oltre a tutto l'altro codice mostrato in e realizzato per questa tesi, al seguente indirizzo `https://github.com/akira002/weatherStation`.

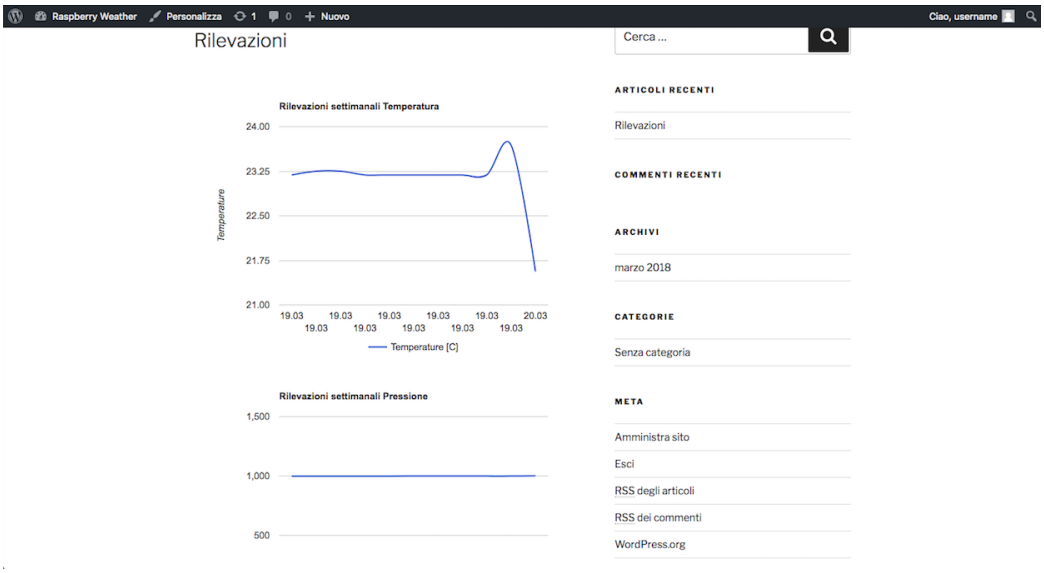

Figura 4.6: Uno screenshot del sito di visualizzazione delle rilevazioni

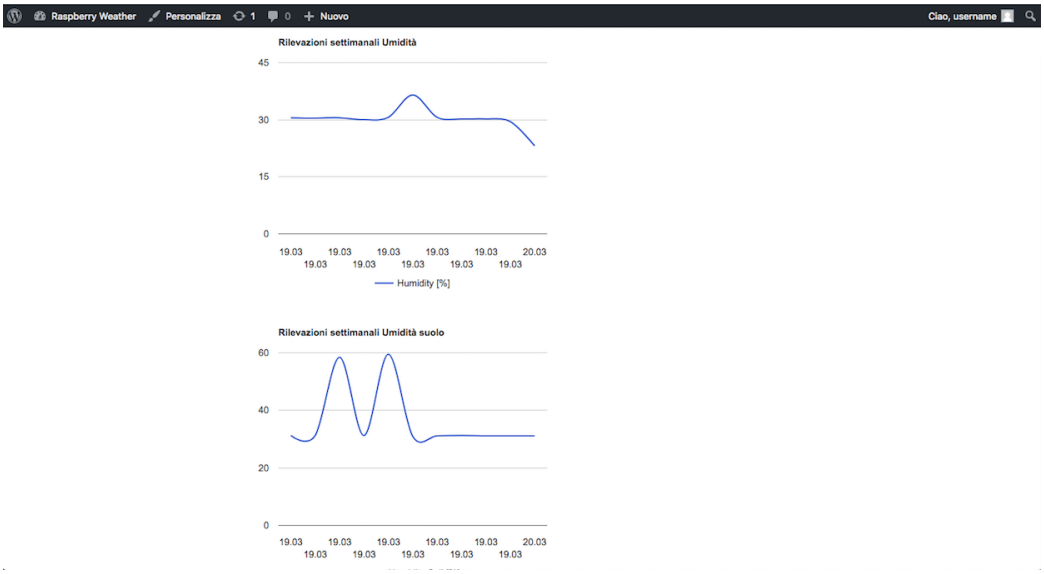

Figura 4.7: Un altro screenshot del sito di visualizzazione delle rilevazioni

4.4 Ultimi dettagli e messa in posa della stazione meteo

Una volta realizzati i circuiti, si può pensare di renderli definitivi, col vantaggio di una maggiore solidità e la possibilità di isolarli dall'umidità, praticando una stagnatura.

In rete è possibile reperire numerosi file cad da stampare in 3D raffiguranti case per il Raspberry Pi, fondamentali se si intende effettivamente usarlo all'esterno. Si può anche optare di acquistare direttamente un case (o outdoor enclosure) sui numerosi portali web che trattano accessori per il Raspberry Pi, oppure di adattare scatole di plastica (tipo tupperware o scatole elettriche) a questo scopo.

Oltre a questo è poi necessario passare un cavo di alimentazione dall'interno della scuola al punto in cui si deciderà di piazzare il dispositivo, anche se probabilmente non esistono cavi microUSB lunghi a sufficienza. Una soluzione può essere quindi usare una prolunga e collegarci l'alimentatore all'esterno. Diventa però necessario isolare anch'esso dalla pioggia.

Un ultimo accorgimento riguarda la connessione alla rete della scuola, necessaria per accedere via ssh al dispositivo e per mantenere online il sito delle rilevazioni. La via consigliata è quella di collegare un modulo wifi al Raspberry Pi (i modelli più moderni sono dotati di un modulo integrato).

Per quanto riguarda il posizionamento del dispositivo, bisogna compiere una scelta oculata. Sicuramente va piazzato ad altezza terra, altrimenti diventa impossibile posizionare correttamente il sensore di umidità del terreno. Altro accorgimento utile è non piazzarlo al riparo di una tettoia o di un albero, altrimenti i valori dell'umidità del terreno stessi potrebbero non essere veritieri. Infine bisognerà, al momento della lettura dei dati, tenere conto degli orari in cui il dispositivo è esposto direttamente alla luce del sole, in quanto questo sicuramente influenzerà i valori di temperatura.

Capitolo 5

Applicazione in classe

5.1 Il contesto

I primi di giugno del 2018 ho avuto modo di far sperimentare ad un ristretto numero di studenti una delle esperienze di laboratorio presentate nel mio lavoro di tesi.

Si è trattato di un laboratorio pomeridiano di due ore, tenuto presso il Malpighi Lab del Liceo Malpighi di Bologna. Gli studenti provenivano da due sezioni diverse di due classi seconde del liceo scientifico, opzione scienze applicate. Avendo due ore di informatica a settimana, quindi, avevano già maturato esperienza di programmazione. Nello specifico, gli studenti avevano avuto modo di lavorare sia in Scratch, durante il loro primo anno, sia in C, primo e unico linguaggio testuale da loro conosciuto.

L'esperienza che si è scelto di attuare è stata quella del gioco della biglia per il Sense Hat, da realizzare in Python [3.4.10]. Si è ritenuto che questo esercizio, seppur inizialmente classificato come adatto alle scuole medie, avesse l'equilibrio giusto tra nozioni da apprendere, difficoltà, interesse suscitato agli studenti e spazio per la personalizzazione, considerando il tempo a disposizione. Specialmente l'ultimo parametro ha pesato sulla decisone: si è scelto infatti un esercizio semplice ma coinvolgente, per avere tempo di applicare un approccio più costruzionista e quindi libero per gli studenti, ri-

spetto a prevedere esercizi con un solo approccio e quindi una sola soluzione possibile.

Ad ogni studente era stato fornito un computer portatile su cui lavorare individualmente e, dal momento che il numero dei partecipanti era superiore a quello dei Raspberry Pi equipaggiati di Sense Hat, era previsto che lavorassero sul simulatore di Sense Hat, `https://trinket.io/sense-hat`. Una volta svolto il grosso dell'esercizio, era previsto che lavorassero sugli ultimi ritocchi e la fase di testing vera e propria sui Raspberry fisici.

5.2 La struttura dell'esperienza

Essendo gli studenti completamente digiuni di Python, si è reso necessario preparare e svolgere una breve introduzione, se non ai concetti necessari per lo svolgimento dell'esercizio, da loro in gran parte già affrontati durante il percorso di studi, perlomeno alla sintassi di Python. Gli unici concetti che mancavano agli studenti erano quello di tupla e quello di lettura di un sensore.

Ho quindi preparato una breve serie di slide da presentare all'inzio dell'esperienza, che contenessero per ogni concetto un conciso esempio di applicazione di quel concetto, anche se applicato non nello stesso contesto in cui l'avrebbero dovuto usare per lo svolgimento dell'esercizio, per non facilitare troppo le cose.

Agli studenti è stata poi fornita una copia cartacea delle slide, in modo che potessero aiutarsi, durante la programmazione del gioco, usando pezzi di codice funzionanti con una funzione ben precisa. Nel pensare questo supporto agli studenti mi sono ispirato all'idea di Scratch di fornire ai discenti delle Scratch cards (`https://scratch.mit.edu/info/cards`), fogli con da un lato una funzionalità realizzata in Scratch e dall'altro la concatenazione di blocchi necessaria per attuarla.

Una volta terminata la parte di lezione frontale, mostrato agli studenti un esempio di gioco funzionante, chiedendo loro di realizzarne uno simile (ma con molte possibilità di personalizzazione e variazione) e distribuite le slide,

gli studenti sono stati lasciati liberi di cercare autonomamente il modo di comporre gli ingredienti forniti per ottenere il risultato desiderato. Questo non significa però che non fosse fornito nessuno scaffolding -ovvero una struttura di sostegno che li aiutasse nello svolgimento dei compiti- agli studenti: in caso di necessità era previsto l'intervento di mentor per aiutare e guidare gli studenti mediante l'uso di domande strategiche e consigli, oltre al sostegno dato dal già citato obiettivo iniziale assegnato e dalle slides o cards fornite.

5.3 Risultati dell'esperienza e feedback degli studenti

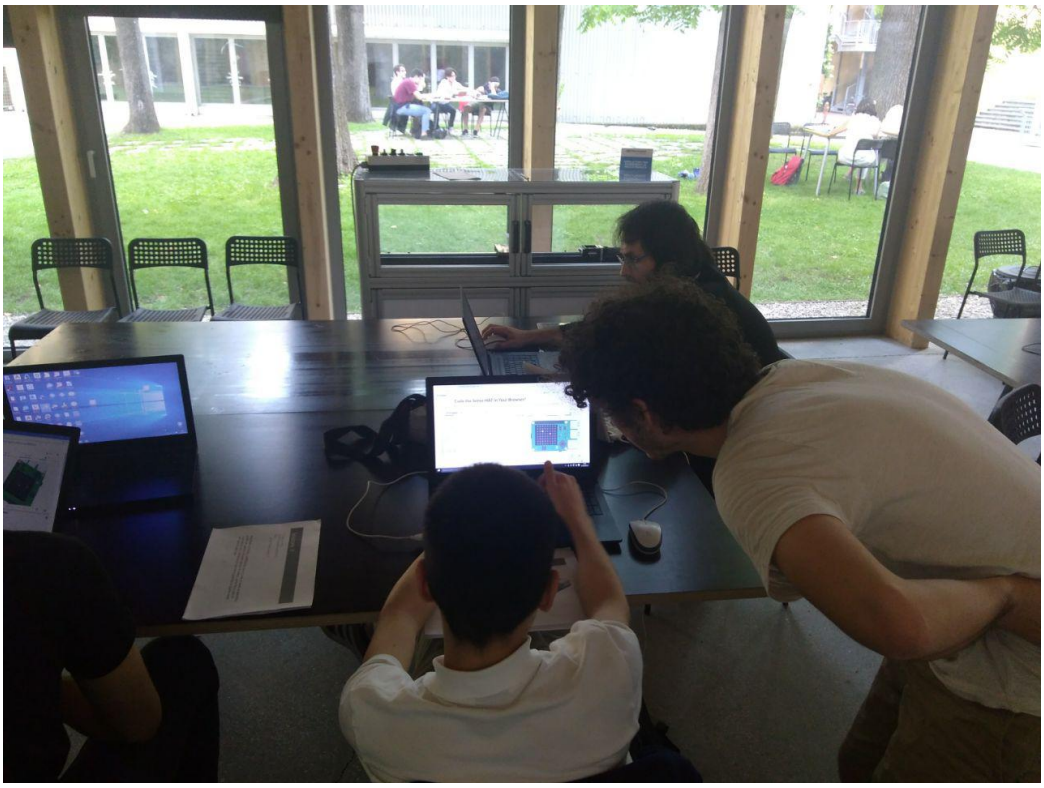

Figura 5.1: Uno studente in azione su Trinket

Già durante la parte teorica introduttoria, c'è stata molta partecipazione da parte degli studenti, con osservazioni intelligenti (hanno fatto notare co-

me nel generare casualmente la posizione di partenza della buca, si dovesse impedire di crearla esattamente al di sotto della pallina, che invece sarebbe partita da un punto fisso) ed apprezzamenti verso la semplicità e la potenza di Python. In particolare hanno apprezzato la leggerezza della sintassi e la facilità con cui è possibile generare numeri pseudocasuali, a differenza del C col metodo srand, da loro visto in classe pochi giorni prima dell'esperienza di laboratorio.

Quando però si è trattato di iniziare a scrivere il codice, si sono trovati tutti un po' spaesati ed è stato necessario più supporto da parte dei mentor, guidando gli studenti verso una realizzazione passo per passo del gioco, fornendo loro quindi obiettivi intermedi. Prima accendendo un led, poi accendendone uno in una posizione casuale, quindi muovendolo di un passo leggendo il sensore solo una volta, infine aggiungendo un ciclo while e mettendo quando scritto fino a quel punto in un ciclo terminante nel momento in cui le posizioni di biglia e buca sarebbero andate a coincidere.

Tutti gli studenti hanno avuto difficoltà nel comprendere quali istruzioni si sarebbero dovute iterare e quali no: modificavano infatti la posizione della biglia, senza poi richiamare la funzione che accendesse il led corrispondente.

Di contro, tutti sono riusciti autonomamente a comprendere come effettuare il movimento usando i dati rilevati dall'accelerometro. In questa occasione si è potuta osservare l'importanza di lavorare su dei device fisici, invece che compiere solo una programmazione astratta: i ragazzi infatti, per capire come si sarebbe dovuta muovere la biglia, si aiutavano inclinando i Raspberry equipaggiati di sense hat anche da spenti. Bastava loro appoggiarsi ad un device fisico per potersi costruire un modello mentale corretto.

Una problematica rilevata in classe è stata l'uso del simulatore `trinket.io`, che, essendo una web app, nel caso gli studenti lasciassero accidentalmente la pagina, causava la perdita totale del lavoro effettuato fino a quel punto. Questo ha spinto due studenti a lavorare in coppia, perché uno dei due avrebbe dovuto ricominciare da capo; l'approccio del pair programming è stato fruttuoso: hanno infatti finito per richiedere meno aiuto rispetto ai compagni

che lavoravano individualmente.

Quanto al mio lavoro, mi è stata fatta notare dagli altri mentor la mia tendenza ad imboccare troppo gli studenti, suggerendo la soluzione, invece che aiutarli a raggiungerla mediante domande mirate. Inoltre, anche a causa del tempo limitato, tendevo ad anticipare la presenza di un bug, invece di lasciare che gli studenti lo scoprissero autonomamente all'esecuzione.

In ogni caso l'aspetto positivo è stato che tutti gli studenti, pur con alcuni bug minori, sono riusciti a produrre, al termine delle due ore, un gioco funzionante, che abbiamo avuto modo di testare su Raspberry Pi fisici equipaggiati di Sense Hat. Sicuramente questo successo non sarebbe stato possibile, in un tempo così limitato, se i ragazzi non avessero già avuto esperienza con un linguaggio di programmazione testuale come C. Inoltre la parte più costruttivista dell'esperienza, in cui ogni studente personalizza e modifica a seconda dei suoi desideri il gioco proposto, non è stata attuata. Anche questo è da attribuire ai tempi stretti, infatti prima di personalizzare il gioco è necessario padroneggiarlo e prima di padroneggiarlo è necessario testarlo. Ma al termine delle due ore gli studenti avevano a malapena concluso la prima stesura del codice, testandolo solo una volta sui Raspberry fisici.

Al termine dell'esperienza di laboratorio ai ragazzi è stato sottoposto un sondaggio con le seguenti domande:

1. Quali sono i concetti che hai capito di più?

2. Quali sono i concetti che hai capito di meno?

3. Qual è (se c'è) la differenza tra come è stata svolta questa esperienza di laboratorio ed una normale lezione di informatica?

4. Hai capito meglio concetti di programmazione che già conoscevi grazie a questa attività? Se sì, perché?

Quanto alle prime due risposte, gli studenti sono molto netti, dicendo che tutti i concetti gli sono stati chiariti e che niente è rimasto oscuro. Questo è

probabilmente dovuto al loro essere già a conoscenza di gran parte dei concetti spiegati. Nella terza domanda c'è chi fa notare come maggior differenza il fatto di essere molto più seguiti rispetto ad una lezione normale, dato il numero inferiore di studenti. Altri hanno apprezzato la presenza di maggior pratica, rispetto ad una lezione di informatica curricolare. Hanno quindi implicitamente osservato ed apprezzato l'approccio maker e "hands-on" del laboratorio, auspicabilmente differente rispetto ad una lezione ordinaria. Questo nonostante il fatto che, tra le esperienze presentate in questa tesi, quelle con il Sense Hat prevedono meno making vero e proprio. Non ha quindi inciso negativamente, sotto questo aspetto, il fatto che gli studenti non abbiano potuto lavorare con un Sense Hat a testa, dato che sicuramente l'uso del simulatore non è comparabile a lavorare sul dispositivo fisico. Quanto all'ultima domanda, c'è chi ha risposto di aver capito meglio concetti di programmazione che aveva già visto, come il while, non fornendo però adeguate giustificazioni a questa affermazione.

Capitolo 6

Conclusioni

Lo studio della situazione italiana nell'ambito del making nelle scuole ha mostrato che c'è ancora molto lavoro da fare ed una proposta come quella presentata in questa tesi è originale. Come mostrato dall'esperienza tenuta in classe, sono presenti vantaggi, rilevati anche dagli studenti, di questo tipo di approccio che possiamo definire di making e physical computing, con lezioni molto "hands-on", al fine di migliorare l'insegnamento della programmazione a scuola. Anche se indubbiamente una sola sperimentazione di una sola lezione tra quelle proposte, su un numero ristretto di studenti, non può sicuramente essere considerata sufficiente per un giudizio globale sul tipo di approccio, indubbiamente può fornire spunti ed intuizioni utili. Ed il parere degli studenti è stato certamente positivo: anche se escludiamo il fatto che molti abbiano apprezzato di essere stati più seguiti, quasi a livello 1-1 individuale, cosa non ripetibile in una attuazione dell'esperienza durante lo svolgimento di una ordinaria lezione curricolare, c'è stato un grande apprezzamento dell'approccio "hands-on" o, per dirla con le parole degli stessi studenti, "più pratico". Anche la scelta del linguaggio di programmazione proposto, ovvero Python, è stata apprezzata dagli studenti, che si sono mostrati entusiasti della semplicità della sintassi e della potenza del linguaggio.

Oltretutto imparare a destreggiarsi con il Sense Hat e Python con le lezioni presentate in questo lavoro, può essere un primo step, per poi portare

gli studenti a partecipare al concorso di programmazione, promosso dall'ESA, Astro Pi; questo può essere utile, oltre che per il valore intrinseco della competizione, per stimolare ulteriormente gli studenti all'apprendimento dei concetti di programmazione.

Un fatto fondamentale da tenere in considerazione è la matrice parzialmente costruttivista degli esercizi e lezioni proposti: in molti casi non c'è un unico modo per raggiungere la soluzione, inoltre la stessa soluzione può essere variata o estesa con la fantasia degli studenti, come per esempio nel caso del già citato gioco della biglia, quando i ragazzi avevano proposto una biglia che rispuntasse dall'altro lato quando superato l'estremo opposto, usando parole loro "come in pacman". Dico "parzialmente" perché gli esercizi sulla stazione meteorologica per natura stessa dell'esperimento, si prestano meno a personalizzazioni: gli studenti sono però coinvolti perché vengono guidati nella realizzazione di un grande progetto, che poi useranno attivamente e quotidianamente una volta realizzato.

Una questione da tenere in conto è la necessità di un investimento iniziale da parte della scuola (o dell'ente) che decide di tenere lezioni basate su queste proposte. Sicuramente uno dei pregi del Raspberry Pi è il suo costo contenuto, oltrettutto può essere riutilizzato come computer vero e proprio per allestire un laboratorio di informatica. Altrettanto limitato nei costi, ma meno versatile è Arduino. Sense Hat richiede un investimento importante, considerando che si tratta di un "accessorio", però integra un gran numero di sensori, che permettono di inventare tantissime esperienze diverse, la sua matrice di LED RGB è molto accattivante per i ragazzi e, soprattutto, l'esperienza in classe ha dimostrato i limiti dall'approccio di simulazione, ovvero di usare il simulatore di Raspberry Pi e Sense Hat, Trinket. Per quanto riguarda le breadboard, i ponticelli, ed i vari sensori utilizzati per realizzare la stazione meteorologica, possono essere reperiti facilmente e ad un costo decisamente modico anche per piccole quantità.

E' molto delicata la questione della suddivisione delle lezioni tra scuole medie e scuole superiori. Inizialmente è stato scelto come spartiacque l'uso

o meno del Sense Hat, che, rendendo molto facile, grazie alle sue librerie, la lettura dei sensori, era giudicato appropriato anche per i più piccoli. Sicuramente infatti i sensori esterni, che necessitano per essere utilizzati di leggere valori scritti su file e che risultano comodi se incapsulati in classi, non si prestano agli studenti delle medie. Sicuramente l'uso della breadboard per connettere i sensori ai pin complica le cose, rispetto al semplicissimo modo di collegare il Sense Hat al Rapsberry Pi. Ma ciò non vuol dire che tutti gli esercizi che si possono pensare per il Sense Hat siano banali.

L'esperienza in classe ha mostrato che un gruppo di studenti delle superiori, con già un background solido di programmazione testuale, è riuscita a concludere l'esercizio del gioco della biglia (non l'esperienza con il Sense Hat più difficile) in 2 ore, venendo seguiti da mentor e con anche alcuni bug rimasti nel codice. Di conseguenza per presentare la stessa esperienza ad un gruppo di studenti delle medie è come minimo necessario molto più tempo, ma se questi sono digiuni di programmazione testuale, la cosa è difficilmente proponibile. Forse quindi la separazione più adeguata degli esercizi è la seguente: Scratch per le medie e Python per le superiori.

Come obiettivi futuri, auspicandomi un lavoro da insegnante, c'è sicuramente quello di portare queste lezioni in classe ed applicarle, proporre il lavoro ad altri docenti affinchè lo adottino, anche solo parzialmente e, facendo frutto della pratica in contesto scolastico, poter risuddividere ed ampliare gli esercizi, tra superiori e medie, in modo da calibrare nella maniera più corretta la difficoltà degli esercizi.

Elenco delle figure

Bibliografia

[1] Chris Anderson. *2012. Makers: The New Industrial Revolution. New York: Crown.*

[2] Dale Dougherty. *2012. The Maker Movement. Innovations, 7(3), 11–14.*

[3] Kimberly M. Sheridan Erica Rosenfeld Halverson. *2014. The Maker Movement in Education. Harvard Educational Review, 84(4), 495–504.*

[4] Mark Hatch. *2013. The Maker Movement Manifesto - Rules for Innovation in the New World of Crafters, Hackers, and Tinkerers. McGraw-Hill Professional, New York, NY.*

[5] Seymour Papert. *1986. Constructionism: a new opportunity for elementary science education. Proposal to the National ScienceFoundations. Cambridge, MA: MIT Media Laboratory.*

[6] Seymour Papert. *1993. Mindstorms – Children, Computers and Powerful Ideas. Second Edition. New York, USA. BasicBooks.*

[7] Heather Allen Pang Paulo Blikstein Sylvia Libow Martinez. *2015. Meaningful Making: Projects and Inspirations for FabLabs and Makerspaces. Constructing Modern Knowledge Press. Torrance, CA USA.*

[8] Ministero dell'istruzione dell'università e della ricerca. *Programma Operativo Nazionale - Per la scuola, competenze e ambienti per l'apprendimento.* URL: `http://www.istruzione.it/pon/ilpon.html`.

[9] Cynthia Solomon Seymour Papert. *1971. Twenty Things to Do with a Computer, Artificial Intelligence Memo 248, MIT Artificial Intelligence Laboratory, Cambridge, MA.*

[10] Gary Stager Sylvia Libow Martinez. *2013. Invent to Learn: Making, Tinkering and Engineering in the classroom. Torrance: Constructing Modern Knowledge Press. ISBN: 978-0-989-1511-0-8.*

Ringraziamenti

Desidero innanzitutto ringraziare il Professor Renzo Davoli, la cui passione per l'insegnamento e per il making mi ha spinto a dedicarmi a questo argomento di tesi. Senza aver assistito alle sue lezioni di informatica così uniche, probabilmente adesso non sarei qui.

In secondo luogo desidero ringraziare il mio correlatore, Michael Lodi, che con attenzione, competenza, sensibilità e disponibilità, mi ha dato una grandissima mano per completare questo lavoro di tesi.

Ringrazio quindi i miei genitori e la mia famiglia, che con grande lungimiranza mi hanno sempre sostenuto e lasciato sbagliare, quando avevo bisogno di farlo. Grazie perché avete la capacità unica e rara di lasciarmi i miei spazi quando serve o di fare 1000 chilometri per venire da me, quando ho bisogno di loro.

Non posso non ringraziare poi gli amici che mi hanno accompagnato, alcuni da tutta una vita, altri solo in questi ultimi tre anni. Uno con più esperienza di me diceva che la felicità non è reale se non è condivisa ed io sono davvero fortunato ad avere persone eccezionali con cui condividerla.

Infine desidero ringraziare Adele, compagna di vita e ragione di tutto. Durante questa magistrale abbiamo affrontato tante difficoltà, che abbiamo superato, e tanti giorni di sole, che siamo riusciti a goderci. Maturare e crescere con e grazie a te è un privilegio, questo momento è lo spartiacque che segna l'inizio della nostra vita adulta. Non vedo l'ora di iniziarla insieme a te.