Girls Who Code At Home

Ti posso aiutare?
Chatbot su Python

Panoramica delle attività

In questa attività esplorerai alcuni concetti informatici di base su Python e imparerai come aiutare la tua community creando un chatbot o un sistema di assistenza automatico! Avrai sicuramente già interagito con un chatbot in passato, magari senza nemmeno accorgertene. Alcuni tra i chatbot più diffusi sono: Siri di Apple, Alexa di Amazon, l'assistente di Google e Lyft; più recentemente, l'Organizzazione mondiale della sanità ne ha ideato uno centrato sugli Eventi legati alla COVID-19. Prima di cominciare con la progettazione e con la programmazione del tuo chatbot, dai un'occhiata alla Women in Tech Spotlight del momento, Erica Kochi. Nella sua veste di cofondatrice dell'Unità di innovazione dell'UNICEF, Erica ha creato un chatbot per aiutare a curare e ad assistere le persone in Nigeria e in Rwanda.

Materiali

- Trinket Editor
- Iniziare a programmare con Trinket
- Esempio di progetto per chatbot
- Facoltativo: Guida alla pianificazione
- Facoltativo: Penna/matita/pennarelli

Women in Tech Spotlight: Erica Kochi

Erica Kochi è cofondatrice e ha la direzione congiunta dell'Unità di innovazione dell'UNICEF, avente lo scopo di migliorare la salute mondiale attraverso le innovazioni tecnologiche. Il suo progetto, denominato RapidSMS, utilizza telefoni cellulari e SMS per facilitare la raccolta dei dati per i servizi sanitari ed educativi.

Il lavoro di Kochi e dei suoi partner ha permesso di sviluppare delle tecnologie open source che hanno consentito la registrazione di oltre 7 milioni di nascite in Nigeria e permesso di fornire cure a migliaia di donne in stato di gravidanza in tutto il Rwanda. Nel 2013 è stata nominata fra le "100 persone più influenti del mondo" da TIME!

Guarda questo video per saperne di più su Erica Kochi e su parte del lavoro che svolge presso l'UNICEF.

Riflettere

Per lavorare in campo informatico non basta avere talento nella programmazione. Prenditi del tempo per riflettere su come Erica e il suo lavoro hanno a che fare con i punti di forza su cui i grandi informatici puntano: coraggio, resilienza, creatività e motivazione.

 CORAGGIO	Una buona parte del lavoro di Erica Kochi consiste nel cercare di aiutare le persone che soffrono. Fino a che punto ciò rappresenta una sfida per la sua carriera?

Condividi le tue risposte con un familiare o un amico. Incoraggia gli altri a ottenere maggiori informazioni su Erica per partecipare alla discussione!

Passo 1: Esplora (5 min)

In questo tutorial creerai un chatbot personalizzato su Python! Un **chatbot** è un programma che simula le conversazioni con una persona reale. Si tratta di una forma semplificata di Intelligenza Artificiale, o IA. Tra i chatbot che puoi già utilizzare: Barista di Starbucks, Siri di Apple, Alexa di Amazon e l'assistente di Google.

Prenditi 5 minuti per esplorare alcune delle caratteristiche di questa chatbot che abbiamo creato sulle donne che operano in ambito tecnologico. Per il nostro esempio abbiamo individuato la tematica, il pubblico e l'obiettivo seguenti:

- **Tematica del progetto:** Donne/Tecnologia
- **Pubblico:** Persone che desiderano saperne di più sulle donne che lavorano con la tecnologia
- **Obiettivo**: Condividere aneddoti sulla tecnologia e fornire informazioni sulle donne che operano in ambito tecnologico

Mentre ti addentri alla scoperta di questo esempio di progetto, rifletti su:

- Come viene presentato il chatbot?
- Che genere di domande vengono poste? Riesce a raggiungere l'obiettivo?
- Quali caratteristiche dell'esempio vorresti integrare nel tuo progetto? Che cosa faresti di diverso nel tuo progetto?

Fase 2: Fai un brainstorming sulle caratteristiche del tuo progetto e pianificalo (10 min)

Adesso che hai avuto la possibilità di visionare un esempio di progetto, potrebbe essere una buona idea prenderti del tempo per creare innanzitutto un piano di gioco. Utilizza questo tempo per cercare di capire che cosa vuoi ottenere con il tuo progetto e qual è l'obiettivo. Perché non utilizzi i documenti sulla pianificazione (da pagina 16 a pagina 18) per raccogliere e organizzare le idee?

1. **Scegli una tematica, un pubblico e l'obiettivo del tuo chatbot.**
   Pensa a un argomento sul quale desideri concentrarti per il tuo chatbot e al risultato che vorresti ottenere. Puoi creare un chatbot a scopo informativo, un chatbot dal fondo umoristico, attorno al quale puoi costruire una community, o un chatbot basato su fatti interessanti/quiz. Se ti senti bloccato, qui di seguito troverai alcune potenziali idee per il tuo progetto:

   - Un chatbot che comunichi delle parole di incoraggiamento a quanti si sentono annoiati e/o isolati a casa
   - Un chatbot che dia dei consigli su come fare cose più divertenti a casa
   - Un chatbot che racconti degli aneddoti divertenti
   - Un chatbot che condivida dei fatti sui tuoi spettacoli, artisti, musicisti o hobby preferiti

   Quando scegli un pubblico per il tuo chatbot pensa al tipo di pubblico target che troverebbe il tuo prodotto interessante. Pensa a quali sono i loro interessi, che cosa vorrebbero sapere e in che modo potrai catturare la loro attenzione.

Fase 2: Pianifica il tuo progetto - Continua

2. **Scegli tre domande cui è possibile rispondere con "sì" o "no" per il tuo chatbot, e scrivi delle risposte per ogni domanda.**
   Pensa alle domande che vuoi porre e al modo in cui queste domande possono ricollegarsi alla tua tematica e all'obiettivo del tuo progetto. Che cosa accade se l'utente risponde "sì" alla tua domanda? E se invece rispondesse "no"? E nel caso in cui l'utente non rispondesse né con un sì, né con un no? Come dovrà rispondere il tuo chatbot a questi errori degli utenti?

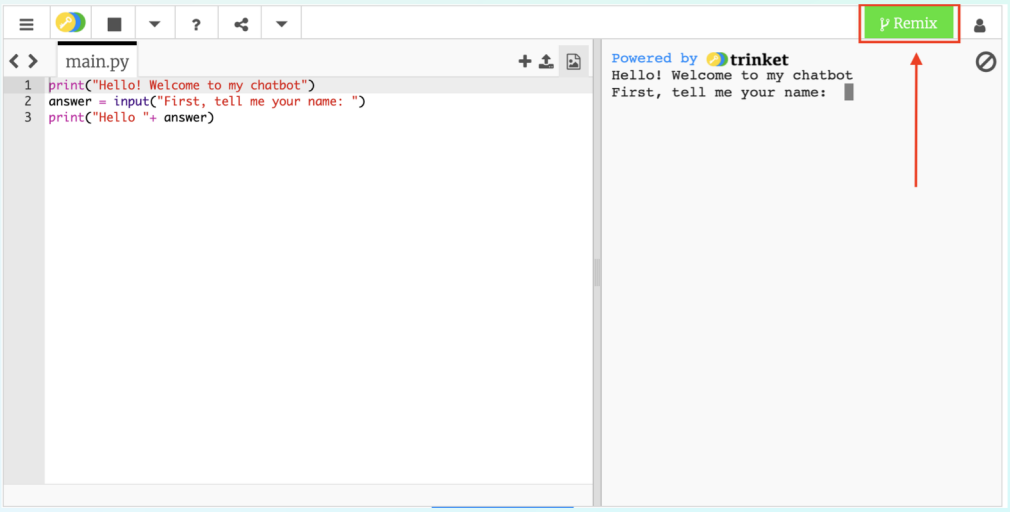
Fase 3: Cominciare con Trinket (10 min)

Python è un linguaggio di programmazione basato sul testo. Questo significa che tutti i comandi dovranno essere scritti! Su Pyhton i programmatori utilizzano diverse variabili, strutture di dati e funzioni per aiutare a immagazzinare informazioni e dare comandi! Python è un programma basato sul testo e ciò lo rende leggermente più ostico rispetto ad altri linguaggi, come per esempio Scratch, ma non impossibile da affrontare!

Utilizzeremo la piattaforma di programmazione Trinket per scrivere ed eseguire il nostro codice Python. Esistono diversi programmi che possono essere utilizzati per scrivere ed eseguire il codice Python. Abbiamo scelto questo perché puoi utilizzarlo direttamente sul tuo browser!

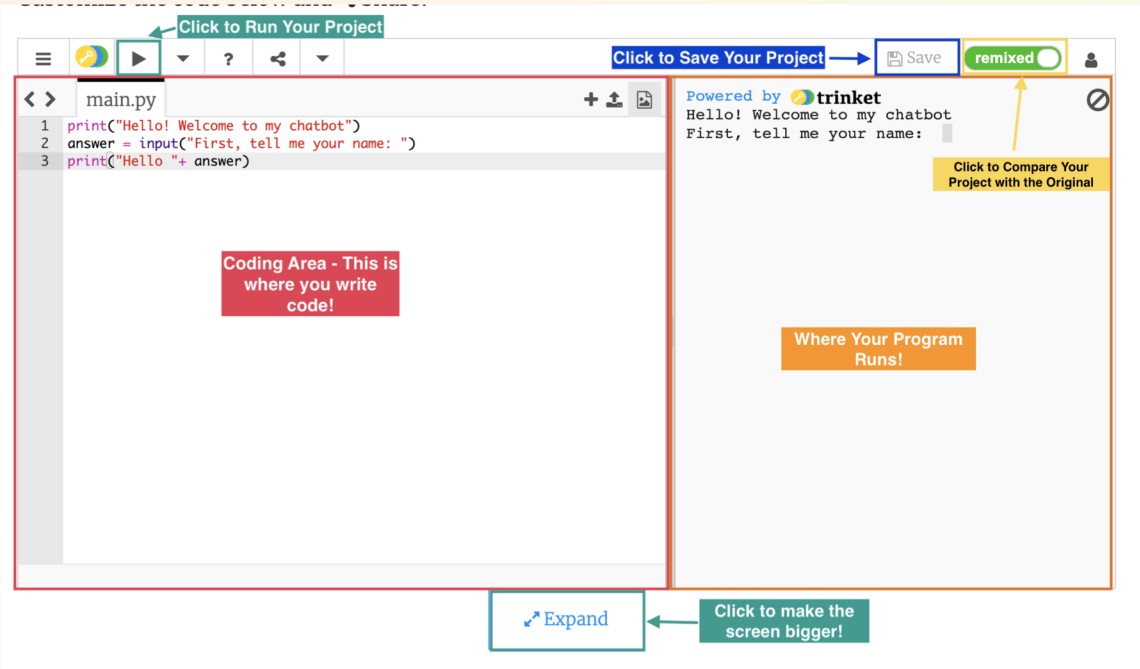
1. **Iscriviti o Accedi a Trinket.io**
   Per salvare il tuo lavoro su Trinket, dovrai creare un account, se non ne hai già uno. Segui le istruzioni sul modulo di iscrizione per creare un account. Se hai meno di 13 anni avrai bisogno dell'indirizzo e-mail dei tuoi genitori per registrarti.

2. **Remixa questo codice di avvio per il tuo chatbot.**
   Dopo aver trovato il codice di avvio, dovrai innanzitutto **remixare** o salvare una copia del progetto. Fai clic sul pulsante **Remix** che si trova sulla destra della finestra di Trinket.



Fase 3: Cominciare con Trinket - Continua

3. **Scopri l'interfaccia Trinket.**
   Se sei un nuovo utente di Trinket, impiega qualche minuto ad esplorarne l'interfaccia. Prendi dimestichezza su come salvare ed eseguire il tuo codice.



Fase 4: Scopri il Progetto iniziale (2 min)

Diamo un'occhiata al file d'avvio. Premi il pulsante Esegui __ per vedere qual è la funzione di questo codice.

Ci sarà un breve messaggio introduttivo da parte del programma:
Ciao! Benvenuto sul mio chatbot.

Questo messaggio sarà poi seguito da una domanda:
Sei contento di interagire con questo programma?

Prova a scrivere "sì" e guarda la risposta, poi scrivi "no" e di nuovo scrivi "Sì".

Noterai che tutte e tre le risposte riceveranno una replica diversa da parte del programma. Poiché Python è un programma basato sul testo, l'ortografia e la presenza di lettere maiuscole o minuscole conta! Le risposte "sì" e "Sì" vengono lette in modo differente dal computer.

Gli aspetti più importanti di un chatbot sono: (1) la capacità di chiedere all'utente di rispondere a una domanda e di (2) parlare con l'utente attraverso delle risposte. Impareremo innanzitutto come parlare all'utente attraverso i codici e poi parleremo di come porre domande e dare risposte.

Fase 5: Aggiungere una Introduzione (4 min)

Per permettere al computer di “parlare” con l’utente scriviamo delle istruzioni di stampa in codice. Successivamente, l’istruzione apparirà sulla sinistra della finestra di Trinket.

Diamo un’occhiata alla prima riga del codice presente sul file d’avvio. Premi il pulsante Esegui __ per vedere qual è la funzione di questo codice.

Ti accorgerai che la prima riga del codice si legge print(“Ciao! Benvenuto sul mio chatbot”) e la prima riga dello schermo dei **risultati** è Ciao! Benvenuto sul mio chatbot.

Diamo un’occhiata ora ai simboli chiave e ai termini per l’utilizzo di un’istruzione di print :

print(“testo campione”)

- print: Questa parola chiave permette al computer di stampare qualcosa sulla schermata dei risultati
- (): Le parentesi permettono al computer di stampare qualsiasi cosa si trovi al loro interno.
- “ “: Le doppie virgolette permettono al computer di riconoscere come parola qualunque elemento si trovi al loro interno.
- testo campione: Possiamo includere qualsiasi parola all’interno delle virgolette e il contenuto verrà stampato tale e quale sulla schermata dei **risultati**.

Provaci tu!

Aggiungi una stringa di codice per stampare una breve presentazione del tuo chatbot. In questa descrizione potresti includere la tematica e l’obiettivo del tuo chatbot.

Premi il pulsante Esegui __ per vedere se il computer stampa l’introduzione.

Soluzioni di debug:

- Errore di sintassi: Verifica la presenza delle doppie virgolette all’inizio e alla fine della domanda.
- Errore di sintassi: Verifica la presenza delle parentesi prima e dopo la domanda e che la stringa di codice **termini** con una parentesi di chiusura.

Fase 6: Fai la tua prima domanda (5 min)

Prima di porre la tua prima domanda, dobbiamo parlare di due termini importanti in informatica: **input** e **output.**

Il termine **input** viene utilizzato quando delle informazioni vengono fornite al computer. A volte ciò può significare premere un tasto, collegare un dispositivo come un mouse oppure, nel nostro caso, digitare una risposta. Il termine **output** viene utilizzato quando delle informazioni vengono fornite dal computer a un altro processo. La risposta da parte dell'utente (che si tratti di un "sì" o di un "no") costituisce l'**input** mentre la risposta del computer corrisponde all'**output** del processo di interrogazione.

Al fine di ottenere un input a una domanda su Python utilizziamo il comando input().

Nota che la seconda stringa di codice si legge risposta = input("Sei contento di interagire con questo programma?") e la prima stringa sulla schermata dei **Risultati** è
Sei contento di interagire con questo programma?

Diamo un'occhiata ora ai simboli chiave e ai termini per l'utilizzo di un'istruzione di input :

risposta = input("risposta campione ")

- risposta: Si tratta di una **variabile** che racchiude la risposta dell'utente. Tale variabile viene chiamata "risposta".
- =: Il simbolo dell'uguale rappresenta l'assegnazione o la riassegnazione di un valore a una risposta **variabile**.
- input: Questa parola chiave permette al computer di sapere come porre una domanda all'utente
- (): Le parentesi permettono al computer di stampare il testo che si trova al loro interno.
- " ": Le doppie virgolette permettono al computer di riconoscere qualunque elemento si trovi al loro interno come parola e facente parte del messaggio di input.
- domanda campione: Possiamo includere qualsiasi parola all'interno delle virgolette e tale domanda verrà stampata tale e quale sulla schermata dei **risultati**.

La **variabile** in informatica è un termine che viene utilizzato per immagazzinare informazioni (dati) in un programma. Alle variabili viene dato un nome specifico affinché possano essere facilmente ritrovate e modificate all'interno di un programma.

Fase 6: Fai la tua prima domanda - Continua

Provaci tu!

Dai nuovamente un'occhiata al tuo foglio di pianificazione e poni la prima domanda all'utente. Aggiungi una stringa di codice che segua lo stesso formato di quello riportato qui sotto.

```
risposta = input("domanda#1")
```

Sostituisci domanda#1 con la tua domanda.

Premi il pulsante Esegui __ per visualizzare se il computer pone la domanda.

Soluzioni di debug:

- Errore di sintassi: Verifica la presenza delle doppie virgolette all'inizio e alla fine della domanda.
- Errore di sintassi: Verifica la presenza delle parentesi prima e dopo la domanda e che la stringa di codice **termini** con una parentesi di chiusura.
- Errore di sintassi: Verifica di avere il simbolo dell'uguale dopo aver scritto la risposta del nome della variabile.

Fase 7: Replicare a una risposta (10 min)

Avrai sicuramente notato che il computer pone una domanda ma non fa altro che rispondere dopo che hai digitato una risposta. Questo succede perché al computer non era stato indicato che cosa fare di quei dati!

Quando si digita il codice risposta = input("domanda#1") la risposta dell'utente è **immagazzinata** nella risposta variabile. È così che potremo determinare se l'utente avrà risposto "sì" o "no".

Innanzitutto, dovremo usare delle istruzioni **condizionali** per paragonare le possibili risposte ("sì" o "no"). Un'istruzione **condizionale** verifica se una serie di regole (o di istruzioni) sono state rispettate e poi determina quali azioni debbano essere eseguite nel caso in cui tali regole o istruzioni siano vere o false. Esistono tre scelte possibili per la nostra domanda: "sì", "no" o qualsiasi altra risposta.

Quando su Python utilizziamo delle condizioni, dobbiamo usare le parole chiavi if, elif e/o else. Diamo ora un'occhiata al modo in cui i termini condizionali vengono utilizzati nel chatbot per determinare una replica appropriata a ogni possibile risposta.

Fase 7: Rispondere a una domanda - Continua

```
risposta = input("Sei contento di interagire con questo programma? ")
if (risposta == "sì"):
  print ("Anch'io non vedo l'ora!")
elif (risposta == "no"):
  print ("Oh, spero che tu possa cambiare idea!")
else:
  print ("Mmm... Non riesco a capire la tua risposta".)
```

- if: Parole chiave che indicano un'istruzione if di eventualità
- elif: Parola chiave che indica un'istruzione else-if di concessione. Si tratta di un'istruzione facoltativa ma deve apparire **dopo** un'istruzione if.
- else: Parola chiave che indica un'istruzione else di alterità. Si tratta di un'istruzione facoltativa ma deve apparire **dopo** un'istruzione if ed elif.
- ( ): le parentesi sono facoltative. Vengono aggiunte prima e dopo una condizione per aiutarci a mantenere il nostro codice organizzato e facile da leggere.
- risposta: Si tratta di una **variabile** che racchiude la risposta dell'utente. Tale variabile viene chiamata risposta.
- ==: Il simbolo di uguale è un simbolo di confronto. Viene utilizzato per confrontare la risposta variabile con un altro valore oppure, nel nostro caso, con il "sì" o con il "no"
- "sì"/"no": Le doppie virgolette vengono utilizzate per indicare al computer di leggere come testo il valore che si trova al loro interno. Poiché vogliamo confrontare la replica con queste risposte di parole, mettiamo le doppie virgolette a sì e no.
- : La virgola permette al programma di riconoscere la fine dell'istruzione condizionale.
- **Trattino**: Tutte le stringhe di codice che devono essere eseguite se viene rispettata una condizione dovrebbero presentare un trattino dopo l'istruzione di if. Ciò permette al computer di sapere quali stringhe del codice devono essere eseguite. Vedi l'esempio riportato sopra, dove il comando print è tratteggiato.

Ricordati che utilizziamo la variabile di risposta per *immagazzinare* la risposta dell'utente alla nostra domanda. La prima istruzione if innanzitutto verifica se la risposta variabile è "sì". Se la risposta è "sì", allora la stringa di codice tratteggiata che si trova sotto l'istruzione if viene stampata. Se la risposta non è "sì", allora si passa all'istruzione condizionale successiva, l'elif. Ciò permette quindi di confrontare la risposta con "no". Allo stesso modo, se la risposta è "no", allora la stringa di codice tratteggiata che si trova sotto l'istruzione elif viene stampata. L'ultima istruzione else è un'istruzione valida per ogni altro tipo di risposta che possa essere data. Poiché non ci aspettiamo di ricevere altre risposte al di là di sì o no, il computer stamperà l'istruzione tratteggiata per comunicare all'utente che si è trattato di una risposta non valida.

Fase 7: Continua

Provaci tu!

Dai nuovamente un'occhiata al tuo foglio di pianificazione e verifica le tue repliche alle eventuali risposte "sì", "no" o non valida date dall'utente.

Aggiungi queste stringhe di codice che seguono lo stesso formato **dopo** il codice che pone la tua prima domanda. Assicurati **di tratteggiare** le istruzioni di stampa dopo le istruzioni if, elif, ed else .

```
risposta = input("domanda#1 ")
if (risposta == "sì"):
  print ("risposta sì")
elif (risposta == "no"):
  print ("risposta no")
else:
  print ("altra risposta")
```

Aggiorna le risposte nelle istruzioni di stampa per ciascuna opzione con le risposte che hai riportato sul tuo documento di pianificazione

Premi il pulsante Esegui __ per visualizzare se il tuo computer risponde in modo opportuno alle tue risposte.

Fase 8: Porre più domande per raccogliere dati (10-15 min)

Ora, dopo aver scritto una domanda, segui le istruzioni indicate alle fasi 5 e 6 per porre le domande restanti. Il tuo programma dovrà seguire lo stesso formato di quello presentato qui sotto:

```
risposta = input(“domanda#1”)
if (risposta == “sì”):
  print (“risposta sì”)
elif (risposta == “no”):
  print (“risposta no”)
else:
  print (“altra risposta”)

risposta = input(“domanda#2”)
if (risposta == “sì”):
  print (“risposta sì”)
elif (risposta == “no”):
  print (“risposta no”)
else:
  print (“altra risposta”)

risposta = input(“domanda#3”)
if (risposta == “sì”):
  print (“risposta sì”)
elif (risposta == “no”):
  print (“risposta no”)
else:
  print (“altra risposta”)
```

Una volta che avrai terminato di scrivere una domanda non dimenticarti di **Testare** il tuo programma e di assicurarti che funzioni correttamente.

Soluzioni di debug:

- Errore di sintassi: Verifica la presenza delle doppie virgolette all’inizio e alla fine della domanda.
- Errore di sintassi: Verifica la presenza delle parentesi prima e dopo la domanda e che la stringa di codice **termini** con una parentesi di chiusura.
- Errore di sintassi: Verifica di avere il simbolo dell’uguale dopo aver scritto la risposta del nome della variabile.
- Errore di sintassi: Verifica che le istruzioni di stampa dopo le istruzioni if, elif ed else **siano tratteggiate.**

Fase 9: Estensioni per il tuo chatbot (15-30 min)

Hai a tua disposizione molti strumenti per portare il tuo progetto chatbot al livello successivo!

- **Aggiungere una domanda che presenti delle risposte diverse da "sì" o "no. (5-8 min)**
  Possiamo scrivere istruzioni condizionali per verificare qualsiasi cosa! In questo momento, le nostre istruzioni condizionali non fanno altro che confrontare la risposta dell'utente con "sì" o "no", ma tale stato di cose può essere facilmente modificato. Se per esempio volessimo modificare le nostre risposte possibili in "Vero" o "Falso", quello che dovremmo fare sarebbe semplicemente di aggiornare la condizione che si trova all'interno delle parentesi per leggere la risposta == "Vero"

Codice precedente	Codice aggiornato

risposta = input("domanda")
if (risposta == "sì"):
  print ("risposta sì")
elif (risposta == "no"):
  print ("risposta no")
else:
  print ("altra risposta")
	risposta = input("domanda")
	if (risposta == "Vero"):
	  print ("risposta vero")
	elif (risposta == "Falso"):
	  print ("risposta falso")
	else:
	  print ("altra risposta")

  Programma e aggiungi un'altra domanda che presenti risposte diverse da "sì" o "no".

- **Aggiungere una domanda che presenti più di due risposte possibili. (8-10 min)**
  Abbiamo preso in considerazione unicamente domande che comportano solo due risposte (sì o no); ma se invece volessimo porre una domanda che offra risposte multiple? Se volessimo ottenere un'altra risposta dobbiamo aggiungere un'altra istruzione **condizionale**. Ricorda che l'ordine delle istruzioni condizionali era il seguente: if, elif e poi else. Se volessimo aggiungere un'altra risposta possibile aggiungiamo un'ulteriore istruzione elif prima dell'istruzione else.

  Poiché else è un'istruzione adatta per tutte le altre possibili risposte, vogliamo essere sicuri di ottenere la nostra terza opzione prima di raggiungere l'istruzione else. Per esempio, se volessimo aggiungere la risposta "Forse", dovremmo aggiornare il codice come indicato di seguito.

Codice precedente	Codice aggiornato

risposta = input("domanda")
if (risposta == "sì"):
  print ("risposta sì")
elif (risposta == "no"):
  print ("risposta no")
 	
else:
  print ("altra risposta")
	risposta = input("domanda")
	if (risposta == "sì"):
	  print ("risposta sì")
	elif (risposta == "no"):
	  print ("risposta no")
	elif (risposta == "forse"):
	  print ("risposta forse")
	else:
	  print ("altra risposta")

Fase 9: Estensioni - Continua

- **Eliminare la distinzione fra minuscola e maiuscola nelle risposte. (5-8 min)**
  Non hai più voglia di verificare ogni volta che le risposte vengano scritte tutte in minuscolo? C'è un modo semplice per risolvere il problema! Per questa azione utilizzeremo il comando lower() che converte in caratteri minuscoli tutte le lettere di una parola. Se la parola è già scritta in lettere minuscole, questo comando non avrà nessun tipo di impatto e la parola rimarrà tale e quale. Se la parola presenta invece un mix di lettere maiuscole e minuscole, allora tutte le lettere saranno convertite in minuscole (per es. GWC → gwc)

  Per utilizzare il comando lower() possiamo aggiungere una nuova stringa di codice che riassegni la variabile di risposta alla versione di lettere minuscole come segue:

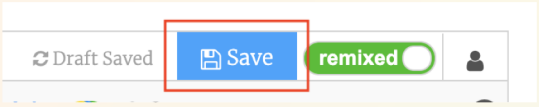
Codice precedente	Codice aggiornato

risposta = input("domanda")
 	
if (risposta == "sì"):
  print ("risposta sì")
elif (risposta == "no"):
  print ("risposta no")
else:
  print ("altra risposta")
	risposta = input("domanda")
	risposta = risposta.lower()
	if (risposta == "sì"):
	  print ("risposta sì")
	elif (risposta == "no"):
	  print ("risposta no")
	else:
	  print ("altra risposta")

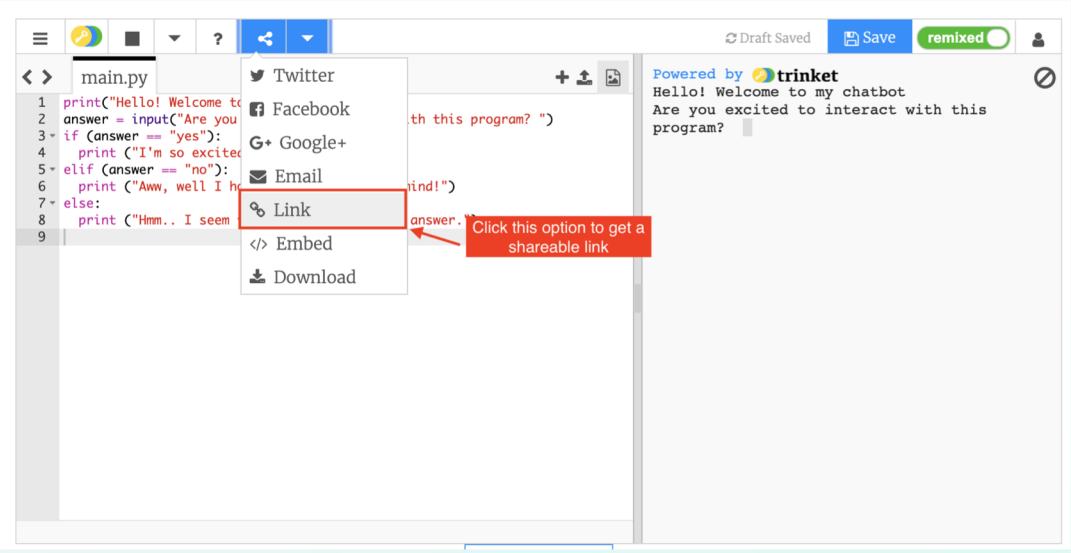
  Dobbiamo aggiungere questa nuova stringa dopo **ciascuna** domanda posta.

Fase 10: Condividi il tuo progetto Your Girls Who Code at Home! (5 min)

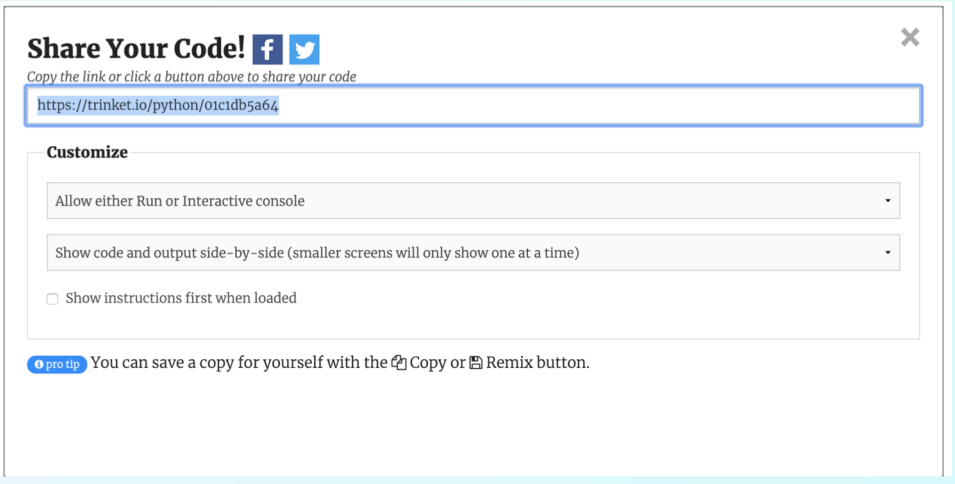
- Non dimenticarti di salvare il tuo lavoro facendo clic sul pulsante **Salva** che si trova sulla destra della finestra di Trinket



- Se desideri condividere il tuo progetto con i tuoi familiari e gli amici, fai clic sull'icona **Condividi** sulla sinistra e seleziona l'opzione **Link** dal menu a tendina.



- Condividi una foto o un video del tuo chatbot in funzione o copia il link e condividi il tuo chatbot sui social media! Ricordati di taggare @girlswhocode e di utilizzare l'hashtag #codefromhome: potremmo anche citarti sul nostro account!



Ti posso aiutare? Progettare foglio di pianificazione

Progettare panoramica pianificazione

Tematica: Di quale tematica parlerai nel tuo chatbot?

Pubblico: A chi servirà il tuo chatbot? Che gruppo di persone sarà interessato al tuo chatbot?

Obiettivo: Che cosa vuoi ottenere con il tuo chatbot? Perché è interessante per il tuo pubblico?

Pianificazione delle domande

Scegli tre domande cui è possibile rispondere con “sì” o “no” da porre ai tuoi utenti. Scrivi una domanda nella prima riga e successivamente scrivi la risposta del chatbot in ogni riga, che corrisponde a “sì” o “no”. Python è un sistema molto pignolo nell’accettare le risposte. L’ultima opzione “altre risposte” viene utilizzata per replicare a risposte diverse da sì o no. Eventualmente, puoi anche aggiungere un messaggio che informi l’utente che la sua risposta non è valida.

Domanda 1:	
Sì	
No	
Altra risposta	

Pianificazione delle domande - Continua

Domanda 2:	
Sì	
No	
Altra risposta	

Domanda 3:	
Sì	
No	
Altra risposta	