Girls Who Code à la maison

Puis-je vous aider?
Agent conversationnel avec Python

Aperçu de l'activité

Dans cette activité, vous explorerez les concepts de base de l'informatique avec Python. Vous apprendrez également à aider votre communauté en créant un agent conversationnel ou un système d'aide automatisé! Vous avez probablement déjà interagi avec un agent conversationnel auparavant, peut-être même sans vous en rendre compte. Certains agents conversationnels courants incluent : Siri d'Apple, Alexa d'Amazon, l'Assistant Google, Lyft et, plus récemment, l'Organisation mondiale de la santé en a mis un sur pied pour les faits relatifs à la COVID-19. Avant de commencer à concevoir et à coder votre agent conversationnel, consultez notre Mise en vedette de femmes en technologie concernant Erica Kochi. En tant que cofondatrice de l'Innovation Unit de l'UNICEF, Erica a conçu un agent conversationnel pour aider à fournir des soins et de l'assistance aux habitants du Nigéria et du Rwanda.

Matériel

- Éditeur Trinket
- Code de démarrage Trinket
- Exemple de projet d'agent conversationnel
- Facultatif : Guide de planification
- Facultatif : Stylo/crayon/marqueurs

Mise en vedette de femmes en technologie : Erica Kochi

Erica Kochi a cofondé et codirige l'Innovation Unit de l'UNICEF, qui vise à améliorer la santé mondiale grâce aux innovations technologiques. Son projet appelé RapidSMS utilisait des téléphones portables et des SMS pour faciliter la collecte de données pour les services de santé et d'éducation.

Le travail d'Erica avec ses partenaires a aidé à développer les technologies de code source libre qui ont enregistré plus de 7 millions de naissances au Nigéria et fourni des soins à des milliers de femmes enceintes à travers le Rwanda. En 2013, elle a été nommée dans la liste des 100 personnes les plus influentes au mode du Time!

Regardez cette vidéo pour en savoir plus sur Erica Kochi et une partie du travail qu'elle accomplit à l'UNICEF.

Réfléchir

Être informaticien, c'est bien plus qu'être doué pour le codage. Prenez le temps de réfléchir à la façon dont Erica et son travail sont liés aux forces que les grands informaticiens s'efforcent de développer : la bravoure, la ténacité, la créativité et la détermination.

 BRAVOURE	Erica Kochi s'efforce principalement d'aider à alléger la souffrance des autres. En quoi cela pourrait-il être difficile dans une carrière?

Partagez vos réponses avec un membre de votre famille ou un ami. Encouragez les autres à en savoir plus sur Erica pour participer à la discussion!

Étape 1: Explorer (5 min)

Dans ce tutoriel, vous allez créer un agent conversationnel personnalisé avec Python! Un **agent conversationnel** est un programme informatique qui simule des conversations avec une personne réelle. C'est une forme très simple d'intelligence artificielle, ou IA. Certains agents conversationnels que vous pouvez déjà utiliser incluent : Barista de Starbucks, Siri d'Apple, Alexa d'Amazon et l'Assistant Google.

Prenez 5 minutes pour explorer certaines des fonctionnalités de l'agent conversationnel que nous avons créé sur les femmes travaillant dans la technologie. Pour notre exemple, nous avons sélectionné le thème, le public et l'objectif suivants :

- **Thème du projet :** Femmes/Technologie
- **Public :** Les personnes qui veulent en savoir plus sur les femmes travaillant dans la technologie.
- **Objectif** : Partager des blagues basées sur la technologie et fournir des informations sur les femmes travaillant dans la technologie.

Lorsque vous explorez l'exemple de projet, pensez à ce qui suit :

- Comment l'agent conversationnel est-il présenté?
- Quels types de questions sont posées? Est-ce que l'agent atteint l'objectif?
- Quelles fonctionnalités de l'exemple voudriez-vous intégrer à votre propre projet? Que voudriez-vous faire différemment dans votre propre projet?

Étape 2: Faites un remue-méninges sur les fonctionnalités et planifiez votre projet (10 min)

Maintenant que vous avez eu la chance d'explorer un exemple de projet, prenez le temps de faire un plan de match. Utilisez ce temps pour déterminer ce que vous voulez que votre projet fasse et son objectif. Vous pouvez utiliser le document de planification (pages 16-18) pour vous aider à capturer et organiser vos idées.

1. **Choisissez un thème, un public et un objectif pour votre agent conversationnel.**
   Pensez au sujet sur lequel vous souhaitez vous concentrer avec votre agent conversationnel et à ce qu'il devrait accomplir. Vous pouvez créer un agent conversationnel informatif, plein d'humour, renforçant la communauté ou présentant des faits/un jeu-questionnaire. Si vous vous sentez coincé, voici quelques idées de projets possibles :

   - Un agent conversationnel qui fournit des mots d'encouragement aux autres s'ils se sentent ennuyés ou isolés à la maison.
   - Un agent conversationnel qui donne des suggestions sur la façon d'intégrer plus de plaisir à la maison.
   - Un agent conversationnel qui raconte des blagues.
   - Un agent conversationnel qui partage des faits sur votre émission, artiste, musicien ou passe-temps préférés.

   Lorsque vous choisissez le public de votre agent conversationnel, pensez au public cible qui serait ravi d'utiliser votre produit. Réfléchissez à ce qu'il souhaite savoir et comment vous capterez son attention.

Étape 2: Planifiez votre projet (suite)

2. **Choisissez trois questions « oui ou non » que votre agent conversationnel doit poser et rédigez des réactions pour chaque réponse possible.**
   Réfléchissez aux questions que vous souhaitez poser et à la façon dont ces questions sont reliées à votre thème et à l'objectif du projet. Que se passe-t-il si la réponse de l'utilisateur à votre question est « oui »? Et si la réponse est « non »? Que se passe-t-il si l'utilisateur ne tape pas oui ou non? Comment votre agent conversationnel doit-il réagir à ces erreurs de l'utilisateur?

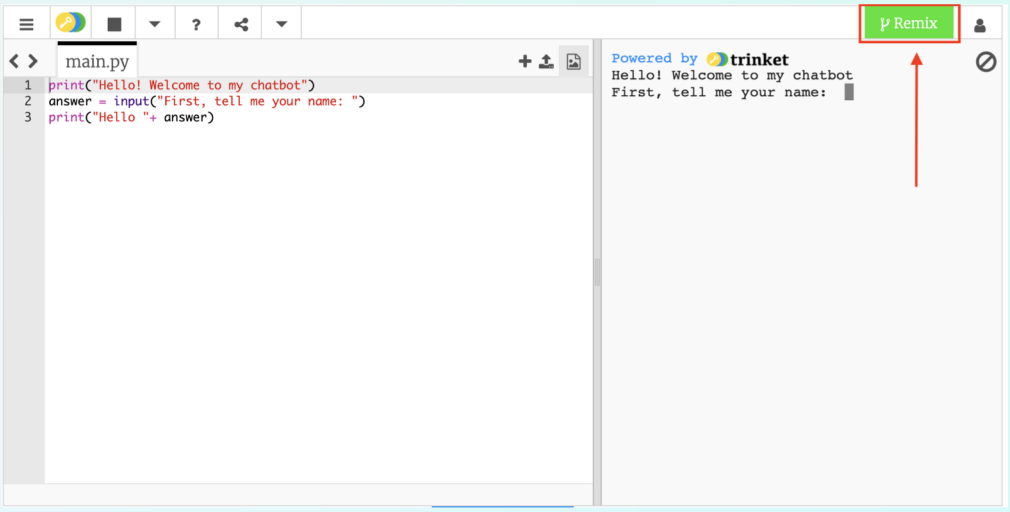
Étape 3: Commencez avec Trinket (10 min)

Python est un langage de programmation textuel, ce qui signifie que toutes les commandes devront être saisies! Avec Python, les programmeurs utilisent de nombreuses variables, structures de données et fonctions pour aider à stocker des informations et à exécuter des commandes. Puisque Python est textuel, il est un peu plus difficile à utiliser que d'autres langages, comme Scratch, mais il n'est pas impossible d'y parvenir!

Nous utiliserons la plate-forme de codage Trinket pour écrire et exécuter notre code Python. De nombreux programmes peuvent être utilisés pour écrire et exécuter du code Python. Nous avons choisi celui-ci, car vous pouvez l'utiliser directement dans votre navigateur!

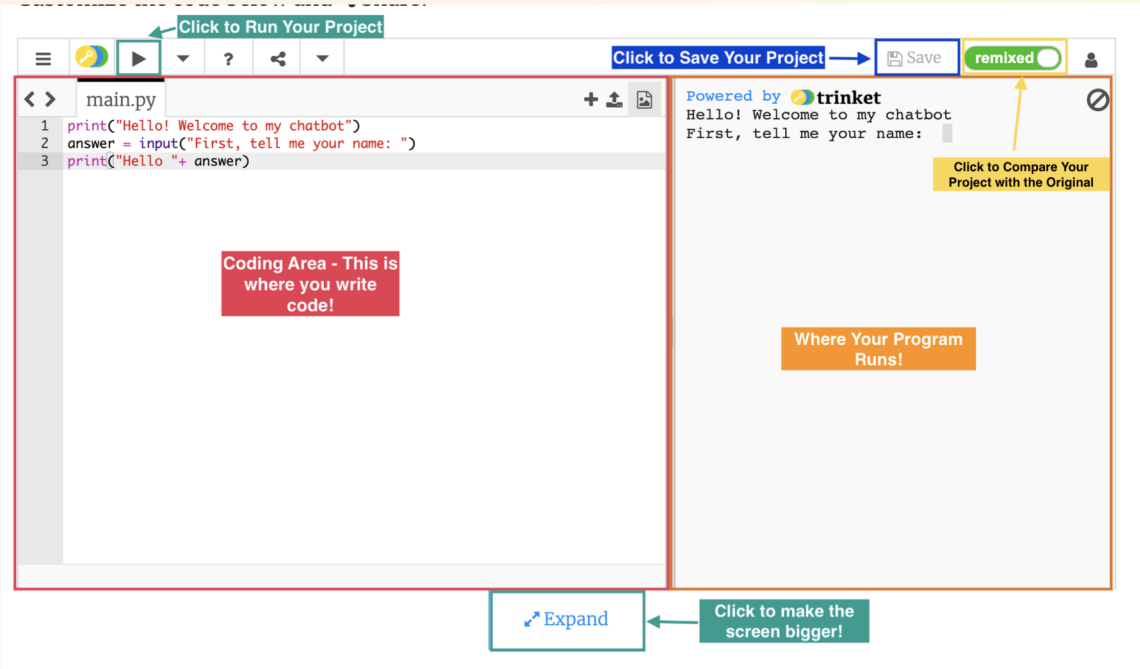
1. **Inscrivez-vous ou connectez-vous dans Trinket.io.**
   Pour enregistrer votre travail sur Trinket, vous devrez créer un compte si vous n'en avez pas déjà un. Suivez les instructions du formulaire d'inscription pour créer un compte. Si vous avez moins de 13 ans, vous aurez besoin de l'adresse courriel de vos parents pour vous inscrire.

2. **Remixez ce code de démarrage pour votre agent conversationnel.**
   Après avoir accédé au code de démarrage, vous devez **remixer** ou enregistrer une copie du projet. Cliquez sur le bouton **Remix** sur le côté gauche de la fenêtre Trinket.



Étape 3 Commencez avec Trinket (suite)

3. **Explorez l'interface Trinket.**
   Si vous êtes nouveau sur Trinket, prenez quelques minutes pour explorer l'interface Trinket. Familiarisez-vous avec la façon d'enregistrer et d'exécuter votre code.



Étape 4 Explorez le projet de démarrage (2 min)

Jetons un œil au fichier de démarrage. Appuyez sur le bouton Exécuter __ pour voir ce que fait ce code.

Vous remarquerez qu'il y a une courte introduction où le programme dit :
Bonjour! Bienvenue sur mon agent conversationnel.

La présentation est suivie d'une question :
Avez-vous hâte d'interagir avec ce programme?

Essayez de taper « oui » pour voir la réponse, tapez « non », puis tapez « Oui ».

Vous remarquerez que ces trois réponses font réagir le programme différemment. Puisque Python est un langage textuel, l'orthographe et la casse des lettres comptent! Les réponses « oui » et « Oui » sont lues différemment par l'ordinateur.

Les aspects les plus importants d'un agent conversationnel sont la possibilité (1) de demander à l'utilisateur une réponse à une question et (2) de lui parler en imprimant les réponses. Nous allons d'abord apprendre à parler à l'utilisateur avec le code, avant de discuter de la façon de poser des questions et de répondre.

Étape 5 Ajouter une introduction (4 min)

Pour que l'ordinateur « parle » à l'utilisateur, nous écrivons des instructions print dans le code, qui s'affichent ensuite sur le côté gauche de la fenêtre Trinket.

Observons la première ligne de code du fichier de démarrage. Appuyez sur le bouton Exécuter __ pour voir ce que fait ce code.

Notez que la première ligne de code affiche print("Bonjour! Bienvenue sur mon agent conversationnel"), et la première ligne de l'écran de **résultats** est Bonjour! Bienvenue sur mon agent conversationnel.

Examinons les symboles et termes clés pour utiliser une instruction print :

print("exemple de texte")

- print : Ce mot-clé indique à l'ordinateur d'imprimer quelque chose sur l'écran des résultats.
- () : Les parenthèses indiquent à l'ordinateur d'imprimer tout ce qui se trouve entre les parenthèses.
- " " : Les guillemets doubles permettent à l'ordinateur de savoir que tout ce qui se trouve à l'intérieur sont des mots.
- exemple de texte : Nous pouvons inclure tous les mots à l'intérieur des guillemets, et ceux-ci seront imprimés exactement tels quels sur l'écran des **résultats**.

Essayez-le!

Ajoutez une ligne de code pour imprimer une courte présentation de votre agent conversationnel. Vous voudrez peut-être inclure le thème et l'objectif dans cette description.

Appuyez sur le bouton Exécuter __ pour voir si l'ordinateur imprime votre présentation.

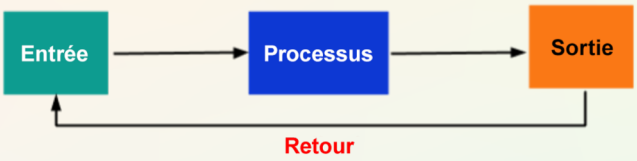
Conseils de débogage

- Erreur de syntaxe : Assurez-vous d'avoir des guillemets doubles autour de votre question, soit au début et à la fin de la question.
- Erreur de syntaxe : Assurez-vous d'avoir des parenthèses autour de la question et que la ligne de code **se termine** par une parenthèse fermante.

Étape 6 Posez votre première question (5 min)

Afin de poser votre première question, examinons deux termes informatiques importants : **saisie** et **sortie.**

La **saisie** est quand une sorte d'information est donnée à l'ordinateur. Il peut s'agir d'appuyer sur une touche, de brancher un appareil comme une souris ou, dans notre cas, de taper une réponse. La **sortie** est quand l'information est donnée de l'ordinateur à un autre processus. La réponse de l'utilisateur (« oui » ou « non ») est la **saisie** tandis que la réponse de l'ordinateur est la **sortie** du processus de question.



Afin d'obtenir une saisie à une question avec Python, nous utilisons la commande input().

Notez que la deuxième ligne de code indique answer = input("Avez-vous hâte d'interagir avec ce programme?") et que la première ligne de l'écran **Résultats** est Avez-vous hâte d'interagir avec ce programme?

Examinons les symboles et termes clés pour utiliser une instruction de saisie :

answer = input("exemple de question")

- answer : Il s'agit d'une **variable** qui stocke la réponse de l'utilisateur. En particulier, cette variable est nommée « answer ».
- = : Le signe égal indique l'affectation ou la réaffectation d'une valeur à la réponse **variable**.
- input: Pour que l'ordinateur « parle » à l'utilisateur, nous écrivons des instructions print dans le code, qui s'affichent ensuite sur le côté gauche de la fenêtre Trinket.
- () : Les parenthèses indiquent à l'ordinateur d'imprimer le texte qui se trouve entre les parenthèses.
- " " : Les guillemets doubles permettent à l'ordinateur de savoir que tout ce qu'elles contiennent est des mots et fait partie du message de saisie.
- exemple de question : Nous pouvons inclure n'importe quel mot à l'intérieur des guillemets, et cette question sera imprimée exactement ainsi sur l'écran des **résultats**.

Une **variable** est un terme informatique utilisé pour stocker des informations (données) dans un programme informatique. Les variables reçoivent un nom afin d'être facilement référencées et modifiées dans un programme.

Étape 6 Posez votre première question (suite)

Essayez-le!

Revenez sur votre feuille de travail de planification et posez votre première question à l'utilisateur. Ajoutez une nouvelle ligne de code qui suit le même format que ci-dessous.

```
answer = input("question 1")
```

Remplacez question 1 par votre question.

Appuyez sur le bouton Exécuter __ pour voir si l'ordinateur pose votre question.

Conseils de débogage

- Erreur de syntaxe : Assurez-vous d'avoir des guillemets doubles autour de votre question, soit au début et à la fin de la question.
- Erreur de syntaxe : Assurez-vous d'avoir des parenthèses autour de la question et que la ligne de code **se termine** par une parenthèse fermante.
- Erreur de syntaxe : Assurez-vous d'avoir un signe égal après avoir écrit la réponse à nom variable.

Étape 7 Réagir à une réponse (10 min)

Vous avez peut-être remarqué que l'ordinateur pose une question, mais ne fait rien d'autre pour réagir une fois que vous avez tapé une réponse. C'est parce que nous n'avons pas dit à l'ordinateur de faire quoi que ce soit avec ces données!

Lors de la saisie du code : `answer = input("question 1")`, la réponse de l'utilisateur est **stockée** dans la variable answer. C'est ainsi que nous déterminerons si l'utilisateur a répondu « oui » ou « non ».

Premièrement, nous devons utiliser des **instructions conditionnelles** pour comparer les choix possibles (« oui » ou « non »). Une instruction **conditionnelle** vérifie si un ensemble de règles (ou instruction) est respecté, puis décide quelles actions sont effectuées si les règles ou l'instruction sont vraies ou fausses. Il y a trois choix possibles pour notre question : « oui », « non » ou toute autre réponse.

Lorsque nous utilisons des conditions avec Python, nous devons utiliser les mots-clés if, elif ou else. Voyons comment les conditions sont utilisées dans l'agent conversationnel pour déterminer la réaction appropriée à chaque réponse possible.

Étape 7 Réagir à une réponse (suite)

```
answer = input ("Avez-vous hâte d'interagir avec ce programme?")
if (answer == "oui") :
  print ("J'ai vraiment hâte aussi!")
elif (answer == "non") :
  print ("Oh, j'espère que je peux vous faire changer d'avis!")
else:
  print ("Hum... Je ne comprends pas votre réponse.")
```

- if : Mot-clé pour indiquer une instruction if.
- elif : Mot-clé pour indiquer une instruction else-if. Il s'agit d'une instruction facultative, mais doit **suivre** une instruction if.
- else : Mot-clé pour indiquer une instruction else. Il s'agit d'une instruction facultative, mais elle doit **suivre** une instruction if et elif.
- ( ) : Les parenthèses sont facultatives. Elles sont ajoutées autour de la condition pour aider à garder notre code organisé et qu'il soit plus facile à lire.
- answer : Il s'agit d'une **variable** qui stocke la réponse de l'utilisateur. En particulier, cette variable est nommée answer.
- == : Le signe égal est un comparateur. Il est utilisé pour comparer la réponse variable à une autre valeur, dans notre cas, « oui » ou « non ».
- "oui"/"non" : Les guillemets doubles sont utilisés pour indiquer à l'ordinateur de lire la valeur qui s'y trouve sous forme de texte. Puisque nous voulons comparer la réponse à ces réponses en mots, nous utilisons des guillemets doubles autour de oui et non.
- : Les deux points indiquent au programme la fin de l'instruction de condition.
- **Retrait** : Toutes les lignes de code qui doivent être exécutées si une condition est remplie doivent être mises en retrait après l'instruction if. Ainsi, l'ordinateur peut savoir quelles lignes de code doivent être exécutées. Voir l'exemple ci-dessus où la commande print est en retrait.

Rappelons que nous utilisons la variable answer pour *stocker* la réponse de l'utilisateur à notre question. La première instruction if compare d'abord si la variable answer est « oui ». Si la réponse est « oui », la ligne de code en retrait sous l'instruction if est imprimée. Si la réponse n'est pas « oui », nous passons alors à la prochaine instruction conditionnelle, elif. Cela compare ensuite la réponse à « non ». De même, si la réponse est « non », la ligne de code en retrait sous l'instruction elif est imprimée. La dernière instruction else est un fourre-tout pour tous les autres types de la réponse. Étant donné que nous n'attendons aucune autre réponse que oui ou non, l'ordinateur imprime ensuite l'instruction en retrait pour faire savoir à l'utilisateur qu'il s'agissait d'une réponse non valide.

Étape 7 (suite)

Essayez-le!

Consultez votre feuille de travail de planification et vérifiez vos réponses pour savoir si l'utilisateur répond « oui », « non » ou une réponse non valide.

Ajoutez ces lignes de code qui respectent le même format **après** le code qui pose votre première question. Veillez à **mettre en retrait** les instructions print après les instructions if, elif et else.

```
answer = input("question 1")
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
else :
  print ("autre réponse")
```

Mettez à jour les réponses dans les instructions print pour chaque option avec les réponses écrites dans votre document de planification.

Appuyez sur le bouton Exécuter __ pour voir si l'ordinateur répond en fonction de vos réponses.

Étape 8 Poser plus de questions pour recueillir des données (10-15 min)

Maintenant que vous avez écrit une question, suivez les instructions des étapes 5 et 6 pour poser vos questions restantes. Votre programme doit respecter le même format que ci-dessous :

```
answer = input("question 1")
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
else :
  print ("autre réponse")

answer = input("question 2")
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
else :
  print ("autre réponse")

answer = input("question 3")
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
else:
  print ("autre réponse")
```

Après avoir terminé d'écrire une question, n'oubliez pas de **tester** votre programme pour vous assurer qu'il fonctionne correctement.

Conseils de débogage

- Erreur de syntaxe : Assurez-vous d'avoir des guillemets doubles autour de votre question, soit au début et à la fin de la question.
- Erreur de syntaxe : Assurez-vous d'avoir des parenthèses autour de la question et que la ligne de code **se termine** par une parenthèse fermante.
- Erreur de syntaxe : Assurez-vous d'avoir un signe égal après avoir écrit la réponse à nom variable.
- Erreur de syntaxe : Assurez-vous que les instructions print après les instructions if, elif et else sont **en retrait.**

Étape 9 Extensions pour votre agent conversationnel (15-30 min)

Il y a tellement de façons de faire passer votre projet d'agent conversationnel au niveau supérieur!

- **Ajoutez une question avec des réponses autres que « oui » ou « non » (5-8 min)**
  Nous pouvons écrire des instructions conditionnelles pour vérifier n'importe quoi! À l'heure actuelle, nos instructions conditionnelles ne comparent que la réponse de l'utilisateur à « oui » ou « non », mais nous pouvons facilement le modifier. Si nous voulions changer nos réponses possibles en « Vrai » ou « Faux », nous mettons simplement à jour la condition entre parenthèses pour écrire answer == "True".

Code précédent	Code mis à jour

answer = input("question")
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
else:
  print ("autre réponse")
	answer = input("question")
	if (answer == "Vrai"):
	  print ("réponse vraie")
	elif (answer == "Faux"):
	  print ("réponse faux")
	else:
	  print ("autre réponse")

  Planifiez et ajoutez une autre question dont les réponses sont différentes de « oui » ou « non ».

- **Ajoutez une question avec plus de deux réponses possibles (8-10 min)**
  Nous n'avons considéré que des questions avec seulement deux réponses (oui ou non), mais que faire si nous voulions poser une question qui pourrait avoir plusieurs réponses? Si nous voulons capturer une autre réponse, nous devons ajouter une autre instruction **conditionnelle**. Rappelons que l'ordre des instructions conditionnelles était if, elif, puis else. Si nous voulons ajouter une autre réponse possible, nous ajoutons une instruction elif supplémentaire avant l'instruction else.

  Comme else est un fourre-tout pour toutes les autres réponses possibles, nous voulons nous assurer de capturer notre troisième option avant d'atteindre l'instruction else. Par exemple, si nous voulons ajouter la réponse « Peut-être », nous mettrons à jour le code comme illustré ci-dessous.

Code précédent	Code mis à jour

answer = input("question")
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
 	
else:
  print ("autre réponse")
	answer = input("question")
	if (answer == "oui"):
	  print ("réponse oui")
	elif (answer == "non"):
	  print ("réponse non")
	elif (answer == "peut-être"):
	  print ("réponse peut-être")
	else:
	  print ("autre réponse")

Étape 9 Extensions (suite)

- **Supprimer la sensibilité à la casse pour les réponses (5-8 min)**
  Vous en avez assez de devoir vous assurer de taper les réponses en minuscules? Il existe un moyen simple de résoudre ce problème! Pour y parvenir, nous utiliserons la commande lower() qui convertit un mot en minuscules. Si le mot est déjà en minuscules, cette commande ne fait rien et conserve le mot tel quel. Si le mot a des casses mixtes, toutes les lettres seront converties en minuscules (p. ex. GWC → gwc).

  Pour utiliser la commande lower(), nous pouvons ajouter une nouvelle ligne de code qui réaffecte la variable réponse à la version en minuscules comme suit :

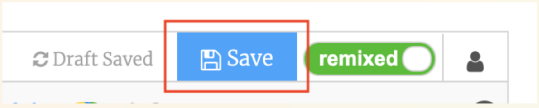
Code précédent	Code mis à jour

answer = input("question")
 	
if (answer == "oui"):
  print ("réponse oui")
elif (answer == "non"):
  print ("réponse non")
else:
  print ("autre réponse")
	answer = input("question")
	answer = answer.lower()
	if (answer == "oui"):
	  print ("réponse oui")
	elif (answer == "non"):
	  print ("réponse non")
	else:
	  print ("autre réponse")

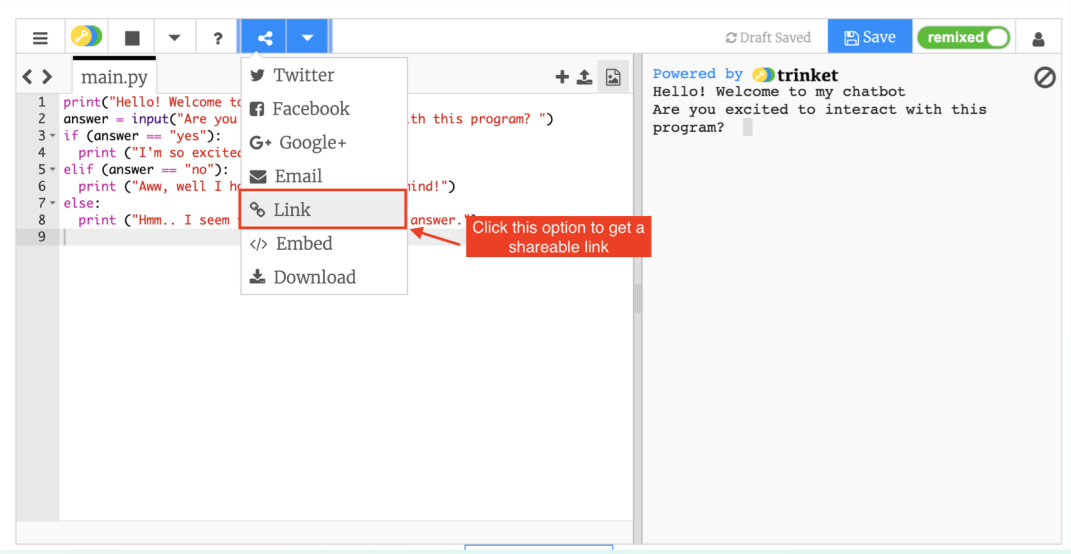
  Nous devons ajouter cette nouvelle ligne après **chaque** question est posée.

Étape 10 Partagez vos projets Girls Who Code à la maison! (5 min)

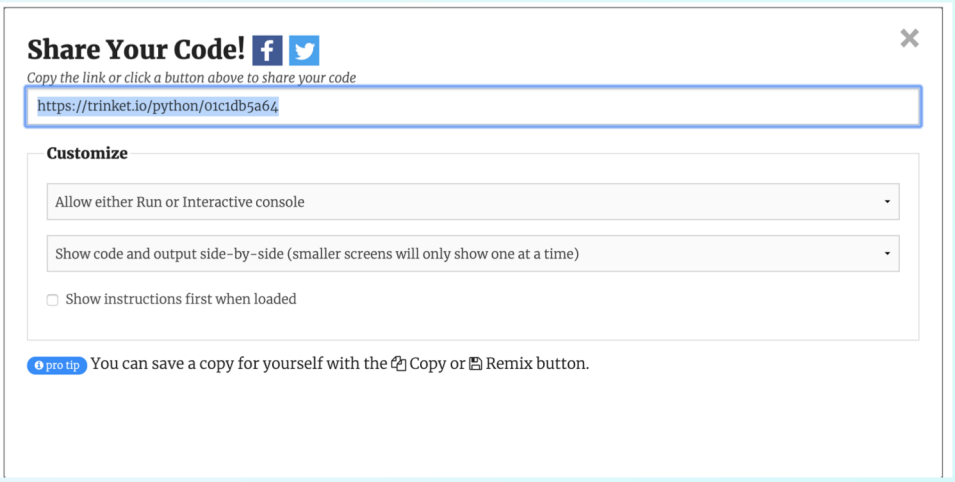
- N'oubliez pas de sauvegarder votre travail en cliquant sur le bouton **Enregistrer** sur le côté droit de la fenêtre Trinket.



- Pour partager votre projet avec vos amis et votre famille, cliquez sur l'icône **Partager** à gauche et choisissez l'option **Lien** dans le menu déroulant.



- Partagez une photo ou une vidéo de votre agent conversationnel de travail, ou copiez le lien et partagez votre agent conversationnel sur les réseaux sociaux! N'oubliez pas d'identifier @girlswhocode et d'utiliser le mot-clic #codefromhome, et nous pourrions même vous présenter sur notre compte!



Puis-je vous aider? Feuille de travail de planification de projet

Planification de l’aperçu du projet

Thème : Quel sujet sera traité avec votre agent conversationnel?

Public : Qui cet agent conversationnel va-t-il servir? Quel groupe de personnes sera intéressé par votre produit?

Objectif : Que voulez-vous que votre agent conversationnel accomplisse? Pourquoi votre public serait-il intéressé?

Planification des questions

Choisissez trois questions oui ou non à poser à vos utilisateurs. Écrivez la question dans la première ligne et la réponse de votre agent conversationnel dans chaque ligne pour « oui » ou « non ». Python est très pointilleux dans l'acceptation des réponses. La dernière option « autres réponses » sert à réagir à des réponses autres que oui ou non. Vous souhaiterez peut-être ajouter un message indiquant à l'utilisateur que sa réponse n'est pas valide.

Question 1 :	
Oui	
Non	
Autre réponse	

Planification des questions (suite)

Question 2 :	
Oui	
Non	
Autre réponse	

Question 3 :	
Oui	
Non	
Autre réponse	