**Liste des commandes Python**



Table des matières

montrer

* [Introduction](https://www.interviewbit.com/blog/python-commands/#introduction)
* [Liste des commandes Python](https://www.interviewbit.com/blog/python-commands/#python-commands-list)
  + [Commandes de base en python que vous devez connaître](https://www.interviewbit.com/blog/python-commands/#basic-python-commands-you-must-know)
  + [Commandes intermédiaires en Python](https://www.interviewbit.com/blog/python-commands/#intermediate-python-commands)
  + [Liste avancée des commandes en Python](https://www.interviewbit.com/blog/python-commands/#advanced-python-commands-list)
* [Conclusion](https://www.interviewbit.com/blog/python-commands/#conclusion)
* [Foire aux questions](https://www.interviewbit.com/blog/python-commands/#frequently-asked-questions)
* [Ressources supplémentaires](https://www.interviewbit.com/blog/python-commands/#additional-resources)

## **Introduction**

Python est l'un des langages de programmation les plus populaires. Il s'agit d'un langage général de haut niveau interprété. Python est développé par Guido van Rossum en 1991. Sa syntaxe est très conviviale pour les débutants et facile à apprendre. C'est la raison pour laquelle la plupart des gens suggèrent le langage python aux débutants comme premier langage de programmation à apprendre. Dans ce billet, nous allons discuter des **commandes python de haut niveau** qui peuvent faciliter votre [**parcours d'apprentissage du python**](https://www.interviewbit.com/online-python-compiler/).

Dans le langage de programmation Python, les commandes se réfèrent essentiellement à différentes fonctions ou méthodes que nous pouvons exécuter sur le shell de python pour les travailler comme des commandes. Selon la documentation officielle de [Python](https://docs.python.org/3/), il n'y a pas de « commandes » en Python mais nous avons différents types de fonctions comme input(), type(), len(), ainsi de suite. Donc dans ce billet, nous allons utiliser les termes commandes et fonctions indifféremment. Maintenant, commençons avec la liste des commandes de python et discutons chaque commande en détail.

## **Liste des commandes Python**

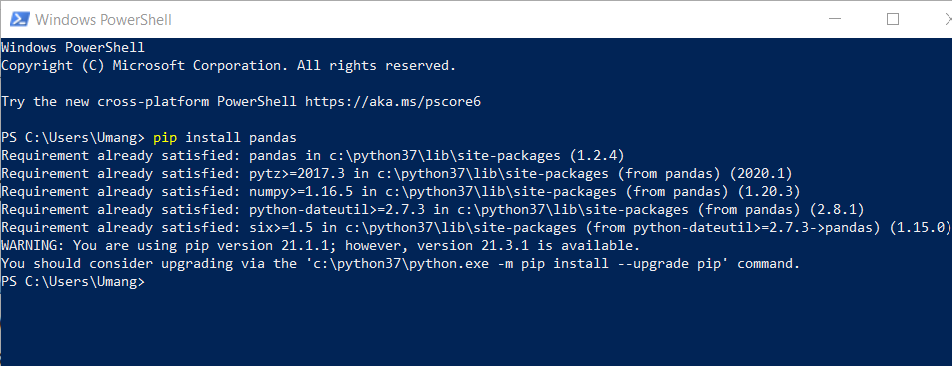
### **Commandes de base en python que vous devez connaître**

Voici la liste des commandes de python de base.

#### **Commande d'installation de pip**

pip est un gestionnaire de paquets qui est écrit en [**langue Python**](https://www.scaler.com/topics/python/). Il est utilisé pour installer et gérer des progiciels. La commande d'installation pip est utilisée pour installer n'importe quel logiciel à partir d'un dépôt en ligne de paquets publics, appelé Python Package Index. Pour exécuter cette commande dans les fenêtres, vous devez ouvrir votre Windows PowerShell, puis utiliser la syntaxe suivante pour installer n'importe quel paquet.

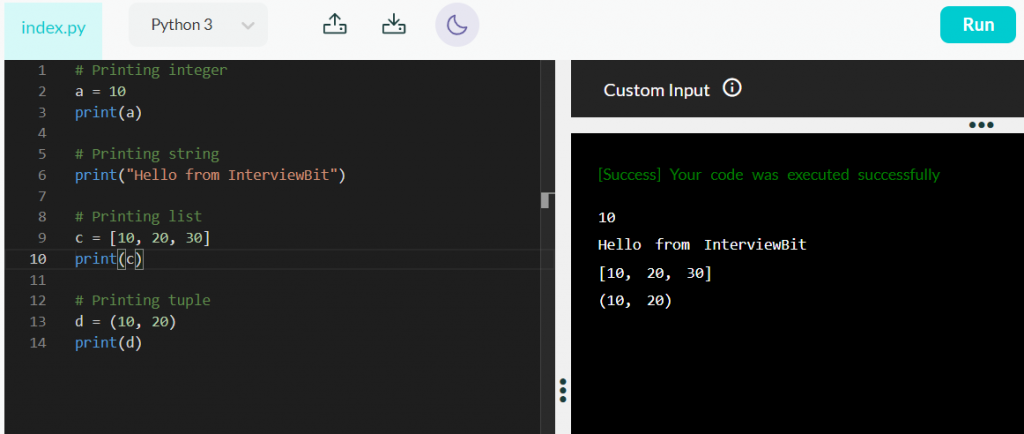
**Syntaxe:** pip install package-name  
**Exemple:**

****

#### **Commande d'impression**

Une commande d'impression est utilisée pour imprimer un message sur l'écran ou un autre dispositif de sortie standard. Le message peut être une chaîne ou tout autre objet. La commande d'impression peut être utilisée pour imprimer n'importe quel type d'objet comme entier, chaîne, liste, tuple, etc.

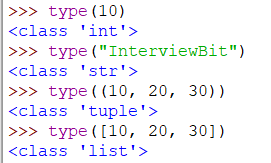
**Syntaxe:** impression (objet)  
**Exemple:**

****

#### **Commande de type**

Une commande de type est utilisée pour vérifier le type ou la classe d'un objet.

**Syntaxe:** type(object)  
**Exemple:**

****

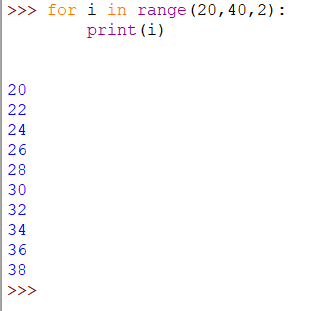
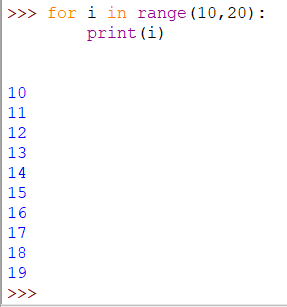
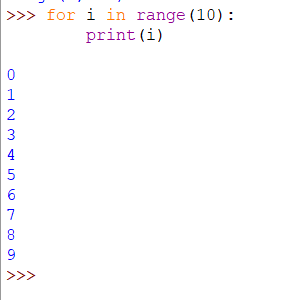
#### **Commande de plage**

La commande de distance est utilisée pour générer une séquence d'entiers commençant de 0 par défaut à n où n n n n n n est inclus dans les nombres générés. Nous utilisons cette commande pour la plupart des boucles.

**Syntaxe:** plage (démarrage, arrêt, étape)

* *début* se réfère au début de la plage (optionnelle; 0 par défaut)
* *stop* se réfère au numéro avant que vous voulez arrêter (obsent)
* *l'étape* fait référence au comptage de l'incrémentation (optionnel; 1 par défaut)

**Exemple:**

****

**Remarque:** Si vous fournissez deux paramètres à la fonction distance(), il les considérera toujours comme (démarrage, arrêt) et non comme (stop, étape).

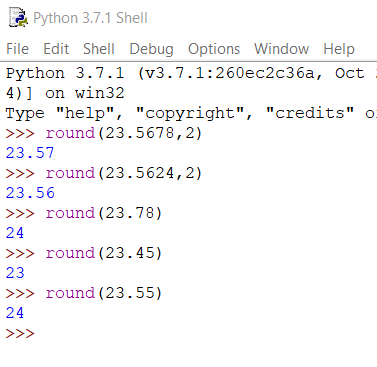
#### **Commande du tour**

une commande de tour est utilisée pour arrondir un nombre à une précision donnée en chiffres décimaux. Cela signifie que si vous avez tellement de chiffres après vicimal dans un nombre à virgule flottante, vous pouvez utiliser la commande ronde pour compléter le nombre spécifié. Vous pouvez mentionner combien de chiffres vous voulez après la virgule.

**Syntaxe:** round (numéro, chiffres)

* *Le nombre* fait référence au numéro de virgule flottant.
* *les chiffres* se réfèrent au comptage des chiffres que vous voulez après la virgule. (Facultatif; par défaut 0)

**Exemple:**

****

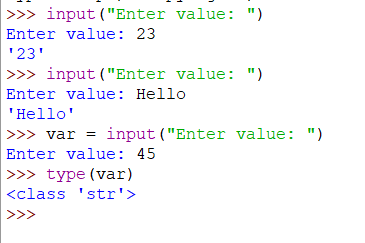
#### **Commande d'entrée**

une commande d'entrée est utilisée pour prendre entrée de l'utilisateur. Le flux du programme sera arrêté jusqu'à ce que l'utilisateur ait entré n'importe quelle valeur. Quel que soit l'utilisateur qui l'entrera, il sera converti en une chaîne par la fonction d'entrée. Si vous voulez prendre un entier comme entrée, alors vous devez le convertir explicitement.

**Syntaxe:** entrée (message)

*Un message* fait référence au texte que vous souhaitez afficher à l'utilisateur. (Facultatif)

**Exemple:**

****

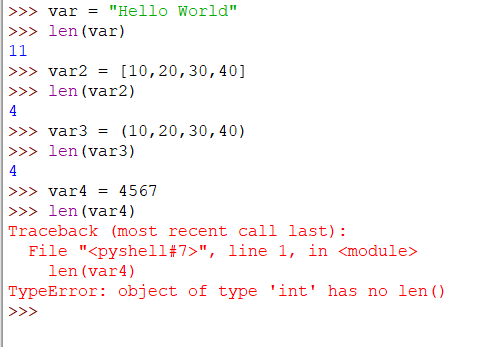
#### **Len commandement**

La commande len ou len() est utilisée pour obtenir le nombre d'éléments dans un objet. Si l'objet est une chaîne alors la fonction len() retourne le nombre de caractères qu'elle contient. Si l'objet est une liste ou un tuple, il retournera le nombre d'éléments présents dans cette liste ou tuple. len() donne une erreur si vous essayez de lui faire passer une valeur entière.

**Syntaxe:** len(object)

*objet* dont vous voulez trouver la longueur (Requis)

**Exemple:**

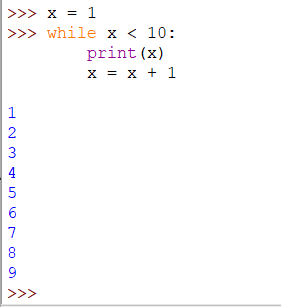
****

#### **Commandes en boucle**

En python, nous avons deux commandes de boucle primitives à savoir pendant et pour. La commande *en* boucle *est* utilisée pour exécuter un ensemble d'instructions tant que la condition donnée est vraie.

**Syntaxe de la boucle:** en tout état:  
 Déclarations  
 mettre à jour l'itinéraire

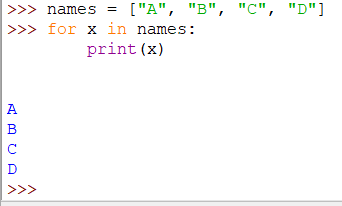
**Exemple:**

****

L'ordre de boucle est utilisé pour exécuter un ensemble d'instructions en itérant sur une séquence. Cette séquence peut être une liste, tuple, chaîne, dictionnaire, etc.

**Syntaxe pour la boucle:** pour x en séquence:  
 Déclarations

**Exemple:**

****

### **Commandes intermédiaires en Python**

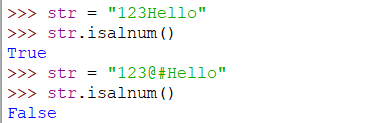
#### **Commandes de chaîne**

Dans le langage de programmation python, nous avons différentes commandes de chaîne que nous pouvons utiliser sur des objets de chaîne. Ces commandes ne changent pas l'objet de chaîne d'origine, elles reviennent simplement un nouvel objet. Certaines des fonctions de cordes les plus importantes sont:

isalnum(): Il vérifie si tous les caractères d'une chaîne donnée sont alphanumériques ou non. Il retourne une valeur booléenne.

**Syntaxe:** stringname.isalnum()

**Exemple:**

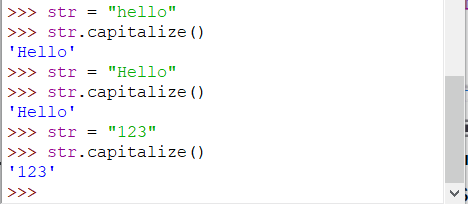
****

#### **capitaliser()**

la fonction capitaliser() change le premier caractère de la chaîne en majuscules si elle est minuscule. Si le premier caractère est une majuscule ou un entier ou un caractère spécial, il ne fait rien.

**Syntaxe:** stringname. capitalize()

**Exemple:**

****

#### **find()**

find() commande est utilisée pour rechercher une sous-chaîne dans une chaîne. Il retourne l'indice de la première occurrence de la sous-corre si elle est présente sinon il retourne -1.

**Syntaxe:** string.find(substring)

* *string* se réfère à la chaîne dans laquelle vous voulez rechercher.
* *substring* fait référence à la valeur que vous voulez rechercher.

**Exemple:**

****

1. count(): la fonction count() retourne le nombre d'occurrences d'une sous-corre dans un objet de chaîne.

**Syntaxe:** stringname.count (sous-corde, début, fin)

* *stringname* fait référence à la chaîne dans laquelle vous souhaitez rechercher.
* *substring* fait référence à la valeur dont vous voulez trouver le comptage.
* *début* se réfère à cet indice de départ dans la chaîne où la recherche démarre (optionnelle)
* *end* fait référence à cet indice de fin dans la chaîne où la recherche se termine (facultatif)

**Exemple:**

****

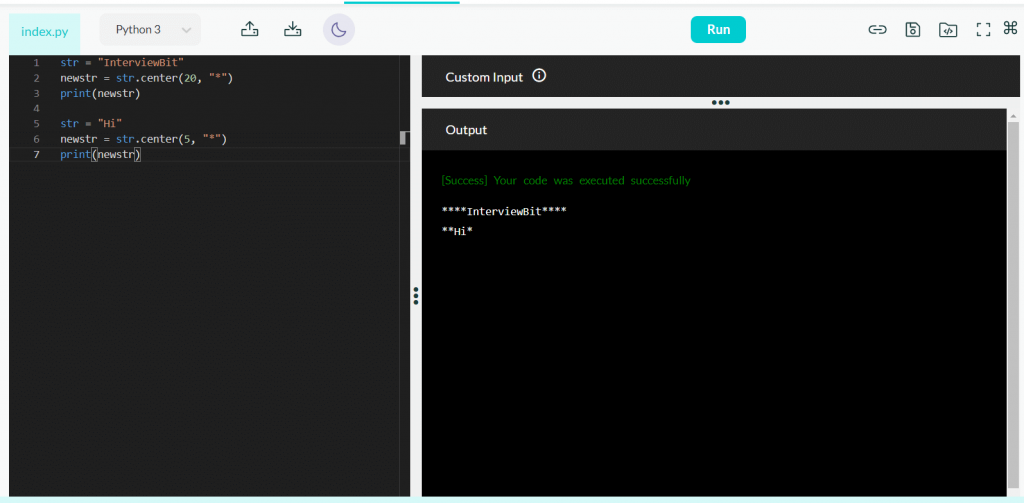
#### **centre()**

Une commande centrale est utilisée pour aligner une chaîne au centre, en utilisant un caractère spécifié (par défaut, il est de l'espace) comme caractère de remplissage.

**Syntaxe:** string.center (longueur, caractère)

* *string* est la chaîne que vous voulez aligner au centre
* *la longueur* est la longueur entière prise par la nouvelle corde dans laquelle les deux côtés seront remplis par *caractère* et au centre, nous aurons la corde d'origine.
* *Le caractère* fait référence au caractère utilisé pour remplir l'espace manquant de chaque côté. Par défaut est « » (espace).

**Exemple:**

****

#### **Commandes de la liste**

Tout comme une chaîne, nous avons différentes commandes pour les objets de liste aussi bien. Les listes sont utilisées pour plusieurs éléments dans un seul objet. Nous pouvons utiliser des listes pour stocker des éléments de différents types de données. Parmi les méthodes de liste les plus importantes, on peut citer:

append(): la commande d'ajout est utilisée pour ajouter un élément à la fin de la liste.

**Syntaxe:** list.append(élément)

* *list* est l'objet de liste dans lequel vous voulez ajouter un élément
* *élément* fait référence au nouveau point que vous souhaitez ajouter à la liste

**Exemple:**

****

#### **Copie()**

copy() est utilisé pour créer une nouvelle copie de l'objet de liste. Il retourne un nouvel objet de liste.

**Syntaxe:** list.copy()

**Exemple:**

****

#### **insérer()**

Une commande d'insertion est utilisée pour ajouter un élément à une position spécifiée dans l'objet de liste. Il prend deux paramètres position et élément.

**Syntaxe:** listname.insert (position, élément)

* *position* à laquelle vous souhaitez insérer un nouvel élément. Si vous donnez une position supérieure au nombre d'éléments de la liste, il s'insera toujours à la fin.
* *L'élément* fait référence au nouvel élément qui doit être ajouté.

**Exemple:**

****

#### **pop()**

La méthode pop() est utilisée pour supprimer un élément d'une position spécifiée dans la liste. Il retourne l'élément après l'avoir retiré de la liste.

**Syntaxe:** listname.pop(position)

*position* d'où vous voulez enlever l'élément.

**Exemple:**

****

#### **inverse()**

la méthode inverse inverse l'ordre de tous les éléments de la liste. Il modifie l'objet de la liste d'origine et il ne renvoie rien.

**Syntaxe:** list.reverse()

**Exemple:**

****

#### **tri()**

Une méthode de triage est utilisée pour trier les éléments de la liste dans l'ordre croissant par défaut.

**Syntaxe:** list.sort()

**Exemple:**

****

#### **Commandes de tuples**

Tuple est un type de données intégré qui est utilisé pour stocker plusieurs éléments dans une seule variable. Les objets tubulaires sont ordonnés et immuables. Il n'y a que deux méthodes de tuple intégrées qui sont les suivantes:

count(): la méthode de comptage est utilisée pour compter les occurrences d'un élément dans la tuple.

**Syntaxe:** tuple.count(élément)

**Exemple:**

****

#### **Indice()**

Cette méthode est utilisée pour trouver l'indice de la première occurrence d'un élément. Si l'élément n'est pas trouvé dans le tuple entier, il augmentera une ValueError.

**Syntaxe:** tuple.index(élément)

* *tuple* est l'objet tuple dans lequel vous voulez chercher un élément
* *L'élément* fait référence à l'élément que vous souhaitez rechercher

**Exemple:**

****

### **Liste avancée des commandes en Python**

#### **Commandes de réglage**

Des ensembles sont également utilisés pour stocker plusieurs éléments dans un seul objet, mais ils ne permettent pas d'éléments dupliqués. Les jeux ne sont pas ordonnés et non indexés. Cela signifie que si vous imprimez tous les éléments d'un ensemble, ils seront imprimés dans l'ordre aléatoire. Une fois l'ensemble créé, vous ne pouvez pas modifier les éléments, mais vous pouvez ajouter de nouveaux éléments ou supprimer des éléments existants. Maintenant, discutons de certaines commandes de configuration importantes que le python fournit en tant que méthodes intégrées.

add(): ajouter la commande est utilisée pour ajouter un nouvel élément dans l'ensemble.

**Syntaxe:** setname.add(élément)

* *setname* est le nom de cette variable d'ensemble dans laquelle vous voulez ajouter un nouvel élément.
* *L'élément* fait référence au nouveau point que vous souhaitez ajouter.

**Exemple:**

****

#### **clair()**

l'éclaire élimine tous les éléments d'un ensemble. Il ne prend aucun paramètre.

**Syntaxe:** setname.clear()

**Exemple:**

****

#### **Jetez()**

Le jet est utilisé pour retirer l'élément spécifié de l'ensemble. Si l'élément spécifié n'est pas trouvé dans l'ensemble, il ne donnera pas d'erreur.

**Syntaxe:** setname.discard(élément)

* *setname* est le nom de cette variable d'ensemble à partir de laquelle vous voulez supprimer un élément.
* *L'élément* fait référence à l'élément que vous souhaitez supprimer.

**Exemple:**

****

#### **enlever()**

l'ordre de suppression est également utilisé pour supprimer un élément spécifié de l'ensemble, mais elle est différente de l'élimination car supprimer donnera une erreur si l'élément spécifié n'est pas trouvé dans l'ensemble.

**Syntaxe:** setname.remove(élément)

* *setname* est le nom de cette variable d'ensemble à partir de laquelle vous voulez supprimer un élément.
* *L'élément* fait référence à l'élément que vous souhaitez supprimer.

**Exemple:**

****

#### **différence()**

La méthode de différence est utilisée pour obtenir un ensemble qui contient la différence de deux ensembles. La différence d'ensembles signifie qu'il n'aura que les éléments qui ne sont présents que dans un ensemble et non dans un autre ensemble. Supposons que nous ayons deux ensembles A et B. L'ensemble A a un -12,3 % et le jeu B a un '2, 4, 6'. Alors la différence de A et B sera de -1,3".

**Syntaxe:** setA.différence(setB)

* Les éléments de setB seront retirés de setA s'ils existent.

**Exemple:**

****

#### **Différence-mise à jour()**

la méthode de différence-update est utilisée pour obtenir un ensemble d'éléments présents dans le premier ensemble et non communs dans les deux ensembles. Cela signifie qu'il enlève les éléments qui existent dans les deux ensembles. Il ne renvoie pas un nouvel ensemble, il vient simplement d'enlever des éléments communs du premier ensemble.

**Syntaxe:** setA.diffence-update(setB)

* Les éléments qui existent à la fois dans setA et setB seront retirés de setA.

**Exemple:**

****

#### **intersection()**

la méthode d'intersection retourne un ensemble ayant des éléments qui existent dans tous les ensembles spécifiés.



**Syntaxe:** set.intersection(set1, set2, ... setn)

* L'ensemble d'éléments qui existent dans l'ensemble, set1, set2, ... setn sera retourné.

**Exemple:**

****

#### **issubset()**

Une méthode issubset vérifie si tous les éléments de l'ensemble A sont présents dans l'ensemble B ou non. Il retourne une valeur booléenne.

**Syntaxe:** setA.issubset(setB)

**Exemple:**

****

#### **symétriques ()**

Cette méthode retourne un ensemble contenant des éléments des deux ensembles sauf ceux qui sont communs dans les deux ensembles.



**Syntaxe:** setA.symmetric-diffence(setB)

**Exemple:**

****

#### **Syndicat()**

La méthode de l'union retourne un ensemble contenant tous les éléments des deux ensembles, à l'exception des ensembles dupliqués.

**Syntaxe:** setA.union(setB)

**Exemple:**

****

#### **Commandes du dictionnaire**

Le dictionnaire est un type intégré en Python. Il est utilisé pour stocker des paires de valeur-clés. Il est ordonné, modifiable, et ne permet pas les clés dupliquées. Cela signifie que dans un dictionnaire nous ne pouvons pas ajouter deux paires ayant la même valeur de clés. Python fournit un ensemble de méthodes intégrées que nous pouvons utiliser sur les objets de dictionnaire.

fromkeys(): la méthode fromkeys() est utilisée pour générer un dictionnaire avec des clés spécifiées et une valeur spécifiée.

**Syntaxe:** dict.fromkeys (clés, valeur)

* *les clés* sont le tuple ou la liste des éléments clés.
* *la valeur* se réfère à la valeur qui serait appariée avec toutes les clés spécifiées.

**Exemple:**

****

#### **get()**

get method est utilisé pour obtenir une valeur de la clé spécifiée. Si une clé n'est pas trouvée dans le dictionnaire, elle ne reviendra rien à moins que nous ne précisions quelque chose dans les paramètres.

**Syntaxe:** dictionnaire.get (clé, valeur)

* *dictionnaire* est le nom de l'objet du dictionnaire dans lequel vous voulez rechercher.
* *key* fait référence à la clé que vous souhaitez rechercher dans le dictionnaire.
* valeur est la valeur qui serait retournée si la clé ne se trouve pas dans le dictionnaire.

**Exemple:**

****

#### **points()**

La méthode des éléments est utilisée pour afficher les éléments du dictionnaire. Il affichera toutes les paires clé-valeur présentes dans le dictionnaire. Il retourne un objet de visualisation qui contiendra toutes les paires clé-valeur en tant que tuples dans une liste. Il ne prend aucun paramètre.

**Syntaxe:** dictionnaire.items()

**Exemple:**

****

#### **clés()**

la méthode des clés est utilisée pour obtenir toutes les clés présentes dans le dictionnaire. Il retourne un objet de vue contenant toutes les clés du dictionnaire sous forme de liste. Il ne prend aucun paramètre.

**Syntaxe:** dictionnaire.keys()

**Exemple:**

****

#### **valeurs()**

Une méthode de valeurs est utilisée pour obtenir toutes les valeurs présentes dans le dictionnaire. Il retourne un objet de visualisation contenant toutes les valeurs du dictionnaire sous forme de liste. Il ne prend aucun paramètre.

**Syntaxe:** dictionnaire.valeurs()

**Exemple:**

****

#### **pop()**

La méthode pop est utilisée pour supprimer une paire clé-valeur du dictionnaire en spécifiant la clé. Il retourne la valeur de la paire de valeur clé qui doit être supprimée.

**Syntaxe:** dictionnaire.pop(key)

* *key* fait référence à la clé de la paire que vous souhaitez retirer du dictionnaire.

**Exemple:**

****

#### **popitem()**

La commande popitem est utilisée pour supprimer la dernière paire insérée du dictionnaire. Il ne prend aucun paramètre. Il retourne la paire enlevée comme un tuple.

**Syntaxe:** dictionnaire.popitem()

**Exemple:**

****

#### **par défaut()**

la méthode setdefault est utilisée pour obtenir la valeur d'une clé spécifiée. Si la clé n'existe pas, elle insèrera la clé avec la valeur passée comme paramètre. Si vous ne spécifiez aucune valeur, il insérera la clé avec la valeur None.

**Syntaxe:** dictionnaire.setdefault(clé, valeur)

**Exemple:**

****

## **Conclusion**

Dans ce billet, nous avons discuté des commandes de python les plus grandes que chaque programmeur Python devrait apprendre. Vous devriez certainement essayer chaque commande par vous-même. Essayez également d'expérimenter des paramètres d'entrée aléatoires pour voir le comportement des commandes. Les commandes Python sont faciles à utiliser, faciles à écrire et faciles à apprendre. Vous n'avez pas besoin de vous souvenir de toutes les commandes, mais vous devez être familier avec ce que toutes les commandes fournissent comme fonctionnalité.

## **Foire aux questions**

**Q. Qu'est-ce que l'ordre de Python 3 ?**A. Les commandes Python 3 sont publiées en 2008. Ces commandes sont introduites avec des caractéristiques Python 2 et elles sont également compatibles avec Python 2. Les commandes Python 3 sont plus intuitives pour les programmeurs et plus précises tout en fournissant le résultat.

**Q. Comment utiliser les commandes Python?**A. Pour utiliser ou exécuter des commandes Python, vous pouvez utiliser le shell de python. Python Shell est l'interpréteur python qui est utilisé pour exécuter des programmes Python simples et pour exécuter des commandes de python. Il fournit simplement un moyen rapide d'exécuter des commandes sans créer de fichier.

**Q. Quelles sont les commandes Python courantes ?**A. Certaines commandes Python communes sont entrées, impression, portée, round, pip install, len, tri, commandes en boucle comme et pendant et pendant et ainsi de suite.

**Q. Qu'est-ce que les commandes magiques en Python ?**A. Les commandes magiques sont des raccourcis ou des améliorations sur la syntaxe Python habituelle. Ces commandes sont conçues pour faciliter les tâches de routine. Ils nous permettent de contrôler facilement le comportement du système IPython et de résoudre divers problèmes communs dans l'analyse des données standard, par example l'exécution d'un script externe ou le calcul du temps d'exécution d'un morceau de code.

## **Ressources supplémentaires**

[Cours gratuit de Python](https://www.scaler.com/topics/course/python-for-beginners/)[Codage pratique](https://www.interviewbit.com/practice/)[Python MCQ](https://www.interviewbit.com/python-mcq/)[Questions d'entretiens de python](https://www.interviewbit.com/python-interview-questions/)[PHP Vs Python](https://www.interviewbit.com/blog/php-vs-python/)[Développeur de Python Reprendre](https://www.interviewbit.com/blog/python-developer-resume/)[Demandes en Python](https://www.interviewbit.com/blog/applications-of-python/)[Compétences des développeurs de Python](https://www.interviewbit.com/blog/python-developer-skills/)

* [Python](https://www.interviewbit.com/blog/tag/python/)
* [Commandes Python](https://www.interviewbit.com/blog/tag/python-commands/)

##### **Article précédent**

****

* [Commandes](https://www.interviewbit.com/blog/category/commands/)

## [**Top PostgreSQL Commandes que vous devez connaître**](https://www.interviewbit.com/blog/postgresql-commands/)

8 janvier 2024

##### **Prochain billet**

****

* [Commandes](https://www.interviewbit.com/blog/category/commands/)

## [**Commandes de haut Docker avec des exemples**](https://www.interviewbit.com/blog/docker-commands/)

8 janvier 2024

##### **Catégories**

* [Demandes](https://www.interviewbit.com/blog/category/applications/)
* [Architecture](https://www.interviewbit.com/blog/category/architecture/)
* [Livres](https://www.interviewbit.com/blog/category/books/)
* [Carrières](https://www.interviewbit.com/blog/category/careers/)
* [Caractéristiques](https://www.interviewbit.com/blog/category/characteristics/)
* [Problèmes de codage](https://www.interviewbit.com/blog/category/coding-problems/)
* [Commandes](https://www.interviewbit.com/blog/category/commands/)
* [Comparer](https://www.interviewbit.com/blog/category/compare/)
* [Composants](https://www.interviewbit.com/blog/category/components/)
* [Cours](https://www.interviewbit.com/blog/category/courses/)
* [Caractéristiques](https://www.interviewbit.com/blog/category/features/)
* [Cadres](https://www.interviewbit.com/blog/category/frameworks/)
* [IDE](https://www.interviewbit.com/blog/category/ide/)
* [Question d'interview](https://www.interviewbit.com/blog/category/interview-question/)
* [Entreprises de technologies de](https://www.interviewbit.com/blog/category/it-companies/)
* [Rôles des emplois](https://www.interviewbit.com/blog/category/job-roles/)
* [Bibliothèques](https://www.interviewbit.com/blog/category/libraries/)
* [Méthodes](https://www.interviewbit.com/blog/category/methodologies/)
* [Modèle](https://www.interviewbit.com/blog/category/model/)
* [Principes](https://www.interviewbit.com/blog/category/principles/)
* [Projets](https://www.interviewbit.com/blog/category/projects/)
* [Reprise](https://www.interviewbit.com/blog/category/resume/)
* [Salaires](https://www.interviewbit.com/blog/category/salaries/)
* [Compétences](https://www.interviewbit.com/blog/category/skills/)
* [Technologies](https://www.interviewbit.com/blog/category/technologies/)
* [Outils](https://www.interviewbit.com/blog/category/tools/)
* [Types](https://www.interviewbit.com/blog/category/types/)

* [Commandes](https://www.interviewbit.com/blog/category/commands/)

7 minutes lire

# **Commandes de haut Docker avec des exemples**

8 janvier 2024



Table des matières

montrer

* [Introduction](https://www.interviewbit.com/blog/python-commands/#introduction)
* [Qu'est-ce que Docker?](https://www.interviewbit.com/blog/python-commands/#what-is-docker)
* [Commandes de haut de poupe](https://www.interviewbit.com/blog/python-commands/#top-docker-commands-)
  + [1. docker -version](https://www.interviewbit.com/blog/python-commands/#1-docker---version)
  + [2. recherche docker](https://www.interviewbit.com/blog/python-commands/#2-docker-search)
  + [3. pull docker](https://www.interviewbit.com/blog/python-commands/#3-docker-pull)
  + [4. images docker](https://www.interviewbit.com/blog/python-commands/#4-docker-images)
  + [5. course de docker](https://www.interviewbit.com/blog/python-commands/#5-docker-run)
  + [6. docker exec](https://www.interviewbit.com/blog/python-commands/#6-docker-exec)
  + [7. Arrêt du docker](https://www.interviewbit.com/blog/python-commands/#7-docker-stop)
  + [8. Redémarrage du docker](https://www.interviewbit.com/blog/python-commands/#8-docker-restart)
  + [9. journaux de docker](https://www.interviewbit.com/blog/python-commands/#9-docker-logs)
  + [10. docker rm](https://www.interviewbit.com/blog/python-commands/#10-docker-rm)
* [Conclusion](https://www.interviewbit.com/blog/python-commands/#conclusion)
* [Foire aux questions](https://www.interviewbit.com/blog/python-commands/#frequently-asked-questions)
  + [Q.1 : Combien de commandes y a-t-il à Docker ?](https://www.interviewbit.com/blog/python-commands/#q1-how-many-commands-are-there-in-docker)
  + [Q.2: Docker est-il gratuit?](https://www.interviewbit.com/blog/python-commands/#q2-is-docker-free-to-use)
  + [Q.3 : Docker est-il facile à apprendre ?](https://www.interviewbit.com/blog/python-commands/#q3-is-docker-easy-to-learn)

## **Introduction**

Docker est une plate-forme de conteneurisation à base de Linux open source qui permet aux développeurs de regrouper leurs applications dans des conteneurs, les conteneurs sont les composants exécutables qui combinent le code source de l'application avec les bibliothèques de systèmes d'exploitation et les dépendances nécessaires pour exécuter le code dans n'importe quel environnement. L'avantage est que les développeurs n'ont pas à craindre que leur code fonctionne de la même manière qu'il fonctionne sur leur machine. Docker gagne en popularité au fil des ans et est aujourd'hui un savoir-faire indispensable. Ce blog vous apprendra les commandes de docker qui sont fréquemment utilisées. Avant de passer aux commandes docker et à la liste des commandes docker, il est essentiel d'avoir une bonne compréhension de ce qu'est le docker.

## **Qu'est-ce que Docker?**

**Docker**

Lorsqu'ils travaillent sur un grand projet, plusieurs développeurs collaborent. Plusieurs équipes sont constituées pour construire un projet. À grande échelle, nous pouvons considérer qu'il existe une équipe de développement qui développera les applications, une équipe de test qui testera les applications et une équipe d'exploitation qui déploiera l'application sur le serveur de production. Le processus varie d'une entreprise à l'autre.

En tant que développeur, vous pourriez utiliser certains cadres, comme le cadre de printemps ou le cadre Django. Ce cadre a besoin de certaines dépendances, les dépendances varient à leur tour d'un système d'exploitation à l'autre et d'une version à l'autre. Cela signifie qu'un code qui a été compilé avec succès dans la version 1.0.1 pourrait ne pas être exécuté le même dans la version 1.0.2. Il s'agit d'une question qui pose problème. La première solution consiste à donner toutes les dépendances et spécifications dans un fichier zip ou un fichier de pot pendant qu'il l'expédie à une autre équipe. Le problème ici est que certaines dépendances dépendent du système d'exploitation. Vous ne pouvez pas donner l'ensemble de votre système d'exploitation à une autre personne.

### **Confus à propos de ton prochain emploi ?**

En 4 étapes simples, vous pouvez trouver votre feuille de route de carrière personnalisée dans le développement de logiciels pour FREE

[Étendre dans le nouvel onglet](https://www.scaler.com/career-plan/embedded?utm_source=ib&utm_medium=blog-embed&utm_campaign=career-tool)

La solution utilise un Hypervisor. Un hyperviseur, également connu sous le nom de moniteur de machine virtuelle ou VMM, est un logiciel qui crée et exécute des machines virtuelles (VM). Un hyperviseur permet à un ordinateur hôte de prendre en charge plusieurs VM invitées en partageant virtuellement ses ressources entre les machines virtuelles, telles que la mémoire et le traitement. Pensez à un hyperviseur comme suit: vous avez votre matériel informatique et en plus il y a un système d'exploitation, vous pouvez installer un hyperviseur dans votre système et vous pouvez exécuter un système d'exploitation invité. La meilleure partie est que vous pouvez facilement expédier votre système d'exploitation en créant son image virtuelle pour n'importe quelle autre personne. Une fois que l'autre personne a reçu une image, elle peut créer son instance et la faire fonctionner. C'est le concept de virtualisation.

Cependant, la création d'une nouvelle machine virtuelle chaque fois pour une nouvelle application devient lourde. De là, le concept de conteneurisation vient en ligne de compte. La conteneurisation est l'évolution de la virtualisation. Alors que la virtualisation se concentre sur la distribution des systèmes d'exploitation, la conteneurisation, d'autre part, se concentre sur la décomposition des systèmes d'exploitation en morceaux qui peuvent être utilisés plus efficacement.

En bref, **les conteneurs sont une solution au problème de savoir comment faire fonctionner les logiciels de manière fiable lorsqu'ils sont déplacés d'un environnement informatique à un autre.**

## **Commandes de haut de poupe**

Avant de passer à la liste des commandes docker, il est essentiel que vous ayez installé dans vos systèmes. Si vous l'avez déjà installé, vous pouvez aller de l'avant. Si vous ne disposez pas d'un docker, vous pouvez facilement le faire, en suivant les étapes mentionnées dans la documentation.

Dans le cas où vous commencez et que vous ne voulez pas mettre en place docker, vous pouvez utiliser le terrain de jeu en ligne, Play with Docker, Son projet sponsorisé par docker pour permettre aux nouveaux apprenants de commencer facilement avec docker. Il y a un environnement de laboratoire où vous obtiendrez un terminal en ligne pour exécuter des commandes docker gratuites. C'est simple et facile à utiliser.

Voyons les différentes commandes de docker



### **1. docker -version**

Cette commande rend par défaut les informations de version dans une mise en page facile à lire, Vous pouvez spécifier votre propre format aussi.



La commande peut être utilisée de diverses manières. Quelques-unes des modalités indiquées ci-après

* Pour obtenir la version du serveur, utilisez la commande suivante

**Version docker –format «.Server.Version»**

****

* Pour bourler les données brutes de json, utiliser la commande suivante

**Version docker –format «json .»**

****

### **2. recherche docker**

Cette commande est utilisée pour rechercher le centre de docker pour des images publiques. Les informations renvoyées contiennent le nom de l'image, la description, les étoiles, automatisés et officiels ainsi que de nombreux autres détails.

**Jeu de dockers search'''**

La commande docker recherche MySQL est un exemple où nous recherchons des images MySQL, la commande retournera les images disponibles de MySQL sur le hub docker comme indiqué ci-dessous.



Vous pouvez modifier la commande pour afficher une description non tronquée en utilisant - pas de tronc comme indiqué ci-dessous:

**docker search –filter-stars-3 – no-trunc MySQL**

Par exemple, la commande affichera les images de MySQL avec au moins 3 étoiles avec une description non tronquée.



Lorsque vous utilisez la commande de recherche, par défaut 25 images sont renvoyées. Vous pouvez limiter le résultat renvoyé par une recherche utilisant le drapeau **limite**. Les valeurs pourraient être comprises entre 1 et 100.

Pour filtrer les images de MySQL sur la base de constructions automatisées, utilisez la commande suivante.

**docker search –filter is-automatisé-true MySQL**

****

### **3. pull docker**

Cette commande est utilisée pour extraire une image ou un dépôt distant d'un registre. Il y a beaucoup d'images pré-construites que vous pouvez tirer pour utiliser sans avoir besoin de définir et de configurer les vôtres. La commande ci-dessous tire l'image debian de dockerhub.

**docker pull debian**

****

Comme vous pouvez le remarquer à partir de l'image ci-dessus, le moteur docker utilise l'étiquette la plus récente par défaut. Si vous avez besoin de tirer une version spécifique de l'image, vous pouvez la spécifier explicitement, la commande ci-dessous tire la dernière version de l'image Ubuntu 14.04 par défaut.

**docker pull ubuntu:20.04**

****

Comme vous pouvez le remarquer à partir de la sortie, le docker imprime un condensé de l'image une fois l'attraction terminée. Le digest de l'image dans l'exemple ci-dessus est

**sha256:b5a61709a9a44284d88fb12e5c48db0409cfd5b69d4ff8224077c57302df9cf**

Mais pourquoi avons-nous besoin d'un condensé ?

Un condensé est nécessaire car il remplace une étiquette lorsqu'une image est tirée. Ainsi, au lieu de spécifier la version, vous pouvez spécifier son résumé. Tirons l'image ci-dessus à l'aide de son hachage. L'ordre sera:

**docker pull 'digest'**Allez-y et essayez-les par vous-même.

La commande docker pull tire une image unique du registre, pour tirer toutes les images d'un référentiel, vous pouvez utiliser l'étiquette -a lors de l'utilisation de la commande docker pull. L'exemple ci-dessous illustre la même chose.

**docker pull -a ubuntu**

****

### **4. images docker**

Cette commande est utilisée pour répertorier toutes les images qui sont actuellement présentes dans votre hub docker. Cette commande montrera toutes les images de premier niveau, leurs référentiels et étiquettes et leur taille, vous pouvez vous référer à la capture d'écran ci-dessous.



La commande prend un argument optionnel [REPOSITORY[:TAG) qui limitera la liste à des images qui correspondent à l'argument. Pour répertorier toutes les images dans le dépôt java, vous pouvez utiliser la commande.



### **5. course de docker**

Cette commande est utilisée pour créer un conteneur à partir d'une image ou pour exécuter une commande dans un nouveau conteneur. La commande crée d'abord une couche de conteneur inscriptible sur l'image spécifiée, puis elle a commencé à utiliser une commande spécifique.

La commande ci-dessous est utilisée pour créer un conteneur pour l'image ubuntu.

**docker run -it -d ubuntu**

****

Vous pouvez utiliser la commande docker ps pour visualiser les conteneurs actifs comme indiqué ci-dessous.



Vous pouvez vous référer à la [documentation officielle](https://docs.docker.com/engine/reference/commandline/run/) pour plus de détails.

### **6. docker exec**

Cette commande est utilisée pour accéder au conteneur en cours d'exécution et exécuter la commande dans un conteneur en marche. Vous pouvez accéder à n'importe quel conteneur à l'aide de son id. L'identifiant du conteneur peut être visualisé à l'aide de la commande d'images docker comme indiqué ci-dessous.



La syntaxe de la commande docker exec est  
**docker exec -it bash**

Pour exécuter la commande ci-dessus, vous devez d'abord démarrer un conteneur en utilisant la commande suivante  
**docker run -name ubuntu'bash -rm -i -t ubuntu bash**

Un conteneur nommé ubuntu-bash sera créé et une session bash est commencée.



Maintenant vous pouvez exécuter une commande sur le conteneur  
**docker exec -d ubuntu-bash touch /tmp/execWorks**

Un nouveau fichier nommé /tmp/execWorks sera créé à l'intérieur du conteneur en cours d'exécution ubuntu'bash en arrière-plan.

### **7. Arrêt du docker**

Comme son nom l'indique, cette commande crée un ou plusieurs conteneurs en cours d'exécution.  
**docker stop (conteneur id)**

### **8. Redémarrage du docker**

Vous pouvez recommencer à l'aide du conteneur arrêté par le récipient de redémarrage.  
**Redémarrage docker**

### **9. journaux de docker**

Cette commande est utilisée pour récupérer/extraire les logs présents au moment de l'exécution d'une commande docker. La commande est très utile lors du déboguement de vos conteneurs dockers.  
**logs dockers (nom de conteneur)**

### **10. docker rm**

La commande est utilisée pour enlever un ou plusieurs conteneurs. La commande ci-dessous enlèvera le conteneur

docker rm <container name>

Jusqu'à présent, vous avez maintenant une bonne compréhension des commandes de base docker et savoir comment effectuer des opérations de base à l'aide du docker.

## **Conclusion**

Dans ce guide, nous avons appris les commandes de base du docker. Docker est une plate-forme de conteneurisation open source, basée sur Linux, qui est utilisée par les développeurs pour construire, exécuter et regrouper leurs applications pour des conteneurs. Il offre une abstraction de niveau du système d'exploitation avec une utilisation minimale des ressources.

Connaître le docker est une compétence indispensable, que vous soyez un développeur, un testeur ou un ingénieur DevOps. J'espère que cet article vous a aidés. Si vous vous préparez pour vos interviews comme plus frais ou si vous êtes expérimenté et cherchez à changer d'emploi, alors Interview Bit est le bon endroit pour commencer. Il a plusieurs pistes, dont la programmation, [la conception de systèmes](https://www.scaler.com/courses/system-design/?utm_source=ib&utm_medium=blog-inlink&utm_campaign=ib-inlinks-auto), les puzzles et le script, ainsi que des guides de préparation spécifiques à l'entreprise et des cours rapides. Consultez InterviewBit pour vous perfectionner.

## **Foire aux questions**

### **Q.1 : Combien de commandes y a-t-il à Docker ?**

**Ans :** Il y a 13 commandes de gestion et 41 commandes générales en docker.

### **Q.2: Docker est-il gratuit?**

**Ans:** Docker Desktop reste gratuit pour les petites entreprises (moins de 250 employés ET moins de 10 millions de dollars de revenus annuels), l'usage personnel, l'éducation et des projets à source ouverte non commerciaux. Elle nécessite un abonnement payant (Pro, équipe ou Business), pour un nombre n'est que de 5 dollars par mois, pour un usage commercial dans les grandes entreprises.

### **Q.3 : Docker est-il facile à apprendre ?**

**Ans:** Docker est facile à apprendre et est un outil de gain de temps.

##### **Article précédent**



* [Commandes](https://www.interviewbit.com/blog/category/commands/)

## [**Liste des commandes Python**](https://www.interviewbit.com/blog/python-commands/)

8 janvier 2024

##### **Prochain billet**



* [Commandes](https://www.interviewbit.com/blog/category/commands/)

## [**Commandes SQL: DDL, DML, DCL, TCL, DQL**](https://www.interviewbit.com/blog/sql-commands/)

8 janvier 2024



##### **Catégories**

* [Demandes](https://www.interviewbit.com/blog/category/applications/)
* [Architecture](https://www.interviewbit.com/blog/category/architecture/)
* [Livres](https://www.interviewbit.com/blog/category/books/)
* [Carrières](https://www.interviewbit.com/blog/category/careers/)
* [Caractéristiques](https://www.interviewbit.com/blog/category/characteristics/)
* [Problèmes de codage](https://www.interviewbit.com/blog/category/coding-problems/)
* [Commandes](https://www.interviewbit.com/blog/category/commands/)
* [Comparer](https://www.interviewbit.com/blog/category/compare/)
* [Composants](https://www.interviewbit.com/blog/category/components/)
* [Cours](https://www.interviewbit.com/blog/category/courses/)
* [Caractéristiques](https://www.interviewbit.com/blog/category/features/)
* [Cadres](https://www.interviewbit.com/blog/category/frameworks/)
* [IDE](https://www.interviewbit.com/blog/category/ide/)
* [Question d'interview](https://www.interviewbit.com/blog/category/interview-question/)
* [Entreprises de technologies de](https://www.interviewbit.com/blog/category/it-companies/)
* [Rôles des emplois](https://www.interviewbit.com/blog/category/job-roles/)
* [Bibliothèques](https://www.interviewbit.com/blog/category/libraries/)
* [Méthodes](https://www.interviewbit.com/blog/category/methodologies/)
* [Modèle](https://www.interviewbit.com/blog/category/model/)
* [Principes](https://www.interviewbit.com/blog/category/principles/)
* [Projets](https://www.interviewbit.com/blog/category/projects/)
* [Reprise](https://www.interviewbit.com/blog/category/resume/)
* [Salaires](https://www.interviewbit.com/blog/category/salaries/)
* [Compétences](https://www.interviewbit.com/blog/category/skills/)
* [Technologies](https://www.interviewbit.com/blog/category/technologies/)
* [Outils](https://www.interviewbit.com/blog/category/tools/)
* [Types](https://www.interviewbit.com/blog/category/types/)

InterviewBit

2021 InterviewBit

* [Meilleur cours de python gratuit](https://www.interviewbit.com/blog/best-free-python-course/)
* [Meilleur cours gratuit](https://www.interviewbit.com/blog/best-free-cpp-course/)
* [Meilleur cours gratuit de Java](https://www.interviewbit.com/blog/best-free-java-course/)
* [Cours JavaScript gratuits](https://www.interviewbit.com/blog/top-free-javascript-courses-to-learn-online/)
* [Meilleurs cours sur les sciences des données – gratuits et payés](https://www.interviewbit.com/blog/best-data-science-courses/)
* [Meilleures cours pour les structures de données et algorithmes - gratuits et payés](https://www.interviewbit.com/blog/best-courses-for-data-structures-and-algorithms/)
* [Meilleurs cours d'apprentissage en machine – gratuits et rémunérés](https://www.interviewbit.com/blog/best-machine-learning-courses/)
* [Meilleurs cours de conception de systèmes – gratuits et rémunérés](https://www.interviewbit.com/blog/best-system-design-courses/)
* [Meilleurs cours complets de développement de pile – gratuits et payants](https://www.interviewbit.com/blog/best-full-stack-developer-courses/)
* [Meilleurs cours de développement web – gratuits et payants](https://www.interviewbit.com/blog/best-web-development-courses/)