Python :  
Variables et opérations de base

Pascal Burkhard (pascal.burkhard@rpn.ch)

# Description de la séquence

## Mots clés

Python, variables, opérations de base.

## Résumé

Séquence théorique sur les bases de Python ; plus spécifiquement la création de variables et l’utilisation quelques opérateurs et fonctions simples de Python.

## Type d’activités

* une séquence théorique en présentation frontale ;
* une séquence pratique avec les élèves qui pratiquent de façon autonome.

## Public

Élèves de première année avec peu ou pas de notions de programmation.

# Objectifs

## Objectifs généraux

Apprendre les bases nécessaires à l’utilisation de Python pour manipuler différents types de données simples et les stocker dans des variables.

## Objectifs spécifiques

* apprendre à créer des variables ;
* apprendre à supprimer des variables ;
* apprendre à modifier des variables ;
* comprendre la notion de *type* *de données* dans le contexte des variables ;
* manipuler des valeurs numériques ;
* manipuler des chaines de caractères.

# Création de la séquence

## Connaissances préalables

Dans le cadre de ce premier projet pour le CAS d’Informatique, je souhaitais débuter avec un projet relativement simple dans Python. Le but étant de créer une séquence prévue pour le début de la première année avec quelques bases en rapport avec les variables et des opérations simples.

Je souhaitais cependant faire usage d’un certain nombre de connaissances préalables. J’ai donc utilisé des outils pour la création de ma présentation et de ma séquence d’exercice qui m’était déjà, en tout cas partiellement, connus.

Pour permettre un accès facilité aux documents utilisés pour créer la présentation et la séquence d’exercice, j’ai hébergé un *package*[[1]](#footnote-1)(pour plus d’informations sur la création de tels *package* R, voir *R Packages*[[2]](#footnote-2))sur un espace de stockage *GitHub* en plus de l’espace *Graasp*. Ce stockage sur *GitHub* permet à toute personne souhaitant installer le *package* dans R, de le faire aisément.

## Présentation *HTML* avec *xaringan*

Pour la création de la présentation, qui constitue le document nécessaire à la première partie de ma leçon, j’ai choisi d’utiliser *xaringan*[[3]](#footnote-3), ce *package* permet de rédiger des présentations dans un document *R Markdown*[[4]](#footnote-4) pour ensuite le transformer en document *HTML*.

L’avantage de cette solution, c’est qu’elle permet d’inclure des morceaux (appelés *chunks*) de code dans un document *R Markdown*. Le langage de ces morceaux est spécifié et le moteur qui génère le fichier *HTML* exécute le code et génère automatiquement le résultat des différentes commandes utilisées. J’ai également utilisé un *package* du nom de *flair*[[5]](#footnote-5) pour ajouter des mises en évidence dans certains morceaux de code.

Un autre avantage de ces présentations *xaringan*, c’est que le fichier *HTML* peut aisément être hébergé sur un serveur web et consulté en tout temps par les élèves. Le fichier *HTML* peut d’ailleurs être consulté en mode « raw » sur *Graasp*. Un lien vers le fichier disponible sur *Graasp* et sur *GitHub* se trouve dans les [Annexes](#_Annexes).

## Séquence d’exercices avec *learnr*

Pour la séquence d’exercice, j’ai retenu un *package* appelé *learnr*[[6]](#footnote-6)qui permet de mettre en place des séquences pédagogiques dans R. Comme pour la présentation *xaringan*, *learnr* fonctionne avec des documents *R Markdown* qui sont ensuite transformés et présentés sur une page web. Depuis peu de temps[[7]](#footnote-7), ce *package* intègre la possibilité d’avoir des blocs d’exercices en Python.

L’avantage de cette solution est de ne pas nécessiter d’installation de Python pour pouvoir effectuer les exercices (à condition d’avoir une version des exercices déployée). Un navigateur est amplement suffisant. Cela permettrait par exemple de donner la séquence d’exercice en devoir, même si les élèves ne disposent pas de Python chez eux.

Il faut cependant relever que *learnr* n’intègre Python que depuis peu de temps. Toutes les fonctions ne sont donc par encore disponibles. Il ne m’a pas été possible de trouver une méthode pour « évaluer » le code saisi dans les blocs Python alors que cette fonction est relativement simple pour du code R.

Les parcours pédagogiques que l’on peut créer avec *learnr* me semblent idéals pour des séquences d’apprentissage simples. Elles permettent à chaque élève d’avancer à son propre rythme et permettent une expérimentation « guidée ». Le *package* permet par ailleurs d’ajouter des indices, ce que je n’ai pas fait dans ma séquence.

## Déploiement de la séquence d’exercice

Les parcours pédagogiques créés avec *learnr* peuvent être exécutés sur une machine en local à condition d’avoir tous les composants logiciels nécessaires (R, différents *packages* R, Python, etc.). Ces parcours peuvent également être facilement déployés sur un service d’hébergement intégré dans *RStudio*[[8]](#footnote-8)(un IDE[[9]](#footnote-9), principalement pour R) appelé [shinyapps.io](https://www.shinyapps.io/). Cet hébergeur propose des solutions gratuites pour un nombre réduit d’applications.

En plus d’un déploiement sur [shinyapps.io](https://www.shinyapps.io/), j’ai cherché à héberger l’application sur mon propre serveur dans un *container* Docker[[10]](#footnote-10) ([www.docker.com](https://www.docker.com/)). Un fichier *Dockerfile* permettant de recréer le *container* se trouve à la racine de fichiers de mon *package*.

# En amont de la séquence

## Matériel

* un système de projection (Beamer, télévision, …) ;
* idéalement, un ordinateur par élève pour la séquence d’exercices.

## Références

Les références complètes sont dans la section [Annexes](#_Annexes).

Pour la partie théorique :

* Beginning Programming with Python[[11]](#footnote-11)
* Learn Python 3 the Hard Way[[12]](#footnote-12)
* Think Python[[13]](#footnote-13)

Pour la partie pratique :

* Language de programmation R[[14]](#footnote-14)
* IDE RStudio[[15]](#footnote-15)
* Package : *xaringan*[[16]](#footnote-16) et *learnr*[[17]](#footnote-17)

## Prérequis

* connaissances de base concernant le fonctionnement d’un ordinateur (mémoire par exemple) ;
* idéalement, quelques notions de ce qu’est un langage de programmation.

## Analyse *a priori*

Bien rendre les élèves attentifs à prendre des notes durant la séquence théorique. Certaines questions du parcours didactique (seconde partie de la leçon) portent sur les notions théoriques vues en frontal.

# Pendant la séquence

## Déroulement de l’activité 1 – théorie – 25 minutes

1. L’enseignant présente les bases théoriques avec le support de présentation.
2. L’enseignant répond aux éventuelles questions.

## Déroulement de l’activité 2 – pratique – 20 minutes

1. Les élèves testent leurs connaissances en utilisant le parcours pédagogique prévu.
2. Les élèves posent des questions si c’est nécessaire.

## Conseils pratiques pour l’utilisation de la séquence

Le temps prévu pour la présentation est éventuellement trop court, en fonction des explications données sur certaines diapositives. On peut cependant tout à fait envisager qu’une partie de la pratique se fasse en devoir.

Pour la partie pratique, idéalement les élèves travaillent seuls, alternativement une pratique à deux est envisageable.

# En aval de la séquence

## Analyse a posteriori

La séquence n’a pas pu être testée.

## Évaluation

La seconde partie de la séquence propose des exemples de questions qui examinent les connaissances retenues par les élèves. Ces questions peuvent servir d’exemple à d’éventuelles questions d’évaluation.

# Annexes

## Package R avec tous les documents sources

* <https://github.com/Nenuial/casInfo>

## Documents nécessaires pour la leçon

Présentation :

* Sur *Graasp*: <https://graasp.eu/resources/5fcdf96680dff72185648607> (à consulter en *Raw*)
* Sur *GitHub* : <https://nenuial.github.io/casInfo/inst/presentations/intro_variables.html>

Séquence d’exercices :

* Sur *shinyapps.io* : <https://nenuial.shinyapps.io/CASProjet1/>
* Sur mon hébergement personnel : [https://casprojet1.gondolin.app](https://casprojet1.gondolin.app/)

## Références

Bodwin, Kelly, et Hunter Glanz. *flair: Highlight, Annotate, and Format Your R Source Code*, 2020. https://r-for-educators.github.io/flair/index.html.

Downey, Allen B. *Think Python*. O’Reilly Media, Inc., 2012.

Mueller, John Paul. *Beginning Programming with Python For Dummies*. John Wiley & Sons, 2018.

R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2020. https://www.R-project.org/.

RStudio Team. *RStudio: Integrated Development for R*. Boston, MA: RStudio, PBC., 2020. http://www.rstudio.com/.

Schloerke, Barret, J.J. Allaire, et Barbara Borges. *learnr: Interactive Tutorials for R*, 2020. https://rstudio.github.io/learnr/.

Shaw, Zed A. *Learn Python 3 the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code*. Addison-Wesley Professional, 2017.

Shrestha, Nischal. « Learning Learnr ». Consulté le 13 octobre 2020. https://education.rstudio.com/blog/2020/07/learning-learnr/.

Wickam, Hadley. *R Packages: Organize, Test, Document and Share Your Code*. O’Reilly Media, Inc. Consulté le 7 novembre 2020. https://r-pkgs.org/.

Xie, Yihui. *xaringan: Presentation Ninja*, 2020. https://CRAN.R-project.org/package=xaringan.

Xie, Yihui, J.J. Allaire, et Garrett Grolemund. *R Markdown: The Definitive Guide*. Online version. Chapman & Hall/CRC, 2020. https://bookdown.org/yihui/rmarkdown/.

1. Extension du langage de programmation R. [↑](#footnote-ref-1)
2. Wickam, *R Packages: Organize, Test, Document and Share Your Code*. [↑](#footnote-ref-2)
3. Xie, *xaringan: Presentation Ninja*. [↑](#footnote-ref-3)
4. Xie, Allaire, et Grolemund, *R Markdown*. [↑](#footnote-ref-4)
5. Bodwin et Glanz, *flair: Highlight, Annotate, and Format Your R Source Code*. [↑](#footnote-ref-5)
6. Schloerke, Allaire, et Borges, *learnr: Interactive Tutorials for R*. [↑](#footnote-ref-6)
7. Shrestha, « Learning Learnr ». [↑](#footnote-ref-7)
8. RStudio Team, *RStudio: Integrated Development for R*. [↑](#footnote-ref-8)
9. Un environnement de développement. [↑](#footnote-ref-9)
10. Solution de virtualisation applicative. [↑](#footnote-ref-10)
11. Mueller, *Beginning Programming with Python For Dummies*. [↑](#footnote-ref-11)
12. Shaw, *Learn Python 3 the Hard Way*. [↑](#footnote-ref-12)
13. Downey, *Think Python*. [↑](#footnote-ref-13)
14. R Core Team, *R: A Language and Environment for Statistical Computing*. [↑](#footnote-ref-14)
15. RStudio Team, *RStudio: Integrated Development for R*. [↑](#footnote-ref-15)
16. Xie, *xaringan: Presentation Ninja*. [↑](#footnote-ref-16)
17. Schloerke, Allaire, et Borges, *learnr: Interactive Tutorials for R*. [↑](#footnote-ref-17)