

Initiation à la programmation - 42 Jour 03

Staff 42 bocal@42.fr

Résumé: Ce document est le sujet du jour 03 de la piscine d'initiation à la programmation.

Table des matières

T	Consignes	4
II	Préambule	3
III	${\bf Exercice} {\bf 00}: {\bf UPCASE_IT}$	4
IV	Exercice 01 : calculette	5
\mathbf{V}	Exercice 02 : scan_it	6
VI	Exercice 03: to25	7
VII	Exercice 04 : play_with_arrays	8
VIII	Exercice 05 : play_with_arrays++	9

Chapitre I

Consignes

- La procédure de correction se déroulera durant la dernière heure de la journée. Chaque personne corrigera une autre personne selon le model de peer-correcting.
- Vous avez une question? Demandez à votre voisine de droite. Sinon, essayez avec votre voisine de gauche.
- Lisez attentivement les exemples. Les exercices pourraient bien requérir des choses qui y sont précisées, et non dans le sujet...
- Votre manuel de référence s'appelle Google / man / Internet /

Chapitre II

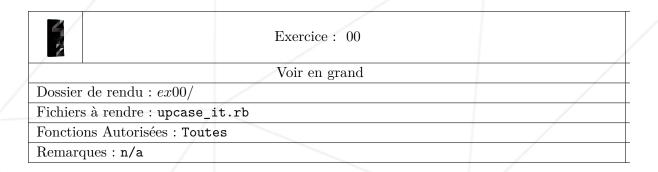
Préambule

Voici la liste des posters de motivation que l'on peut trouver dans le bureau de Barney Stinson au fil des saisons de How I Met Your Mother :

- Awesomeness: "When I get sad, I stop being sad and be Awesome Instead. True Story. Barney Stinson"
- Conformity: "It's the one who is different that gets left out in the cold."
- Courage: "True greatness comes when you're tested. Theodore Roosevelt"
- Challenge: "We either find a way or we make one"
- Opportunity: "You will always miss 100% of the shots you don't take."
- Teamwork: "Coming together is the beginning. Keeping together is progress. Working together is success. Henry Ford."
- Teamwork: "The chain is only as strong as the weakest link"
- Perseverance: "Continuous effort is the key to unlocking your potential. Sir Winston Churchill"
- Perfection: "It is not good enough to win, everybody else should lose"
- Strength: "What the mind can conceive, it can achieve"

Chapitre III

Exercice 00: UPCASE_IT



- Créez un script upcase_it.rb qui prend une chaîne de caractères en paramètre.
- Lorsqu'on l'exécute, le script affiche la chaîne de caractères en majuscules suivie d'un retour à la ligne.
- Si le nombre de paramètres est différent de 1, affichez none suivi d'un retour à la ligne.

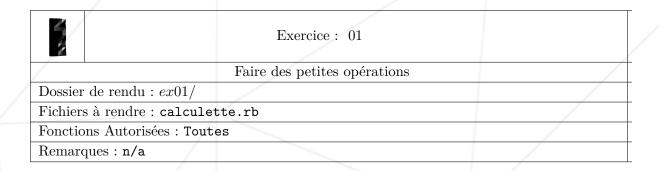
```
?> ./upcase_it.rb | cat -e
none$
?> ./upcase_it.rb "wi-filles" | cat -e
WI-FILLES$
?> ./upcase_it.rb 'CeT eXeRcIcE eSt AsSeZ fAcIlE !' | cat -e
CET EXERCICE EST ASSEZ FACILE !$
?>
```



Google upcase.

Chapitre IV

Exercice 01: calculette

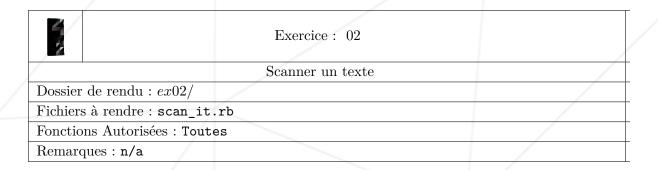


- Créez un script calculette.rb qui ne prend pas de paramètre.
- Votre script va demander 2 nombres à l'utilisateur.
- Vous devez stocker ces nombres sous forme numérique dans deux variables.
- Vous devez ensuite afficher le résultat de leur addition, soustraction, division et multiplication.

```
?> ./calculette.rb
Donne moi un premier nombre : 10
Donne moi un deuxieme nombre : 2
Merci !
10 + 2 = 12
10 - 2 = 8
10 / 2 = 5
10 * 2 = 20
?>
```

Chapitre V

Exercice 02: scan_it



- Créez un script scan_it.rb qui prend deux paramètres.
- Le premier paramètre est un mot clé à chercher dans une chaîne.
- Le deuxième paramètre est la chaîne à parcourir.
- Lorsqu'on l'exécute, le programme affiche le nombre de fois où on trouve le mot-clé dans la chaîne.
- Si le nombre de paramètres est différent de 2 ou que la première chaîne n'apparaît pas dans la deuxième, affichez none suivi d'un retour à la ligne.

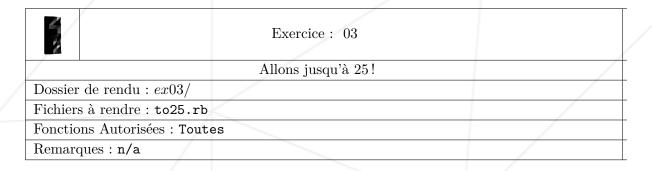
```
?> ./scan_it.rb | cat -e
none$
?> ./scan_it.rb "les" | cat -e
none$
?> ./scan_it.rb "les" "les exercices du J03 ne sont pas les plus difficiles" | cat -e
3$
?>
```



Essayez "string.scan(string)".

Chapitre VI

Exercice 03: to25

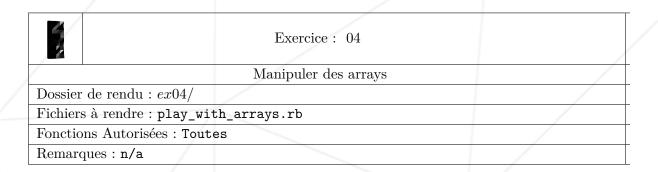


- Créez un script to25.rb qui prend un paramètre.
- Ce paramètre est un nombre, vous le stockerez dans une variable sous forme numérique.
- Vous allez ensuite faire une boucle qui affiche tous les nombres, du nombre fourni jusqu'à 25.
- Si le nombre de paramètres est différent de 1, vous afficherez "none" suivi d'un retour à la ligne.

```
?> ./to25.rb 8 9 10| cat -e
none$
?> ./to25.rb 20 | cat -e
Dans la boucle, ma variable vaut 20.$
Dans la boucle, ma variable vaut 21.$
Dans la boucle, ma variable vaut 22.$
Dans la boucle, ma variable vaut 23.$
Dans la boucle, ma variable vaut 24.$
Dans la boucle, ma variable vaut 25.$
?>
```

Chapitre VII

Exercice 04: play_with_arrays



- Créez un script play_with_arrays.rb.
- Votre script va itérer sur un array de nombres (que vous définirez) et construire un nouvel array en ajoutant 2 à chaque valeur de l'array d'origine.
- Vous devez donc avoir deux arrays dans votre programme, celui d'origine et le nouveau que vous avez créé.
- A la fin, affichez les deux arrays à l'ecran en utilisant la méthode p plutôt que puts.
- Par exemple, si votre array d'origine est [2, 8, 9, 48, 8, 22, -12, 2], vous aurez la sortie suivante :

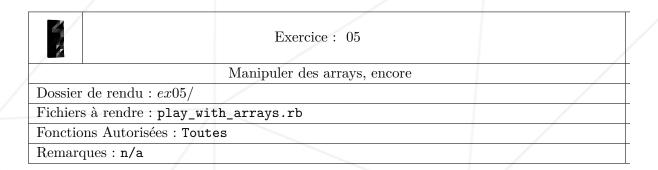
```
?> ./play_with_arrays.rb | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
[4, 10, 11, 50, 10, 24, -10, 4]$
?>
```



Google p method in ruby, array each

Chapitre VIII

Exercice 05: play_with_arrays++



- Reprenez le script précédent, mais cette fois vous ne traiterez que les valeurs supérieures à 5 de l'array d'origine.
- Par exemple, si votre array d'origine est [2, 8, 9, 48, 8, 22, -12, 2], vous aurez la sortie suivante :

```
?> ./play_with_arrays.rb | cat -e
[2, 8, 9, 48, 8, 22, -12, 2]$
[10, 11, 50, 10, 24]$
?>
```



Google p method in ruby, array each