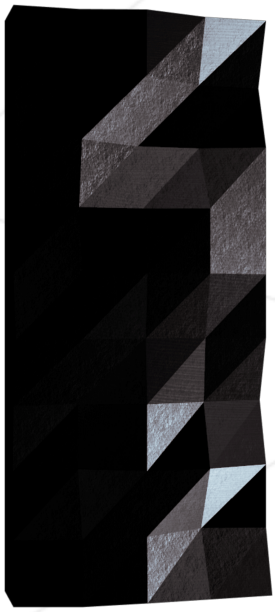

Initiation à la programmation en Ruby - 42

Samedi - révisions

Staff 42 bocal@42.fr

Résumé: Ce document est le sujet du samedi de révisions de la piscine d'initiation à la programmation en Ruby.

Table des matières

I	**Consignes**	**2**
II	**Exercice 00 : what's your name**	**3**
III	**Exercice 01 : age**	**4**
IV	**Exercice 02 : aff_first_param**	**5**
V	**Exercice 03 : scan_it**	**6**
VI	**Exercice 04 : advanced mult**	**8**
VII	**Exercice 05 : play_with_arrays**	**9**
VIII	**Exercice 06 : play_with_arrays+=2**	**10**

Chapitre I

Consignes

- A 42, vous allez faire l'expérience d'une pédagogie un peu particulière : vous avez un "cours" d'introduction d'1h tous les matins, et le reste de la journée, vous avez des exercices à réaliser en autonomie.
- Vous avez une question ? Un problème ? Un blocage ? Demandez à votre voisine de droite. Sinon, essayez avec votre voisine de gauche. A 42, les étudiants ne sont pas en compétition, ils avancent ensemble, en s'entre-aidant.
- Votre manuel de référence s'appelle `Google / man / Internet / ...`. Vous allez devoir apprendre à faire des recherches sur internet, toutes les infos dont vous avez besoin s'y trouvent !
- Lisez attentivement les exemples. Vous devez respecter le formattage des réponses : les majuscules, les retours à la ligne... tout est important. Programmer, c'est avant tout faire preuve de rigueur.
- Un tuteur vous accompagne tout au long de la piscine : il/elle est là pour vous donner des pistes, vous indiquer comment faire vos recherches sur internet et vous encourager si vous êtes démotivées. Le tuteur est un soutien pour vous, mais il n'est pas là pour vous donner les réponses !
- A la fin de la journée, le tuteur de votre rangée va corriger votre travail collectivement. C'est un bon moment pour échanger et s'expliquer, entre élèves, les erreurs que vous avez pu faire, ou au contraire expliquer aux autres ce que vous avez compris. Soyez attentives.
- Bon courage, et n'ayez pas peur de vous tromper ! Faites des tests, tatonnez - en informatique, c'est en faisant des erreurs qu'on apprend !

Chapitre II

Exercice 00 : what's your name

	Exercice : 00
Comment t'appelles-tu ?	
Dossier de rendu : $ex00/$	
Fichiers à rendre : `whatsyourname.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- Créez le script `whatsyourname.rb`
- Ce script va demander d'abord à l'utilisateur d'entrer son nom, puis son prénom, et enfin affiche les deux.

```
?> ./whatsyourname.rb
Hey, what's your first name ? : Laurie
And your last name ? : Mezard
Well, pleased to meet you Laurie Mezard.
?>
```

Chapitre III

Exercice 01 : age

	Exercice : 01
Recupérer et modifier un nombre	
Dossier de rendu : $ex01/$	
Fichiers à rendre : `age.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- Créez un script `age.rb` qui demande à l'utilisateur d'entrer son âge, puis affiche quel âge aura l'utilisateur dans 10 ans, 20 ans et 30 ans.

```
?> ./age.rb
Please tell me your age : 15
You are currently 15 years old.
In 10 years, you'll be 25 years old.
In 20 years, you'll be 35 years old.
In 30 years, you'll be 45 years old.
?>
```

Chapitre IV

Exercice 02 : aff_first_param

	Exercice : 02
Afficher un paramètre	
Dossier de rendu : $ex02/$	
Fichiers à rendre : `aff_first_param.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- Créez un script `aff_first_param.rb`.
- Le script affiche la première chaîne de caractères passée en paramètre suivie d'un retour à la ligne.
- S'il n'y a aucun paramètre, affichez `none` suivi d'un retour à la ligne.

```
?> ./aff_first_param.rb | cat -e
none$
?> ./aff_first_param.rb ``Code Ninja'' ``Numerique'' ``42'' | cat -e
Code Ninja$
?>
```

Chapitre V

Exercice 03 : scan_it

	Exercice : 03
Scanner un texte	
Dossier de rendu : $ex03/$	
Fichiers à rendre : `scan_it.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- 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 l a 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 J02 ne sont pas les plus difficiles'' | cat -e
3$
?>
```

`Essayez string.scan(string)`

```
Par exemple : "Hello there".("hello")
```

Chapitre VI

Exercice 04 : advanced mult

	Exercice : 04
Le retour du retour des tables de multiplication	
Dossier de rendu : $ex04/$	
Fichiers à rendre : `advanced_mult.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- Créez un script `advanced_mult.rb` qui ne prend pas de paramètre.
- Si le nombre de paramètres est différent de 0, vous afficherez “none” suivi d’un retour à la ligne.
- Ce script va afficher toutes les tables de multiplication sous la forme suivante :

```
?> ./advanced_mult.rb ``yolo'' | cat -e
none$
?> ./advanced_mult.rb
Table de 0: 0 0 0 0 0 0 0 0 0 0 0
Table de 1: 0 1 2 3 4 5 6 7 8 9 10
Table de 2: 0 2 4 6 8 10 12 14 16 18 20
Table de 3: 0 3 6 9 12 15 18 21 24 27 30
Table de 4: 0 4 8 12 16 20 24 28 32 36 40
Table de 5: 0 5 10 15 20 25 30 35 40 45 50
Table de 6: 0 6 12 18 24 30 36 42 48 54 60
Table de 7: 0 7 14 21 28 35 42 49 56 63 70
Table de 8: 0 8 16 24 32 40 48 56 64 72 80
Table de 9: 0 9 18 27 36 45 54 63 72 81 90
Table de 10: 0 10 20 30 40 50 60 70 80 90 100
?>
```

- Vous n’avez le droit qu’à deux boucles while.

Chapitre VII

Exercice 05 : play_with_arrays

	Exercice : 05
Manipuler des arrays	
Dossier de rendu : $ex05/$	
Fichiers à rendre : `play_with_arrays.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- Créez un script `play_with_arrays.rb`.
- Vous allez d'abord définir un array de nombres.
- Puis votre script va itérer sur cet array et construire un nouvel array en ajoutant 2 à chaque valeur d e l'array d'origine.
- Vous devez donc avoir deux arrays dans votre programme, celui d'origine et le nouveau que vous avez cré é.
- 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]$
?>
```

Chapitre VIII

Exercice 06 : play_with_arrays+=2

	Exercice : 06
Manipuler des arrays, encore	
Dossier de rendu : $ex06/$	
Fichiers à rendre : `play_with_arrays.rb`	
Fonctions Autorisées : `Toutes`	
Remarques : `n/a`	

- Reprenez le script précédent, mais cette fois vous ne traiterez que les valeurs supérieures à 5 de l'ar ray d'origine.
- Vous n'afficherez pas non plus les doublons.
- 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, 24]$
?>
```