CALCULS ET VARIABLES AVEC PYTHON - EXERCICES

1. CALCULS DANS LA CONSOLE:

Faites les tests vous même!

2. OU EST L´ERREUR?

```
age=input("Quel est ton âge? ")
print("Ok. L´an prochain, tu auras ",age+1, " ans")
```

Le message d´erreur est clair:

TypeError: can only concatenate str (not "int") to str

La variable ***age*** est de type *str* comme toute variable affectée via *input()*. Python interprète donc l´opérateur + comme une concaténation. Python ne peut pas concaténer (c´est à dire mettre bout à bout) une variable de type *str* avec une autre de type *int* (ou *float*).

Pour que l´opérateur + soit bien considéré comme un opérateur d´addition, il faut changer le type (transtyper) de la variable **age** en *int*. Il suffit de le faire dès la première affectation.

```
age=int(input("Quel est ton âge? "))
print("Ok. L´an prochain, tu auras ",age+1, " ans")
```

3. CONVERSION D´UNITES

- Conversion des degrés Celsius en Fahrenheit:

```
# initialisation de la variable d´entrée en la typant comme un
# nombre flottant (voir l´exercice 2!)
c=float(input("Entrez la température en degrés Celsius: "))
# on affecte à la variable f le résultat du calcul
f=c*9/5+32
# mise en forme du résultat
print("La température vaut",str(f)+"ºF")
```

On pouvait plus simplement écrire:

```
print("La température vaut",f,"ºF")
```

La conversion de la variable **f** en *str* permet la concaténation de la valeur calculée avec son unité, ce qui évite d´avoir un espace entre elles.

Le symbole º des degrés est un caractère spécial. Il peut aussi être codé par la suite de caractères `\u00B0` en UTF-8 (Unicode). Faites le test, nous y reviendrons plus tard.

- On pouvait aussi faire la conversion dans l´autre sens:

```
# f=float(input("Entrez la température en degrés Fahrenheit: "))
# on affecte à la variable c le résultat du calcul
c=(f-32)*5/9
# mise en forme du résultat
print("La température vaut",str(c)+"ºC")
```

4. INTENSITE DE LA PESANTEUR

```
# initialisation des variables en typant les entrées comme des
# nombres flottants (voir encore l´exercice 2)
G=6.67E-11
m=float(input("Entrez la masse de l´astre (kg): "))
r=float(input("Entrez le rayon de l´astre (km): "))
# Calcul de g avec la formule, sans oublier de convertir r en
# mètres!
g=G*m/(r*1000)**2
# Mise en forme de l´affichage du résultat arrondi 1 chiffre
# après la virgule
print("La valeur de g est:",round(g,1),"N/kg")
```

Remarque: Il est d´usage d´utiliser des lettres majuscules pour les noms de variables représentant une constante.

Lors du test du programme, on entre **5.97E24** pour la masse de la Terre. On retrouve bien la valeur connue. Faites le test pour d´autres astres.

5. CHAINE DE CARACTERES: x="NSI for ever!"

caractère	N	S	I		F	O	R		E	V	E	R	!
index	0	1	2	3	4	5	6	7	8	9	10	11	12
Index négatif	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
slice			>>>x[2:7] #7 est exclu										

Question	Quelle est la longueur de la chaine?	Quel est le sixième caractère ?		Quelles sont les lettres entre l´index 4 et 11?	Quel est le dernier caractère ?	Quel est le résultat de cette concaténation de la première lettre avec *"ever"*?
Instruction correspondante	**>>> len(x)**	>>> x[5]		>>>x[4:12]	**>>>x[-1]** (voir commentaire)	>>> x[0]+x[8:12]
Affichage de la console	**13**	**'o'**	**'x'**	**'for ever'**	**'!'**	**'Never'**

Commentaires:

- Le *slicing* n´est pas au programme de NSI mais il pourra être très utile pour votre futur projet.

- La troisième colonne est une erreur de saisie du concepteur de l´exercice :(

- Pour trouver le dernier caractère, on pouvait faire `>>>x[12]` ou, plus habilement `>>>x[len(x)-1]`. `>>>x[-1]` est une solution plus rapide proposée par Python pour indexer le dernier caractère (voir le tableau).

- `x[::-1]` permet de renverser la chaine de caractères. Faites le test.