INFORMATIQUE DS N° 1

15 DÉCEMBRE 2017

Clarté, indentation et syntaxe sont les meilleurs alliés des points.
Il est interdit dans ce devoir d'utiliser les fonctions Python `sum, max, min, etc.`

EXERCICE 1 - UN ORDINATEUR

1. Qui est considéré comme le "père" de l'informatique ?
2. Citez deux éléments de l'hardware d'un ordinateur.
3. Pour quelle raison ne peut-on pas qualifier un GPS d'ordinateur ?

EXERCICE 2 - CODAGE DES NOMBRES

1. Écrire les entiers 19 et 53 en binaire.
2. Écrire les nombres binaires 1011 et 10100111 en décimal.
3. Quel sont les chiffres du codage hexadécimal ?

EXERCICE 3 - TIPE

Les élèves de MP veulent stocker en machine les résultats d'une expérience pour leur TIPE. Dans cette expérience ils obtiennent des couples `(temps, potentiel)`, en secondes et en volts.
Les expériences sont faites sur une durée de 2000 secondes, à raison d'une mesure par seconde, et le potentiel prends des valeurs sur l'intervalle [-120 mV, 120mV], les valeurs sont arrondies au mV.
Ils veulent optimiser la place en machine en codant au mieux chaque mesure sur un petit nombre de bits.

1. Sur combien de bits (au minimum) peut-on encoder les valeurs possibles de `temps` ?
2. En complément à 2, quels sont les entiers que l'on code sur 8 bits ? Cela permet-il d'encoder les valeurs possibles de `potentiel` ?
3. En ajoutant du texte à leurs mesures, ils arrivent finalement à faire tenir une mesure `(temps,potentiel)` sur 3 octets. Chacun des 12 binômes de MP fait une expérience indépendante et ils souhaitent stocker l'ensemble des résultats sur une vielle clé USB de 1Mo. La taille de la clé est-elle suffisante ?

Une fois l'expérience réalisée, les données sont saisies dans une liste `potentiel`, telle que `potentiel[i]` donne le potentiel à la seconde `i`. Ils ont maintenant besoin d'aide pour traiter ces données .

4. Tout d'abord ils veulent chercher si des valeurs spécifiques sont dans cette liste. En regardant leur cours ils décident de construire une fonction `recherche(T,a)` qui renvoi `True` ou `False` selon que la valeur `a` soit présente ou non dans le tableau `T`. Écrire cette fonction.
5. Ils souhaitent ensuite déterminer les potentiels maximums et minimums.
   Écrire une fonction `extrema(potentiel)` qui renvoi le couple (maximum,minimum) des potentiels.
6. Il faut maintenant déterminer à quel(s) temps sont atteints ces potentiels maximum et minimum. Écrire une fonction `temps_extrema(potentiel)` qui renvoi deux listes donnant les temps ou sont atteints les maximums et les minimums.

Leur expérience terminée, ils déterminer une modélisation `P(t)` du potentiel P en fonction du temps t en seconde .
Pensant faire plaisir à leurs professeurs, les élèves décident de calculer l'intégrale $I = \int_0^{10} P(t)dt$ avec la méthode des rectangles. Ils se souviennent particulièrement d'une remarque de leur professeur d'informatique parlant d'un dessin.

7. Faire un dessin présentant la méthode des rectangles.
8. Proposer une suite d'instructions qui calcule cette intégrale I avec un pas de 0,01 s. On considère que la fonction `P(t)` est déjà saisie dans le code.

NOTE AUX BENÊTS : SUITE AU VERSO !

EXERCICE 4 - DÉCOMPOSITION D'UN NOMBRE

La fonction suivante détermine le nombre de chiffres dans l'écriture décimale de l'entier `n`.

```
def nb_chiffres(n):
    p = 0
    a = n
    while a !=0 :
        a = a//10
        p = p + 1
    return p
```

1. Parcourir cette fonction pour `n =564`.
2. Écrire une fonction `chiffre(n,c)` qui détermine si le chiffre `c` est présent dans l'écriture décimale de `n`.
3. Écrire une fonction `div_trois(n)` qui renvoie un booléen selon que la somme des chiffres de l'écriture décimale de `n` soit divisible par 3.
4. Écrire une fonction `nb_chiff_bin(n)` qui détermine le nombre de chiffres dans l'écriture binaire de `n`.

EXERCICE 5 - CALCULS ET COMPLEXITÉ

On cherche à effectuer le calcul de la somme $S = \sum_{k=0}^{n} \frac{x^k}{k!}$.

On considère que la fonction suivante est déjà saisie :

```
def puissance(x,n):
    P=1
    for k in range(n):
        P=P*x
    return(P)
```

1. Décrire l'appel `puissance(2,4)`.
2. Montrer que `puissance(x,n)` renvoi bien la valeur x^n en utilisant un invariant de boucle.
3. Donner la complexité de cette fonction en fonction de n.
4. Écrire une fonction `facto(n)` qui prends en argument un entier `n` et renvoi la valeur de `n!`.

On considère maintenant la fonction suivante :

```
def calcul_S(x,n):
    S=0
    for k in range(n+1):
        S = S + puissance(x,k)/facto(k)
    return(S)
```

5. Déterminer l'ordre de grandeur de la complexité en multiplications/divisions de cet algorithme.
6. Proposer une nouvelle version de cette fonction pour calculer S_n dont la complexité soit de l'ordre de n, en justifiant.