UNIVERSITÉ LIBRE DE BRUXELLES Août 2014
École polytechnique de Bruxelles

INFO-H-100 – Informatique – Prof. Th. Massart
1ère année du grade de Bachelier en Sciences de l'Ingénieur
Examen de seconde session

Remarques préliminaires

- On vous demande de répondre à **chaque question sur une feuille séparée**.
- N'oubliez pas d'inscrire votre nom, prénom et numéro de matricule sur chaque feuille.
- Vous disposez de 3 heures et vous ne pouvez pas utiliser de notes.
- La réponse à la question doit comprendre, si approprié, le code *Python* structuré et conforme aux règles de bonne pratique et conventions ainsi que des commentaires pertinents.
- Vous pouvez ajouter des fonctions si cela vous semble nécessaire.
- Sauf mention contraire, vous ne pouvez utiliser aucune fonction de librairies (pas d'`import`).

Question 1 – Récursivité (5 points)

Au départ d'une structure hiérarchique de fichiers (représentant un système de fichiers avec des sous-répertoires), on voudrait trouver tous les fichiers dont la taille est comprise dans un intervalle $[\min, \max]$ donné.

Concrètement, la structure de fichiers serait fournie à la fonction qu'on vous demande d'écrire sous la forme d'un dictionnaire, dont la clef représente respectivement le nom d'un fichier ou d'un sous-répertoire et la valeur associée représente respectivement la taille du fichier, ou un dictionnaire de même structure décrivant les fichiers du sous-répertoire.

Exemple :

```
DIR_DATA = {
    "points.txt": 124,
    "info.txt": 194,
    "examen": {
       "questions.tex": 4894,
       "solutions.tex": 498,
       "resultats.xls": 32938 },
    "notes.txt": 2310 }
```

On vous demande d'écrire la fonction (**récursive**) `find_files_sized(dir,min,max)`, qui renvoie une *liste* de tous les noms de fichiers qui correspondent aux critères de recherche. On suppose que la structure de fichiers `DIR_DATA` est donnée à la fonction, ainsi que les paramètres `min` et `max` qui indiquent l'intervalle dans lequel doit être comprise la taille du fichier pour être sélectionné.

Pour l'exemple ci-dessus, on aurait :

```
>>>print(find_files_sized(DIR_DATA,1,500))
['points.txt', 'info.txt','solutions.tex']
>>>print(find_files_sized(DIR_DATA,2000,10000))
['questions.tex', 'resultats.xls', 'notes.txt']
```

Note : Le résultat de la fonction sera une *liste* (et non un dictionnaire) et ne contiendra pas de sous-liste.

Question 2 - Python (4 points)

En prenant par exemple une instance du problème de la question 1 sur la récursivité, expliquez comment fonctionne la pile (stack) et le tas (heap) d'exécution (runtime) lors de l'exécution d'une fonction récursive. Pour cela, utilisez les diagrammes d'état vus au cours.

Question 3 - Complexité (3 points)

On vous demande ici d'illustrer par du code Python, des exemples dont le temps d'exécution (moyen ou maximal) est d'une certaine complexité (en $\mathcal{O}(f)$ où f est la complexité requise). Pour chaque sous-question, écrivez un code Python de quelques lignes (maximum 5 lignes) et justifiez votre réponse.

Donnez du code Python :
- donnant une complexité maximale en $\mathcal{O}(n \log n)$
- utilisant un dictionnaire et donnant une complexité moyenne en $\mathcal{O}(n)$
- utilisant une ou plusieurs listes et donnant une complexité maximale en $\mathcal{O}(n^3)$
- donnant une complexité moyenne et maximale respectivement en $\mathcal{O}(1)$ et $\mathcal{O}(n)$
- donnant une complexité maximale en $\mathcal{O}(log\, log\, n)$

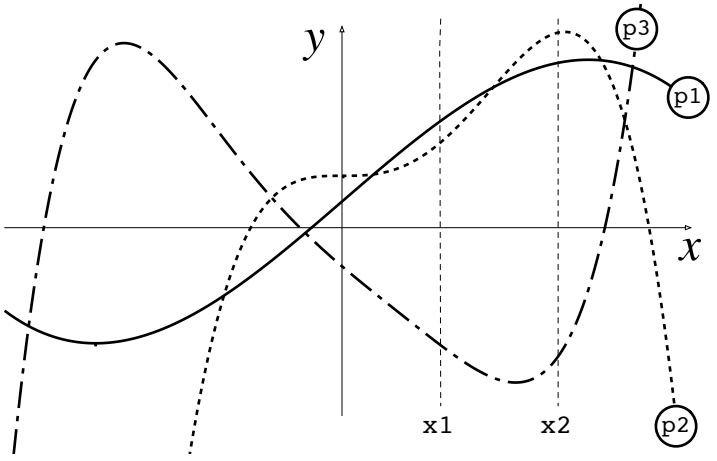
Question 4 - Tri (4 points)

On vous demande d'écrire une fonction `tri_poly_x(li_fn, x)` qui trie un ensemble de fonctions polynomiales (donné sous forme de liste `li_fn = [p1, p2, p3,...]`) dans l'ordre **dé**croissant de leur évaluation pour une valeur de `x` donnée, en utilisant, au choix, la méthode de tri par insertion ou par sélection. Chaque polynôme sera représenté par la liste de ses coefficients, l'indice de l'élément de la liste donnant la puissance de la variable associée.

Exemple :

La fonction a pour but de renvoyer l'ordre des polynômes pour un `x` donné. Prenons les trois polynômes suivants :
- `p1 = [0,1,-2]`, représentant le polynôme $P_1(x) = x - 3x^2$
- `p2 = [1,0,0,2,-5]`, représentant $P_2(x) = 1 + 2x^3 - 5x^4$
- `p3 = [0,1,0,-1]`, représentant $P_3(x) = x - x^3$

Le premier paramètre de la fonction aurait la forme suivante :
`li_fn = [ [0,1,-2], [1,0,0,2,-5], [0,1,0,-1] ]`.
Supposons ensuite que nous voulions les trier pour `x = 1`. Dans ce cas, nous aurions : $P_1(1) = -2$; $P_2(1) = -2$; $P_3(1) = 0$. L'ordre décroissant obtenu serait donc : `p3`, `p1`, `p2`. Pour d'autres valeurs de `x`, l'ordre obtenu pourrait être différent. Ceci peut être illustré à l'aide de l'exemple représenté par la figure ci-dessous :



- pour l'abscisse `x1`, l'ordre (décroissant) des polynômes est : `p1`, `p2`, `p3`
- pour l'abscisse `x2`, l'ordre (décroissant) des polynômes est : `p2`, `p1`, `p3`.

Question 5 – Opérations sur les fichiers (4 points)

On vous demande d'écrire une fonction `file_text_align(fin,fout,width)` qui lit un fichier texte nommé `fin` et crée un fichier texte nommé `fout` qui reprendra le texte `fin`, mais aura redécoupé les lignes de manière à ce que la longueur de chaque ligne du fichier résultat `fout` soit le plus proche possible mais inférieure à la valeur du paramètre `width`.

La fin d'un paragraphe est marquée par une ligne vide. Dans ce cas, la ligne du fichier sortie est interrompue et on commence un nouveau paragraphe (en laissant toujours une ligne vide).

Exemple :

```
file_text_align("in.txt","out.txt",50)
```

Fichier donné ("in.txt")

```
Sed ut perspiciatis unde omnis
iste natus error
sit voluptatem
accusantium
doloremque laudantium,
totam rem aperiam,
eaque
ipsa
quae ab illo inventore
veritatis
et quasi architecto
beatae vitae dicta sunt
explicabo.

Nemo enim ipsam voluptatem
quia voluptas sit aspernatur
aut
odit aut fugit,
sed quia consequuntur
magni dolores
eos qui ratione
voluptatem sequi
nesciunt.
```

➔

Fichier résultat ("out.txt")

```
Sed ut perspiciatis unde omnis iste natus error
sit voluptatem accusantium doloremque laudantium,
totam rem aperiam, eaque ipsa quae ab illo
inventore veritatis et quasi architecto beatae
vitae dicta sunt explicabo.

Nemo enim ipsam voluptatem quia voluptas sit
aspernatur aut odit aut fugit, sed quia
consequuntur magni dolores eos qui ratione
voluptatem sequi nesciunt.
```

Note : On ne tiendra compte que des espaces pour délimiter les mots (on suppose qu'il n'y a qu'un espace entre chaque mot d'une ligne, qu'il n'y a pas de tabulations ou autres espacements particuliers dans le texte fourni en entrée). Les signes de ponctuation seront supposés "collés" au mot à leur gauche. Une ligne pourra donc terminer par un signe de ponctuation, mais on ne pourra trouver ni espace, ni signe de ponctuation comme premier caractère d'une ligne du fichier résultat.