Exercices d'oral – informatique

– **2010**

1. Donner une modélisation d'un jeu de 52 cartes.
2. Écrire un programme permettant de tester si une carte tirée est un as.
3. Écrire un programme permettant de tirer une main de 4 cartes.
4. Écrire un programme permettant de tester si cette main contient au moins un as.
5. Écrire un programme permettant d'évaluer la probabilité d'obtenir au moins un as dans une main sur un ensemnle de n mains.
6. **Question subsidiaire :** Écrire un programme calculant la probabilité d'obtenir un carré dans une main de 5 cartes.

– **2011**

1. On dispose de n engins numérotés de 1 à n. Leur poids varie entre 4 et 10 tonnes. On peut envoyer un lot d'engins par avion si le poids total du chargement est inférieur ou égal à 150 tonnes.
2. Créer une fonction représentant le stock E d'engins qui, à chaque engin i, ($i \in [\![1, n]\!]$), associe son poids $E(i)$.
3. Créer une fonction définissant le sous-ensemble S de E noté V avec $V(i) = 1$ si l'engin i appartient à S et $V(i) = 0$ sinon.
4. Dénombrer les engins de S.
5. Calculer le poids de S.
6. Construire un sous-ensemble de E transportable par avion.
7. Créer une fonction qui, à un sous-ensemble S_ℓ de S associe le sous-ensemble comportant le maximum d'engins et transportable par avion ; avec $\ell \in [\![1\,,\,\frac{n}{10}]\!]$ et le calcul de $\frac{n}{10}$ sous-ensembles.

– **2010**

On dispose d'un ensemble d'indices T, sous la forme de N vecteurs de $E = \{0, 255\}^{\mathcal{D}}$. Étant donné un nouveau vecteur v de E, on souhaite trouver les k vecteurs de T les plus *proches* de v, au sens de la distance euclidienne sur E

1. Dans le cas où T n'est pas structuré et se présente sous la forme d'un tableau de N vecteurs non ordonnés, programmer l'algorithme de recherche des k plus proches voisins de v.

On cherche à présent à structurer T de manière à réduire le coût de la recherche des k plus proches voisins. Une technique dite par *pavage* consiste à découper l'espace E en hypercubes de côtés ρ par un partitionnement régulier de chaque dimension en intervalles d'amplitude ρ. Le recherche est alors réalisée en examinant d'abord les indices contenus dans le même hypercube que v, puis éventuellement les hypercubes voisins etc...

2. Programmer l'algorithme permettant de structurer T de manière adéquate.
3. Programmer ensuite le nouvel algorithme de recherche utilisant la structure précédente.
   Comment déterminer ρ, voire adapter le découpage en fonction des données de T ?

– **2011**

On joue au poker avec un jeu de 52 cartes (4 couleurs, 13 cartes par couleur) ; un brelan est composé de 3 cartes de même valeur ; chaque joueur possède 5 cartes. On cherche à savoir si un joueur possède un brelan.

1. Donner une modélisation appropriée du jeu de cartes.
2. Faire une fonction qui permet de tirer aléatoirement 5 cartes.
3. Réaliser une fonction qui indique la présence ou non d'un brelan.

– **2011**

Une image est composée de pixels. La couleur d'un pixel est représentée dans une matrice par un nombre compris entre 0 (blanc) et 255 (noir) les valeurs intermédiaires représentant les différents niveaux de gris.

1. Écrire une fonction qui donne une image carrée avec un fond noir, et au centre un rectangle blanc de longueur la moitié du côté de l'image et de largeur le quart du côté.
2. Écrire une fonction qui, pour une image donnée, donne la couleur du pixel le plus sombre et du pixel le plus clair.
3. Écrire une fonction qui permet d'améliorer linéairement... (la fin a disparu)

– **2011**
Soit u un vecteur de longueur n constitué d'entiers compris entre 0 et 9.

1. Écrire une fonction qui calcule la somme des composantes de u.
2. Écrire une fonction qui décale toutes les composantes de u vers la droite (la dernière se retrouve en premier à gauche).
3. u est symétrique si son premier terme et son dernier terme sont égaux, son deuxième et son avant-dernier aussi etc...
   Écrire une fonction qui teste si u est symétrique.
4. (a) Écrire une fonction qui calcule le symétrique d'un vecteur ([1, 2, 4, 5] est le symétrique de [5, 4, 2, 1]).
   (b) Écrire une fonction qui calcule u+sym(u).
   (c) Écrire une fonction qui calcule u+sym(u), puis réduit la somme modulo 9 (si un cœfficient est égal à 12 par exemple, il est remplacé par 3).
5. Reprendre la question 1 mais pour un vecteur composé d'entiers quelconques et réduire la somme modulo 9.

– **énoncé original, date inconnue**
Il est possible de colorier toute carte géographique plane avec 4 couleurs seulement, sans qu'aucune région n'en touche une autre de la même couleur (les coins ne comptent pas).
Pour simplifier le problème, on considère une matrice à deux dimensions, les régions étant représentées par des cases de la matrice. Toute case $M(x, y)$ touche les cases $M(x-1, y)$, $M(x+1, y)$, $M(x, y-1)$, $M(x, y+1)$, sauf pour les cases “sur les bords” de la matrice qui touchent une ou deux cases de moins que les autres. On représentera les 4 couleurs par les chiffres 1, 2, 3, 4.

1. Écrire une fonction qui, pour une case donnée, vérifie que les cases voisines ont bien une couleur différente de la sienne.
2. Écrire une fonction prenant en paramètre une matrice à deux dimensions déjà coloriée et renvoyant *vrai* si elle vérifie la condition initiale.
3. Écrire une fonction comptant le nombre de doublets de cases de même couleur se touchant pour une matrice (bien ou non) coloriée.
4. Écrire une fonction qui, pour ces doublets, cherche une couleur permettant de vérifier les contraintes. Si aucune n'est possible, laisser la couleur telle quelle.

– **énoncé original, date inconnue**
En cryptographie, un problème relativement intéressant consiste à déterminer le plus grand sous-ensemble S d'un ensemble D d'entiers relatifs tel que la somme des éléments de ce sous-ensemble soit nulle.
Nous nous proposons de nous intéresser à ce problème particulier.
Nous supposons que notre ensemble D contient n valeurs comprises entre -100 et $+100$.

1. Écrire une fonction qui construit l'ensemble D avec n valeurs choisies aléatoirement entre -100 et $+100$.
2. Nous supposons que les sous-ensembles S de D sont représentés sous la forme d'un vecteur v composé de 0 et de 1 de longueur n. Si la composante $v[i]$ est égale à 0, alors l'élément $D[i]$ n'est pas présent dans le sous-ensemble S.
   Par exemple, si $D = [3, -3, 9, -6, 7, 6]$ alors le vecteur $[0, 1, 0, 1, 1, 0]$ représente le sous-ensemble $\{-3, -6, 7\}$.
   (a) Écrire une fonction qui prend comme argument un sous-ensemble S de D sous la forme d'un vecteur composé de 0 et de 1, et qui renvoie le cardinal de S.
   (b) Écrire une fonction qui prend comme argument un sous-ensemble S de D sous la forme d'un vecteur composé de 0 et de 1, et qui renvoie la somme des éléments contenus dans S.
   (c) Écrire une fonction qui construit de manière aléatoire un sous-ensemble S de D sous la forme d'un vecteur composé de 0 et de 1.
3. Nous cherchons désormais le plus grand sous-ensemble S de D telle que la somme des éléments de S est nulle.
   (a) Écrire une fonction qui construit de manière aléatoire un sous-ensemble S de D tel que la somme des éléments contenus dans S est nulle.
   (b) Écrire une fonction qui construit de manière aléatoire $n/10$ sous-ensemblesde D tel que la somme des éléments contenus dans chacun est nulle et qui renvoie le sous-ensemble S_i ayant la plus grande cardinalité (i.e. le plus grand nombre d'éléments).

   Rappel : la fonction *rand* retourne un nombre réel aléatoire, suivant une loi uniforme, compris dans $[0, 1]$.

– **2003**
Écrire une fonction qui range les composantes d'un vecteur à l'envers.

– **2003**

1. Écrire une fonction donnant le nombre de composantes non nulles d'un vecteur.
2. Écrire une fonction donnant le nombre maximal de composantes non nulles consécutives d'un vecteur.

– **2003**
Écrire une fonction qui range les éléments d'un tableau à une dimension pr ordre croissant.
Méthode imposée : tri par insertion avec utilisation d'un second tableau.

– **2003** Écrire une fonction qui renvoie le numéro d'une colonne d'une matrice telle que la somme des éléments de cette colonne est maximale.

– **2005**

1. Écrire une fonction qui donne la transposée d'une matrice carrée.
2. Écrire une fonction qui donne la transposée d'une matrice carrée sans réécrire la matrice.
3. Écrire une fonction qui calcule le produit de deux matrices.
4. Écrire une fonction qui donne le plus petit et le plus grand élément d'une matrice et les échange.

– **2005**
On donne 3 chiffres et on s'intéresse au plus petit et au plus grand nombre que l'on peut former avec ces 3 chiffres ; par exemple, avec 3, 4 et 5, le plus petit est 345 et le plus grand 543.
On montre que si la différence entre le plus grand et le plus petit est strictement supérieure à 1, en ôtant le plus petit nombre au plus grand et en recommençant, on finit toujours par obtenir 495 : par exemple, $543 - 345 = 198$, $981 - 189 = 792$ etc...

1. Écrire une fonction qui donne le plus grand nombre et une qui donne le plus petit, avec comme paramètre d'entrée un vecteur.
2. Trouver le nombre d'itérations pour arriver à 495 *(on se place dans les conditions décrites ci-dessus).*

– **2005** On donne les fonctions :

- *deplacer_tete_imprimante*(p)
  Si $p = 1$, elle déplace la tête de l'imprimante d'une position vers la droite.
  Si $p = -1$, elle déplace la tête de l'imprimante d'une position vers la gauche.
- *deplacer_ligne*(n)
  Elle déplace la tête de l'imprimante de n lignes vers le bas.
- *imprimer*(c)
  Elle imprime le caractère c à la position de la tête.

L'imprimante imprime de la droite vers la gauche les lignes de numéro impair (caractères de position 1, 2, 3,...88) et de la gauche vers la droite les lignes de numéro pair (caractères de position 88, 87,... 1).
Une page contient 66 lignes et, au bout d'une page, pour passer à la page suivante, l'imprimante saute 10 lignes.
Écrire une focntion qui imprime une liste de caractères à partir d'une position ligne–colonne.