teaching sciences UPSTI for innovation

Proposition de corrigé

Concours : Concours Commun INP

Année : 2023

Filière : TSI

Épreuve : Informatique

Ceci est une proposition de corrigé des concours de CPGE, réalisée bénévolement par des enseignants de Sciences Industrielles de l'Ingénieur et d'Informatique, membres de l'UPSTI (Union des Professeurs de Sciences et Techniques Industrielles), et publiée sur le site de l'association :

https://www.upsti.fr/espace-etudiants/annales-de-concours

A l'attention des étudiants

Ce document vous apportera des éléments de corrections pour le sujet traité, mais n'est ni un corrigé officiel du concours, ni un corrigé détaillé ou exhaustif de l'épreuve en question.

L'UPSTI ne répondra pas directement aux questions que peuvent soulever ces corrigés : nous vous invitons à vous rapprocher de vos enseignants si vous souhaitez des compléments d'information, et à vous adresser à eux pour nous faire remonter vos éventuelles remarques.

Licence et Copyright

En cas de doute, n'hésitez pas à nous contacter à : corrigesconcours@upsti.fr.

Informez-vous !

Retrouvez plus d'information sur les Sciences de l'Ingénieur, l'orientation, les Grandes Ecoles ainsi que sur les Olympiades de Sciences de l'Ingénieur et sur les Sciences de l'Ingénieur au Féminin sur notre site : www.upsti.fr

L'équipe UPSTI

Gestion de Tests dans une entreprise

Corrigé UPSTI

Partie I – Tests de code de sécurité sociale

Question 1 Écrire la fonction `num_secu` qui, à partir de la chaîne de caractères d'un numéro de sécurité sociale, donne le numéro sous la forme d'un entier. Le programme devra parcourir la chaîne de caractères représentant le numéro de sécurité sociale en supprimant les caractères d'espacement, puis la transformer en un nombre entier. Cette fonction a un paramètre de type `string` et retourne une valeur de type `int`.

```python
def num_secu(chaine):
    n_chaine = ""
    for c in chaine:
        if c != " ":
            n_chaine += c
    return int(n_chaine)
```

Question 2 Écrire la fonction `clef` qui détermine la valeur de la clé d'un numéro de sécurité sociale. Cette fonction a un paramètre de type `int` et retourne un élément de type `int`.

```python
def clef(num):
    return 97 - (num % 97)
```

Question 3 Écrire la fonction `num_secu_complet` qui détermine le numéro complet de sécurité sociale. Cette fonction a un paramètre de type `int` et retourne un élément de type `int`.

```python
def num_secu_complet(num):
    #Travail sur les str
    key = clef(num)
    chaine = str(num) + str(key)
    return int(chaine)
```

```python
def num_secu_complet(num):
    #Travail sur les int
    return 100 * num + clef(num)
```

Question 4 Écrire la fonction `test_num_secu` qui détermine si un numéro de sécurité sociale est correct. Cette fonction a un paramètre de type `string` et retourne un élément de type `bool`.

```python
def test_num_secu(chaine):
    #Travail sur les str
    num = num_secu(chaine[:19])
    num_vrai = num_secu_complet(num)
    num_donne = num_secu(chaine)
    return num_vrai == num_donne
```

```python
def test_num_secu(secu):
    #Travail sur les int
    secu_num = num_secu(secu)
    nir = secu_num // 100
    return secu_num == num_secu_complet(nir)
```

Partie II – Test de numéro de carte de crédit

Question 5 Écrire une fonction `num_en_liste` qui transforme un nombre entier en une liste de chiffres. Cette fonction a un paramètre de type `int` et retourne un élément de type `list`.

```python
def num_en_liste(num):
    #Travail sur les str
    chaine = str(num)
    liste = []
    for c in chaine:
        liste.append(int(c))
    return liste
```

```python
def num_en_liste(num):
    #Travail sur les int
    liste_chiffres = []
    while num > 0:
        liste_chiffres = [num % 10] + liste_chiffres
        num = num // 10
    return liste_chiffres
```

Question 6 Écrire une fonction `tuple_pairs_impairs` qui détermine un tuple représentant la liste des chiffres d'indice pair et la liste des chiffres d'indice impair d'un numéro de carte de crédit. Le chiffre le plus à droite de ce numéro est considéré comme le premier chiffre d'indice impair. Cette fonction a un paramètre de type `int` et retourne un tuple composé de deux éléments de type `list`.

```python
def tuple_pairs_impairs(num):
    liste = num_en_liste(num)
    liste.reverse()
    pairs = []
    impairs = []
    for i in range(len(liste)):
        if i%2 == 0:
            pairs.append(liste[i])
        else:
            impairs.append(liste[i])
    return (impairs, pairs)
```

ou

```python
def tuple_pairs_impairs(num):
    liste_chiffres = num_en_liste(num)
    n = len(liste_chiffres)
    liste_pair = [liste_chiffres[i] for i in range(n - 2, -1, -2)]
    liste_impair = [liste_chiffres[i] for i in range(n - 1, 0, -2)]
    return (liste_pair, liste_impair)
```

Question 7 Écrire une fonction `cree_dico qui`, à partir d'un numéro de carte de crédit, crée un dictionnaire avec deux clés nommées 'pair' et 'impair'. La clé 'pair' est constituée de la liste des nombres d'indice pairs du numéro de la carte de crédit et la clé 'impair' de la liste des nombres d'indice impairs.

```python
def cree_dico(num):
    pairs, impairs = tuple_pairs_impairs(num)
    dico = {'pair' : pairs, 'impair' : impairs}
    return dico
```

Question 8 Écrire une fonction `traitement_nb_pairs` qui multiplie par 2 tous les chiffres de la liste associée à la clé 'pair'. Si un chiffre est supérieur à 9, il faut réaliser la somme des deux chiffres qui le composent. Cette fonction a un paramètre de type `dictionnaire` et retourne un dictionnaire.

```python
def traitement_nb_pairs(dico):
    pairs = dico['pair']
    n_pairs = []
    for chiffre in pairs:
        d = chiffre*2
        c = str(d)
        if len(c) == 2:
            d = int(c[0]) + int(c[1])
        n_pairs.append(d)
    dico['pair'] = n_pairs
    return dico
```

```python
def traitement_nb_pairs(dico):
    for i in range(len(dico['pair'])):
        el = dico['pair'][i] * 2
        if el > 9:
            el = el // 10 + el % 10
        dico['pair'][i] = el
    return dico
```

Question 9 Écrire une fonction `test_num_carte_credit` qui utilise l'algorithme de Luhn pour savoir si un numéro de carte de crédit est correct. Vous devez utiliser la fonction `traitement_nb_pair` pour sa réalisation. Cette fonction a un paramètre de type `int` et retourne une valeur de type bool.

```python
def test_num_carte_credit(num):
    dico = cree_dico(num)
    dico = traitement_nb_pairs(dico)
    total = sum(dico['pair']) + sum(dico['impair'])
    return total % 10 == 0
```

Partie III – Tests de QR code

Question 10 Écrire une fonction `init` qui réalise l'initialisation d'une liste de dimension n où chaque élément est également une liste de dimension n. Cette liste de listes représente ainsi une matrice de taille n x n . Cette fonction a un paramètre de type `int` et retourne une liste de listes qui représente un QR code initialisé avec des valeurs 0.

```python
def init(n):
    matrice = []
    for i in range(n):
        liste = [0]*n
        matrice.append(liste)
    return matrice
```

```python
def init(n):
    return [[0 for i in range(n)] for j in range(n)]
```

Question 11 Écrire la fonction `charge_valeur` qui a pour but de réduire les données de l'image dans une liste de listes de dimension 21*21. Attention : l'image correspondant à un QR code représente une liste de listes de dimension 420*420 dont on veut réduire tous les blocs constitués de 20*20 pixels à un seul pixel pour avoir à partir de l'image une liste de listes de dimension 21*21. Ne pas oublier que tous les pixels d'un bloc sont identiques. Cette fonction a un paramètre de type image et retourne une liste de listes de triplets (couleur des pixels).

```python
def charge_valeur(img):
    taille = img.size
    matrice = init(21)
    for i in range(0,taille[0],20):
        for j in range(0,taille[1],20):
            matrice[i//20][j//20] = img.getpixel(i,j)
    return matrice
```

```python
def charge_valeur(im):
    liste = init(21)
    for i in range(21):
        x = i * 20
        for j in range(21):
            y = j * 20
            liste[i][j] = img.getpixel(x, y)
    return liste
```

Question 12 Écrire une fonction `cree_bloc` qui crée un bloc de positionnement. Cette fonction, qui n'a pas de paramètre, retourne une liste de listes de dimension 7*7.

```python
def creer_bloc():
    matrice = init(7)
    for i in range(7):
        for j in range(7):
            if ((i == 1 or i == 5) and (j!=0 and j!=6)) or (i == 2 or i == 3 or i == 4) and (j == 1 or j == 5):
                matrice[i][j] = 1
    return matrice
```

ou

```python
def cree_bloc():
    matrice = init(7)
    for i in range(1, 6):
        matrice[1][i] = 1
        matrice[5][i] = 1
        matrice[i][1] = 1
        matrice[i][5] = 1
    return matrice
```

Question 13 Écrire une fonction `test_bloc` qui teste si un bloc de positionnement (rappel : il y en a trois) est bien représenté pixel par pixel dans un QR code. Cette fonction a 3 paramètres : les coordonnées x et y donnant la position du début d'un bloc de positionnement d'un QR code (toujours les coordonnées du pixel le plus haut et à gauche) et la liste de listes de dimension 21*21 associée au même QR code. Cette fonction retourne un booléen.

```
def test_bloc(x, y, matrice):
    bloc = creer_bloc()
    for i in range(x, x + 7):
        for j in range(y, y + 7):
            if matrice[i][j][0] / 255 != bloc[i-x][j-y]:

                return False
    return True
```

Question 14 On considère qu'un QR code est bien positionné lorsque ses 3 blocs de contrôle sont effectivement présents en haut à gauche, en haut à droite et en bas à gauche (comme sur la figure 1). Écrire une fonction `test_QRcode` qui permet de tester si un QR code est bien positionné. Cette fonction a pour paramètre une matrice de dimension 21*21 et retourne un booléen.

```
def test_QRcode(matrice):
    if test_bloc(0,0,matrice) and test_bloc(0,14,matrice) and test_bloc(14,0,matrice):
        return True
    return False
```

Question 15 Écrire une procédure `tourHoraire` qui réalise une rotation de 90°, dans le sens des aiguilles d'une montre, des 4 éléments du QR code. La fonction a trois paramètres, les coordonnées x et y d'un élément de la liste de listes et une liste de listes de dimension 21*21.

```
def tourHoraire(x, y, mat):
    n = len(mat) - 1
    a = liste[x][y]
    liste[x][y] = liste[n - y][x]
    liste[n - y][x] = liste[n - x][n - y]
    liste[n - x][n - y] = liste[y][n - x]
    liste[y][n - x] = a
```

Question 16 Écrire la procédure `rotationHoraire` qui réalise la rotation de 90° d'un QR code. Cette procédure a un seul paramètre, une liste de listes de dimension 21*21.

```
def rotationHoraire(matrice):
    indices_fait = []
    n = len(matrice) - 1
    for i in range(len(matrice)//2):
        for j in range(i,n-i):
            tourHoraire(i,j,matrice)
            indices_fait += [(i,j), (n-i, j), (i, n-j), (n-i, n-j)]
```

Question 17 Connaissant les 4 positions possibles lors de la lecture d'un QR code par un appareil dédié, écrire la procédure `QRcode_posi` qui positionne correctement un QR code. Cette procédure a un seul paramètre, une liste de listes de dimension 21*21.

```
def QRcode_posi(matrice):
    while not test_QRcode:
        rotationHoraire(matrice)
```

Partie IV – Gestion réseau

Question 18 Compléter les 3 lignes manquantes du tableau $M+$ sur le DR.

	A	B	C	D	E	F
étape initiale	0	∞	∞	∞	∞	∞
$A(0)$	-	2	∞	1	∞	∞
$D(1_A)$	-	2	∞	-	3	∞
$B(2_A)$	-	-	5	-	3	7
$E(3_D)$	-	-	5	-	-	4
$F(4_E)$	-	-	5	-	-	-
$C(5_B)$	-	-	-	-	-	-

Question 19 Donner la valeur du plus court chemin entre "A" et "F". Expliquer comment on obtient cette valeur à l'aide du tableau $M+$. Expliciter le chemin le plus court trouvé pour aller de "A" à "F".

Valeur entre "A" et "F" : 4. C'est la valeur qu'on lit dans la colonne F. Le chemin le plus court est A -> D -> E -> F

Partie V – Requêtes SQL

Question 20 Écrire, en SQL, la requête (1) qui permet d'obtenir le numéro de carte de crédit de toutes les personnes référencées dans la base de données de l'entreprise dont le numéro de sécurité sociale commence par 2. On utilisera le caractère "_" comme le séparateur des milliers. Par exemple 10000000 sera réécrit comme 10_000_000.

```
SELECT num_CB FROM clients
WHERE num_secu BETWEEN 200_000_000_000_000 AND 299_999_999_999_999
```

Question 21 Écrire, en SQL, la requête (2) permettant d'obtenir le nom et le prénom de toutes les personnes ayant effectué un achat avec un résultat sans doublon.

```
SELECT DISTINCT nom, prenom FROM clients
JOIN ventes ON clients.id = num_client
```

Question 22 Écrire, en SQL, la requête (3) qui permet d'obtenir les produits associés à chaque numéro de carte de crédit du client et qui ont été vendus entre le 1 juin 2020 et le 30 juillet 2020. On rappelle que SQL compare les variables de type TEXT grâce à l'ordre lexicographique : par exemple "13-06-1989 < 13-07-1999" est vrai.

```
SELECT nom_produit, num_CB FROM produits
JOIN ventes ON produits.id = ventes.ref_produits
JOIN clients ON num_client = clients.id
WHERE date BETWEEN "2020-06-01" AND "2020-07-30"
```