OFPPT

مكتب التكوين المهني وإنعاش الشغل

Office de la Formation Professionnelle et de la Promotion du Travail

Direction Recherche et Ingénierie de la Formation

Examen de passage à la 2eme année

Session Juillet 2015

Filière : Techniques de Développement Informatique
Niveau: TS
Durée : 5 heures

Epreuve : Synthèse
Variante : V1
Barème : / 120Pts

❖ Partie I : Théorie (40 pts)

➢ Dossier 1: Notions de mathématiques appliquées à l'informatique (12 pts)

Exercice 1:(6 pts)

NB : la calculatrice est strictement interdite.

Compléter le tableau suivant:

670			
	10011011		
		1450	
			15F

Exercice 2: Soit la fonction logique suivante:

$$F(A,B,C) = (AB + \overline{A})C + (AB + \overline{B})\overline{C} + (A + C)\overline{B}$$

1- Simplifier analytiquement la fonction logique F. **(2 pts)**
2- Construire la table de vérité. **(2 pts)**
3- Simplifier avec la méthode de Karnaugh la fonction logique F. **(2 pts)**

➢ Dossier 2: Techniques de programmation structurée (16 pts)

Exercice 1:

Quel résultat fournit l'algorithme suivant **(6 pts)**

```
Variables   i : entier
            Mot, ch : chaine de caracteres
Début
    Mot ← "SALUTATIONS"

    Pour i =1 à 6 faire
        ch ← ""     //chaine vide
        Pour j =1 à 6 – i + 1  faire
            ch ← ch & "  " // chaine espace
        fin pour
        ch ← ch  &  sous_chaine(Mot, 1, 2*i - 1)
        écrire (ch) ;
    fin pour
Fin
```

NB :

- **&** : permet la concaténation de chaines
- **Sous_chaine (chaine, i, n)** : renvoie la partie de la chaîne qui contient n lettres et qui commence à partir de l'indice i. (par exemple pour la chaine « bonjour » si i=3 et n=4 la fonction retourne « njou »)
- **Ecrire(chaine)** : permet d'afficher une chaine et retourner à la ligne

Exercice 2: (10 pts)

Ecrire une procédure **SupVoyelles()** qui permet d'éliminer les voyelles (a, e, y, u, i, o) à partir d'un tableau de caractères. **(6 pts)**

Faites appel à cette procédure dans un algorithme donnant le droit à l'utilisateur de remplir un tableau avec **N** valeurs de type caractères. **(4 pts)**

Exemple :

Tableau saisi :

b	o	n	j	o	u	r

Tableau résultat :

b	n	j	r

Dossier 3: Conception et modélisation d'un système d'information (12 pts)

Conception d'un système d'information pour une amicale

Une amicale, ayant pour objectif la réalisation des appartements répondant à des normes de qualité et de sécurité, désire réaliser un site web pour gérer ses adhérents et ses projets.

L'amicale propose plusieurs projets, chacun se situe dans une zone spécifique et se caractérise par une date de démarrage et une date de fin prévisionnelle.

Chaque projet propose plusieurs types d'appartements. Le type détermine la superficie, le nombre de chambres, le montant total et le montant de la cotisation trimestrielle à payer par l'adhérent.

Un adhérent peut bénéficier de plusieurs type d'appartement dans le cadre du même projet voire même différents projets.

Le système devra également mémoriser le montant a payé par l'adhérent chaque trimestre pour chaque projet dont il veut bénéficier ainsi que le mode de paiement (virement, chèque, espèce).

En ce qui concerne les adhérents, on doit mémoriser leur cin, nom, prénom, adresse actuelle et le n° de téléphone, ainsi que la première date d'inscription

a) Etablir le dictionnaire de données. **(3 pts)**
b) Etablir le modèle conceptuel de données. **(6 pts)**
c) Etablir le modèle logique de données. **(3 pts)**

❖ Partie II: Pratique (80 pts)

➢ Dossier 1: Langage de programmation structurée (20 pts)

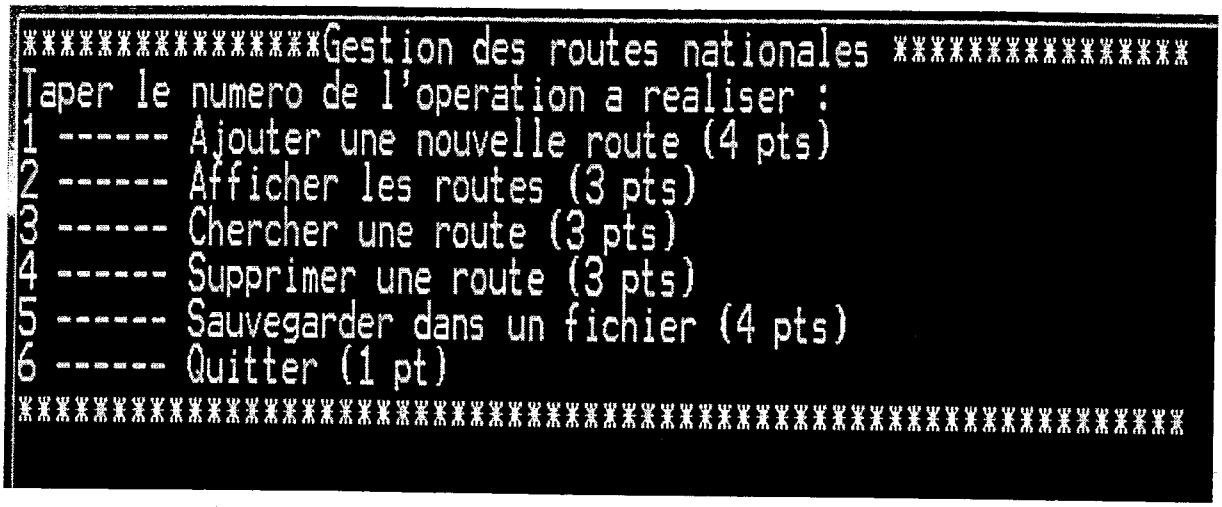
On souhaite écrire un programme permettant de gérer l'ensemble des routes nationales au Maroc. Pour cela nous allons considérer la structure **RouteN** ayant pour données :

- **Nom** qui correspond au nom de la route
- **Distance** qui correspond à la longueur (en km) de la route associée à la structure,
- **VilleD** qui correspond à la ville de départ,
- **VilleA** qui correspond à la ville d'arrivée

On suppose que les routes sont stockées dans un tableau **les_routes** de dimension maximale 100 et dont la taille réelle est enregistrée dans une variable n.

Ecrire un programme complet permettant la gestion des routes nationales.

1. Ce programme doit afficher au départ le menu suivant : **(2 pts)**



Ci-dessous l'explication du menu :

1. Ajouter une nouvelle route dont les informations sont saisies au clavier au tableau des routes.
2. Afficher les routes saisies.
3. Afficher les routes dont la ville de départ est saisie au clavier.
4. Supprimer du tableau une route dont le nom est donné par l'utilisateur.
5. Copier le contenu de tableau des routes dans un fichier texte dont le nom est saisi par l'utilisateur. Chaque route est stockée dans une ligne, les champs sont séparés par le caractère virgule (« , »).

➢ Dossier 2: Programmation orientée objet (30 pts)

Développement d'une application orientée objet pour la gestion d'un magasin

On souhaite informatiser la gestion des ventes au sein d'un magasin. On considère alors qu'un article est caractérisé par son numéro de série, son prix hors taxe, sa quantité en stock, et la quantité minimale

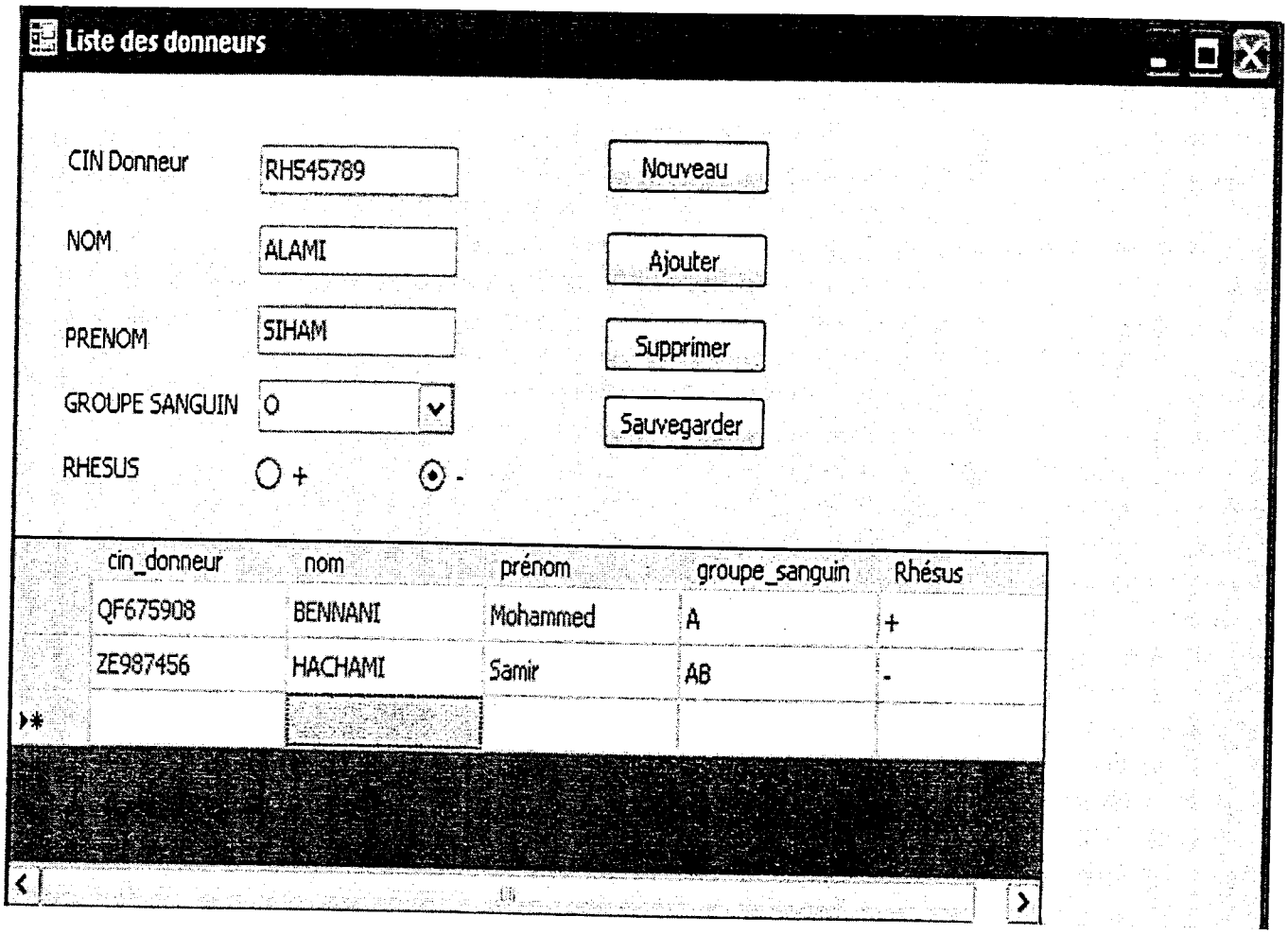
1) a) Ecrire la classe « **Article** ». **(2 pts)**
   Ajouter à cette classe un constructeur permettant d'instancier des objets de la classe « Article » dont on précisera le numéro de série, le prix hors taxe, la quantité en stock, la quantité minimale et un constructeur sans paramètres. **(2 pts)**
   b) Réécrire la méthode ***ToString()*** pour afficher les caractéristiques d'un article. **(2 pts)**
   c) Ajouter à la classe Article les méthodes suivantes:
   - **S'approvisionner (int qte) :** qui permet d'approvisionner le stock par une quantité donnée. **(2pts)**
   - **Achat (int qte)** permet de traiter un achat d'un article par un client. Une opération d'achat aura pour effet de déduire la quantité achetée du stock. Si la quantité qui reste est inférieure à la quantité minimale on avise par un message. **(2 pts)**

2) Un habit est un article qui a une taille et une couleur :
   a) Ecrire la classe « **Habit** » héritant de la classe « **Article** ». **(2 pts)**
   b) Récrire le constructeur de cette classe afin d'initialiser, en plus, la couleur et la taille avec des valeurs passées en paramètre. . **(2 pts)**
   c) Réécrire la méthode ***toString()*** pour afficher les caractéristiques de l'habit. . **(2 pts)**
3) Un électroménager est un article qui a un poids et une durée de garantie.
   a) Ecrire la classe « **Electroménager** » héritant de la classe « **Article** ». . **(2 pts)**
   b) Récrire le constructeur de cette classe pour définir, en plus, le poids et la durée de garantie en mois. **(2 pts)**
   c) Ajouter la méthode **datefinGarantie ()** : retourne la date de fin de la garantie à partir de la date actuelle. **(2 pts)**
   d) Réécrire la méthode ***toString()*** donnant les caractéristiques d'un électroménager et la date de fin de sa garantie à partir de la date courante. **(2 pts)**
4) **Classe Program :** Tester ces trois classes dans un programme principal.
   a) Créer un article de type habit **(1 pt)**
   b) Approvisionner le stock de cet article et l'afficher. **(1,5 pts)**
   c) Créer un article de type électroménager **(1 pt)**
   d) Effectuer un achat de cet article. **(1 pt)**
   e) Afficher la date fin de garantie de cet article. **(1 pt)**
   f) Afficher cet article. **(0,5 pt)**

➢ Dossier 3:Programmation événementielle (30 pts)

- **NB : Dans ce dossier, on vous demande de donner uniquement le code à mettre à l'intérieur des méthodes événementielles. L'entête de ces méthodes événementielles n'est pas demandé !**

Soit le formulaire suivant dont l'objectif consiste à gérer les donneurs de sang :



1. Ajouter dans la méthode de chargement du formulaire le code permettant de remplir la liste groupe sanguin par (A, B, O, AB). **(3 pts)**

2. Ecrire le code du bouton « ***Nouveau*** » qui permet d'initialiser tous les champs pour saisir un nouveau donneur. **(3 pts)**

3. Ecrire le code du bouton « ***Ajouter*** » permettant d'ajouter un nouveau donneur à la liste des donneurs (la dataGridView). **(8 pts)**

4. Ecrire le code du bouton « ***Supprimer*** » permettant de supprimer de la dataGridView un donneur dont le CIN est saisi. **(6 pts)**

   Un message demandant la confirmation doit être affiché. **(2 pts)**

5. On suppose que les donneurs sont stockées dans une collection d'objets «Donneurs » de type ArrayList ou Vector. Ecrire le code du bouton « ***Sauvegarder*** » qui permet d'enregistrer la liste des donneurs dans un fichier *objet* (sérialisation) qui porte le nom « ***Donneurs.dat*** ». **(8 pts)**