Struct Comptes{

Chaîne numeroCpte[11];

Entier etat;

Reel solde;

Chaîne typeCompte[10];

Chaîne devise[10];

};

Struct abonnes{

Chaîne numCpte[11];

Chaîne nom[50];

Chaîne prenom[50];

Chaîne sexe[8];

Chaîne adresse[50];

Chaîne tel[12];

};

Struct Employes{

Chaîne numEmp[15];

Chaîne nom[50];

Chaîne prenom[50];

Chaîne sexe[8];

Chaîne adresse[50];

Chaîne tel[12];

Reel salaire;

};

Struct transactions{

Chaîne numCompte[11];

Chaîne typeTransaction[20];

Chaîne nomAb[30];

Reel montant;

};

Struct archives{

Chaîne numEmp[15];

Chaîne numArch[15];

Chaîne nom[50];

Chaîne prenom[50];

Chaîne sexe[8];

Chaîne adresse[50];

Chaîne tel[12];

Reel salaire;

};

Struct Comptes Cp[1000000];

Struct abonnes ab[50000];

Struct Employes emp[550];

Struct transactions tr[90000000];

Struct archives arch[5000];

Var nbC<--0, nbA<--0, nbe<--0, nbt<--0, nbarch<--0: entiers;

Fonction testLongueurComptes(Chaîne: num[11]): entier;

Fonction testTypeComptes(Chaîne: type[10]): entier ;

Fonction testChampVide(Chaîne: champ[50]):entier;

Fonction testDevise(Chaîne: devise[10]): entier ;

Fonction testDoublonCode(Chaîne: num[11]): entier ;

Fonction testSexe(Chaîne: sexe[8]): entier ;

Fonction testTel(Chaîne: tel[12]): entier ;

Fonction menuGestionComptes(): entier ;

Fonction menuGestionEmployes(): entier ;

Fonction menuGestionTransactions(): entier ;

Fonction menuGestionArchives(): entier ;

Fonction menuGestionAbonnes(): entier ;

Fonction menuGestionPrincipal(): entier ;

Procédure enregistrerComptes();

Procédure listerComptes();

Procédure rechercherComptes();

Procédure modifierComptes();

Procédure supprimerComptes();

Procédure gestionComptes();

Procédure enregistrerAbonnes();

Procédure listerAbonnes();

Procédure rechercherAbonnes();

Procédure modifierAbonnes();

Procédure supprimeerAbonnes();

Procédure gestionAbonnes();

Procédure enregistrerEmployes();

Procédure listerEmployes();

Procédure rechercherEmployes();

Procédure modifierEmployes();

Procédure revoquerEmployes();

Procédure gestionEmployes();

Procédure depot();

Procédure retrait();

Procédure transfert();

Procédure listerTransaction();

Procédure listerParType();

Procédure gestionTransactions();

Procédure listerArchives();

Procédure rechercherArchives();

Procédure gestionArchives();

Procédure menuPrincipal();

Fonction menuGestionComptes(): entier;

Debut

Var ch: entier;

Afficher("Menu Gestion Comptes");

Afficher("1. Enregistrer");

Afficher("2. Lister")

Afficher("3. Rechercher")

Afficher("4. Modifier");

Afficher("5. Supprimer");

Afficher("6. Retour au menu principal");

Afficher("Faîtes votre choix");

Lire(ch);

Retourne ch;

FinFonction

Procédure gestionCompte()

Debut

Var x: entier ;

Repeter

x←menuGestionComptes();

Selon(x) faire

Cas 1: {

enregistrerComptes();

};

Cas 2: {

listerComptes();

};

Cas 3: {

rechercherComptes();

};

Cas 4: {

modifierComptes();

};

Cas 5: {

supprimerComptes();

}

Cas 6: {

menuPrincipal();

}

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à (x>=1 ou x<=6);

FinProcédure

Fonction menuGestionAbonnes(): entier

Debut

Var ch: entier;

Afficher("Menu Gestion Abonnes");

Afficher("1. Enregistrer");

Afficher("2. Lister")

Afficher("3. Rechercher")

Afficher("4. Modifier");

Afficher("5. Supprimer");

Afficher("6. Retour au menu principal");

Afficher("Faîtes votre choix");

Lire(ch);

Retourne ch;

FinFonction

Procédure gestionAbonnes()

Debut

Var x: entier ;

Repeter

x←menuGestionAbonnes();

Selon(x) faire

Cas 1: {

enregistrerAbonnes();

};

Cas 2: {

listerAbonnes();

};

Cas 3: {

rechercherAbonnes();

};

Cas 4: {

modifierAbonnes();

};

Cas 5: {

supprimeerAbonnes();

}

Cas 6: {

menuPrincipal();

}

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à (x>=1 ou x<=6);

FinProcédure

Fonction menuGestionEmployes(): entier

Debut

Var ch: entier;

Afficher("Menu Gestion Employés");

Afficher("1. Enregistrer");

Afficher("2. Lister")

Afficher("3. Rechercher")

Afficher("4. Modifier");

Afficher("5. Revoquer");

Afficher("6. Retour au menu principal");

Afficher("Faîtes votre choix");

Lire(ch);

Retourne ch;

FinFonction

Procédure gestionEmployes()

Debut

Var x: entier ;

Repeter

x←menuGestionEmployes();

Selon(x) faire

Cas 1: {

enregistrerEmployes();

};

Cas 2: {

listerEmployes();

};

Cas 3: {

rechercherEmployes();

};

Cas 4: {

modifierEmployes();

};

Cas 5: {

revoquerEmployes();

}

Cas 6: {

menuPrincipal();

}

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à (x>=1 ou x<=6);

FinProcédure

Fonction menuGestionTransactions(): entier

Debut

Var ch: entier;

Afficher("Menu Gestion Transaction");

Afficher("1. Dépôt");

Afficher("2. Retrait");

Afficher("3. Transfert");

Afficher("4. Lister");

Afficher("5. Lister par type");

Afficher("6. Retour au Menu Principal");

Afficher("Faîtes votre choix");

Lire(ch);

Retourne ch;

FinFonction

Procédure gestionTransactions()

Debut

Var x: entier ;

Repeter

x←menuGestionTransactions();

Selon(x) faire

Cas 1: {

depot();

};

Cas 2: {

retrait();

};

Cas 3: {

transfert();

};

Cas 4: {

listerTransaction();

};

Cas 5: {

listerParType();

}

Cas 6: {

menuPrincipal();

}

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à (x>=1 ou x<=6);

FinProcédure

Fonction menuGestionArchives(): entier

Debut

Var ch: entier;

Afficher("Menu Gestion Archives");

Afficher("1. Lister Archives");

Afficher("2. Rechercher Archives");

Afficher ("3. Retour au Menu Principal");

Afficher("Faîtes votre choir");

Lire(ch);

Retourne ch;

FinFonction

Procédure gestionArchives()

Debut

Var x: entier;

x←menuGestionArchives();

Repeter

Selon(x) faire

Cas 1: {

listerArchives();

};

Cas 2: {

rechercherArchives();

};

Cas 3: {

menuPrincipal();

};

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à(x>=1 ou x=<3)

FinProcédure

Fonction menuGestionPrincipal(): entier

Début

Var ch: entier;

Afficher ("Menu Principal");

Afficher("1. Gestion Comptes");

Afficher("2. Gestion abonnés");

Afficher("3. Gestion Employés");

Afficher("4. Gestion Transaction");

Afficher("5. Gestion Archives");

Afficher("Faîtes votre choix");

Lire(ch);

Retourne chr;

FinFonction

Procédure menuPrincipal();

Debut

Var x: entier;

x←menuGestionPrincipal();

Repeter

Selon(x) faire

Cas 1: {

gestionComptes();

};

Cas 2: {

gestionAbonnes();

};

Cas 3: {

gestionEmployes();

};

Cas 4: {

gestionTransactions();

};

Cas 5: {

gestionArchives();

};

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à (x>=1 ou x<=5)

FinProcédure

Fonction testLongueurComptes(Chaîne: num[11]): entier;

var r←0 : entier;

Si (longueur(num)<>10) alors

r←1;

FinSi

Si (r=1) alors

Afficher ("Le numéro de compte ne peut pas être inférieur à 10 chiffres");

FinSi

Retourne r;

FinFonction

Fonction testChampVide(Chaîne: champ[50]):entier

Debut

Var r←0:entier;

Si(code="") alors

r←1;

FinSi

Si(r=1) alors

Afficher("Le champ ne doit pas être vide");

Retourne r;

FinFonction

Fonction testDoublonCode(Chaîne: code[11]): entier

Debut

Var r←0, i: entier;

Pour i allant de 0 à nbC-1 faire

Si(Cp[i].numeroCompte=code) alors

r←1;

FinSi

FinPour

Si (r=1) alors

Afficher("Ce numéro de compte existe déjà");

FinSi

Retourne r;

FinFonction

Fonction testTypeComptes(Chaîne : compte[10]): entier

Var r←0: entier;

Si(compte<>"Epargne" ou compte<>"Courant") alors

r←1;

FinSi

Si(r=1) alors

Afficher ("Le type de compte doit être Épargne ou Courant");

FinSi

Retourne r;

FinFonction

Fonction testDevise(Chaîne : devise[10]): entier

Debut

Var r←0: entier;

Si(devise<>"USD" ou devise<>"HTG") alors

r←1;

FinSi

Si(r=1) alors

Afficher("La devise doit être USD ou HTG")

FinSi

Retourne r;

FinFonction

**Fonction testSexe(Chaîne: sexe[8]):** entier ;

Debut

Var r←1: entier;

Si(sexe<>"Féminin" ou sexe<>"Masculin") alors

r←1;

FinSi

Si(r=1) alors

Afficher("Le sexe doit être Féminin ou Masculin"

FinSi

Retourne r

FinFonction

**Fonction testTel(Chaîne: tel[12]):** entier ;

**Début**

Var r←0: entier;

Si(longueur(tel)<>12) alors

r←1;

FinSi

Si(r=1) alors

Afficher("Le numéro téléphone doit être de 11 chiffres précédé du signe +")

FinSi

Retourne r;

**FinFonction**

**Procédure enregistrerComptes();**

**Début**

Var i←nbC; rep←0: entier;

**Répéter**

**Répéter**

Afficher("Entrer le numéro compte de 10 chiffres");

Lire(Cp[i].numeroCpte);

**FRJ**

Jusqu'à(testChampVide(Cp[i].numeroCpte)=1 ou testDoublonCode(Cp[i].numeroCpte)=1 ou testLongueurComptes(Cp[i].numeroCpte)=1);

**Répéter**

Afficher ("Entrer le type de compte (Épargne ou Courant)");

Lire(Cp[i].typeCompte);

**FRJ**

Jusqu'à(testChampVide(Cp[i].typeCompte)=1 ou testTypeComptes(Cp[i].typeCompte)=1);

**Répéter**

Afficher("Entrer la devise du compte");

Lire(Cp[i].devise);

**FRJ**

Jusqu'à(testChampVide(Cp[i].devise)=1 ou testDevise(Cp[i].devise)=1)

Cp[i].etat ←0;

Cp[i].solde←0;

Afficher("Voulez-vous enregistrer un autre compte? (1. Oui 2. Non)");

Lire(rep);

i←i+1;

i← nbC+1;

**FRJ**

Jusqu'à(rep=1);

**FinProcédure**

**Procédure listerComptes()**

**Debut**

Var i←0: entier;

Pour i allant de 0 à nbC-1 faire

Afficher(Cp[i].numeroCpte);

Afficher(Cp[i].typeCompte);

Afficher(Cp[i].devise);

Afficher(Cp[i].etat);

Afficher(Cp[i].solde);

**FinPour**

**FinProcédure**

**Procédure rechercherComptes()**

**Début**

Var i, pos← -1, r←0: entier;

Var nu[11]: chaîne;

**Répéter**

Afficher("Entrer le numéro de compte que vous voulez rechercher");

Lire(num);

**FRJ**

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbC-1 faire

Si(Cp[i].numeroCpte=num) alors

pos←i;

r←1;

FinSi

FinPour

Si(r= 1) alors

Afficher(Cp[pos].numeroCpte);

Afficher(Cp[pos].etat);

Afficher(Cp[pos].typeCompte);

Afficher (Cp[pos].solde);

FinSi

Sinon

Afficher("Ce numéro de compte n'existe pas");

FinSinon

FinProcédure

**Procédure modifierComptes()**

Début

Var num[11], type[10]; devise[10], ch: chaînes;

Var r←0, pos← -1,i: entiers;

Répéter

Afficher("Entrer le numéro de compte à modifier");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbC-1 faire

Si(Cp[i].numeroCpte=num) alors

pos←i;

r←1;

FinSi

FinPour

Si(r=1) alors

Répéter

Afficher("1. Modifier type");

Afficher("2. Modifier devise");

Afficher("3. Retour au Menu Gestion Compte");

Afficher("4. Retour au Menu Principal");

Afficher("Faîtes votre choix");

Lire(ch);

Selon(ch) faire

Cas 1: {

Répéter

Afficher("Entrer le nouveau type (Épargne ou Courant)");

Lire(type);

FRJ

Jusqu'à(testChampVide(type)=1 ou testTypeComptes(type)=1);

Cp[pos].typeCompte←type;

};

Cas 2: {

Répéter

Afficher("Entrer la nouvelle devise");

Lire(devise);

FRJ

Jusqu'à(testChampVide(devise)=1 ou testDevise(devise)=1);

Cp[pos].devise←devise;

};

Cas 3: {

gestionComptes();

};

Cas 4: {

menuPrincipal();

};

Sinon

Afficher("Mauvais choix");

FinSinon

FinSelon

FRJ

Jusqu'à(ch>=1 ou ch<=4);

FinSi

Sinon

Afficher("Compte introuvable");

FinSinon

FinProcédure

Procédure supprimerComptes()

Début

Var i, r←0, pos← -1:

Var num[11]: chaîne;

Répéter

Afficher ("Entrer le numéro Compte que vous voulez supprimer");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbC-1 faire

Si(Cp[i].numeroCpte=num et Cp[i].etat =0) alors

pos←i;

r←1;

FinSi

FinPour

Si (r=1) alors

Cp[pos].numeroCpte←"";

Cp[pos].typeCompte←"";

Cp[pos].etat←0;

Cp[pos].solde←0;

Cp[pos].devise←"";

FinSi

Sinon

Afficher("Ce numéro n'existe pas");

FinSinon

FinProcédure

**Procédure enregistrerAbonnes()**

Début

Var i←nbA, j, pos← -1, ch← -1, r←0: entier;

Pour i allant de 0 à nbC-1 faire

Si(Cp[j].etat = 0) alors

pos ←j;

r←1;

FinSi

FinPour

Si(r=1) alors

ab[i].numCpte ← Cp[pos].numeroCpte;

Répéter

Afficher("Entrer le nom de l'abonné");

Lire(ab[i].nom);

FRJ

Jusqu'à(testChampVide(ab[i].nom)=1);

Répéter

Afficher("Entrer son prénom");

Lire(ab[i].prenom);

FRJ

Jusqu'à(testChampVide(ab[i].prenom)=1);

Répéter

Afficher("Entrer le sexe de l'abonné");

Lire(ab[i].sexe);

FRJ

Jusqu'à(testChampVide(ab[i].sexe)=1 ou testSexe(ab[i].sexe)=1);

Répéter

Afficher("Entrer l'adresse de l'abonné");

Lire(ab[i].adresse);

FRJ

Jusqu'à(testChampVide(ab[i].adresse)=1);

Répéter

Afficher("Entrer le numéro de téléphone de l'abonné");

Lire(ab[i].tel);

FRJ

Jusqu'à(testChampVide(ab[i].tel)=1 ou testTel(ab[i].tel)=1);

Cp[pos].etat ←1;

FinSi

i←i+1;

nba ←nba+1;

Sinon

Afficher("Il n'y a pas de compte disponible");

FinSinon

Répéter

Afficher("Pressez 1 pour aller au Menu Gestion Abonnés");

Lire(ch);

FRJ

Jusqu'à(ch=1);

Si(ch =1) alors

gestionAbonnes();

FinSi

FinProcédure

**Procédure listerAbonnes()**

Début

Var i, ch← -1: entièrs;

Pour i allant de 0 à nbA-1 faire

Afficher(ab[i].numCpte);

Afficher(ab[i].nom);

Afficher(ab[i].prenom);

Afficher(ab[i].sexe);

Afficher(ab[i].adresse);

Afficher(ab[i].tel);

FinPour

Répéter

Afficher("Pressez 1 pour aller au Menu Gestion Abonnés");

Lire(ch);

FRJ

Jusqu'à(ch=1);

Si(ch =1) alors

gestionAbonnes();

FinSi

FinProcédure

**Procédure rechercherAbonnes()**

Début

Var r← 0, pos← -1, i: entiers;

Var num[11]: chaîne;

Répéter

Afficher("Entrer le numéro de compte à rechercher ");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbA-1 faire

Si(ab[i].numCpte=num) alors

pos← i;

r← 1;

FinSi

FinPour

Si(r= 1) alors

Afficher(ab[pos].numeroCpte);

Afficher(ab[pos].etat);

Afficher(ab[pos].typeCompte);

Afficher (ab[pos].solde);

FinSi

Sinon

Afficher("Ce numéro d'abonné n'existe pas");

FinSinon

Répéter

Afficher("Pressez 1 pour aller au Menu Gestion Abonnés");

Lire(ch);

FRJ

Jusqu'à(ch=1);

Si(ch =1) alors

gestionAbonnes();

FinSi

FinProcédure

**Procédure modifierAbonnes()**

Début

Var num[11], nom[50], prenom[50], sexe[8], adresse[50], tel[12], ch: chaînes;

Var r←0, pos← -1,i: entiers;

Répéter

Afficher("Entrer le numéro de compte à modifier");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbA-1 faire

Si(ab[i].numCpte=num) alors

pos←i;

r←1;

FinSi

FinPour

Si(r=1) alors

Répéter

Afficher("1. Modifier nom");

Afficher("2. Modifier prénom");

Afficher("3. Modifier adresse");

Afficher("4. Modifier téléphone");

Afficher("5. Modifier sexe");

Afficher("6. Retour au Menu Gestion Abonnés");

Afficher("7. Retour au Menu Principal");

Afficher("Faîtes votre choix");

Lire(ch);

Selon(ch) faire

Cas 1: {

Répéter

Afficher("Entrer le nouveau nom ");

Lire(nom);

FRJ

Jusqu'à(testChampVide(nom)=1)

ab[pos].nom← nom;

};

Cas 2: {

Répéter

Afficher("Entrer le nouveau prénom");

Lire(prenom);

FRJ

Jusqu'à(testChampVide(prenom)=1);

ab[pos].prenom ← prenom;

};

Cas 3 : {

Répéter

Afficher ("Entrer la nouvelle adresse");

Lire(adresse);

FRJ

Jusqu'à(testChampVide(adresse)=1);

ab[pos].adresse ← adresse;

};

Cas 4 : {

Répéter

Afficher("Entrer le nouveau numéro")

Lire(numero);

FRJ

Jusqu'à(testChampVide(numero)=1 ou testTel(numero)=1);

ab[pos].tel ← numero;

};

Cas 5: {

Répéter

Afficher("Entrer le sexe");

Lire(sexe);

FRJ

Jusqu'à(testChampVide(sexe)=1 ou testSexe(sexe)=1);

ab[pos]. sexe ← sexe;

};

Cas 6: {

gestionAbonnes();

};

Cas 7: {

menuPrincipal();

};

FinSi

Sinon

Afficher("Mauvais choix");

FinSelon

FRJ

Jusqu'à(ch>=1 ou ch<=7);

FinSi

Sinon

Afficher("Ce numéro d'abonné n'existe pas");

FinSinon

FinProcédure

**Procédure supprimerAbonnes()**

Début

Var i, j, pos ← -1, posC ← -1, r←0, s←0: entier;

Var num[11]: chaîne;

Répéter

Afficher("Entrer le numéro de l'abonné à supprimer");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nba-1 faire

Si(ab[i].numCpte=num) alors

pos ← i;

r← 1;

FinSi

FinPour

Si (r=1) alors

Pour j allant de 0 à nbC-1 faire

Si(Cp[j].numeroCpte=num) alors

posC ←j;

s← 1;

FinSi

FinPour

Si (s=1) alors

Si (Cp[posC].solde>500) alors

Afficher("Désolé, vous devrez d'abord effectuer un retrait avant de supprimer le compte");

FinSi

Si(Cp[posC].solde=500) alors

ab[pos].numCpte ←"";

ab[pos].nom←"";

ab[pos].prenom←"";

ab[pos].adresse ←"";

ab[pos].tel ←"";

ab[pos].sexe←"";

Cp[posC].etat ←0;

FinSi

FinSi

FinSi

Sinon

Afficher("Ce numéro d'abonné n'existe pas");

FinSinon

FinProcédure

**Procédure enregistrerEmployes()**

Début

Var i←nbe, rep←0: entiers;

Répéter

Répéter ("Entrer le numéro de l'employé");

Lire(emp[i].numEmp);

FRJ

Jusqu'à(testChampVide(emp[i].numEmp)=1 ou testLongueurComptes(emp[i].numEmp)=1);

Répéter

Afficher ("Entrer le nom de l'employé");

Lire(emp[i].nom);

FRJ

Jusqu'à(testChampVide(nom)=1);

Répéter

Afficher("Entrer le prénom de l'employé");

Lire(emp[i].prenom);

FRJ

Jusqu'à(testChampVide(emp[i].prenom)=1);

Répéter

Afficher("Entrer le sexe de l'employé");

Lire(emp[i].sexe);

FRJ

Jusqu'à(testChampVide(emp[i].sexe)=1 ou testSexe(emp[i].sexe)=1);

Répéter

Afficher("Entrer l'adresse de l'employé");

Lire(emp[i].adresse);

FRJ

Jusqu'à(testChampVide(emp[i].adresse)=1);

Répéter

Afficher("Entrer le numéro téléphone de l'employé");

Lire(emp[i].tel);

FRJ

Jusqu'à(testChampVide(emp[i].tel)=1 ou testTel(emp[i].tel)=1);

Répéter

Afficher("Entrer le salaire de l'employé");

Lire(emp[i].salaire);

FRJ

Jusqu'à(testChampVide(emp[i].salaire)=1 ou emp[i].salaire<=0);

Afficher("Voulez-vous enregistrer un autre employé ? (1. Oui 2. Non ");

Lire(rep)

i←i+1;

nbe ← nbe+1;

Jusqu'à(rep=1);

FinProcédure

**Procédure listerEmployes()**

Var i: entier;

Pour i allant de 0 à nbe-1 faire

Afficher(emp[i].numEmp);

Afficher(emp[i].nom);

Afficher(emp[i].prenom);

Afficher(emp[i].sexe);

Afficher(emp[i].tel);

Afficher(emp[i].salaire);

Afficher(emp[i].adresse);

FinPour

FinProcédure

**Procédure rechercherEmployes()**

Début

Var num[11]: chaîne;

Var i, pos← -1; r←0: entiers;

Répéter

Afficher("Entrer le numéro de l'employé");

Lire(num)

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbe-1 faire

Si(emp[i].numEmp=num) alors

pos← i;

r← 1;

FinSi

FinPour

Si(r=1) alors

Afficher(emp[pos].numEmp);

Afficheremp[pos].nom);

Afficher(emp[pos].prenom);

Afficher(emp[pos].sexe);

Afficher(emp[pos].salaire);

Afficher(emp[pos].salaire);

Afficher(emp[pos].adresse);

FinSi

Sinon

Afficher("Ce numéro d'abonné n'existe pas");

FinSinon

FinProcédure

**Procédure modifierEmployes()**

Début

Var num[11], nom[50], prenom[50], adresse[50], tel[12], sexe[8]: chaînes;

Var r←0, pos← -1,i: entiers;

Répéter

Afficher("Entrer le numéro de compte à modifier");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbe-1 faire

Si(emp[i].numCpte=num) alors

pos←i;

r←1;

FinSi

FinPour

Si(r=1) alors

Répéter

Afficher("1. Modifier nom");

Afficher("2. Modifier prénom");

Afficher("3. Modifier adresse");

Afficher("4. Modifier téléphone");

Afficher("5. Modifier sexe");

Afficher("6. Modifier salaire");

Afficher("7. Retour au Menu Gestion Employés");

Afficher("8. Retour au Menu Principal");

Afficher("Faîtes votre choix");

Lire(ch);

Selon(ch) faire

Cas 1: {

Répéter

Afficher("Entrer le nouveau nom ");

Lire(nom);

FRJ

Jusqu'à(testChampVide(nom)=1)

emp[pos].nom← nom;

};

Cas 2: {

Répéter

Afficher("Entrer le nouveau prénom");

Lire(prenom);

FRJ

Jusqu'à(testChampVide(prenom)=1);

emp[pos].prenom ← prenom;

};

Cas 3 : {

Répéter

Afficher ("Entrer la nouvelle adresse");

Lire(adresse);

FRJ

Jusqu'à(testChampVide(adresse)=1);

emp[pos].adresse ← adresse;

};

Cas 4 : {

Répéter

Afficher("Entrer le nouveau numéro")

Lire(numero);

FRJ

Jusqu'à(testChampVide(numero)=1 ou testTel(numero)=1);

emp[pos].tel ← numero;

};

Cas 5: {

Répéter

Afficher("Entrer le sexe");

Lire(sexe);

FRJ

Jusqu'à(testChampVide(sexe)=1 ou testSexe(sexe)=1);

emp[pos].sexe ← sexe;

};

Cas 6: {

Répéter

Afficher("Entrer le nouveau salaire");

Lire(salaire);

FRJ

Jusqu'à(testChampVide(salaire)=1 ou salaire<=0);

emp[po].salaire← salaire;

Cas 7: {

gestionEmployes();

};

Cas 8: {

menuPrincipal();

};

FinSi

Sinon

Afficher("Mauvais choix");

FinSelon

FRJ

Jusqu'à(ch>=1 ou ch<=8);

FinSi

Sinon

Afficher("Ce numéro d'employés n'existe pas");

FinSinon

FinProcédure

**Procédure revoquerEmployes()**

Début

Var num[15]: chaîne;

Var i, r←0, j, pos← -1: entiers;

Répéter

Répéter

Afficher("Entrer le numéro de l'employé à révoquer");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour i allant de 0 à nbe-1 faire

Si(emp[i].numEmp=num) alors

pos←1;

r←1;

FinSi

FinPour

Si(r=1) alors

arch[j].numEmp←emp[pos].numEmp;

arch[j].nom←emp[pos].nom;

arch[j].prenom←emp[pos].prenom;

arch[j].sexe←emp[pos].sexe;

arch[j].adresse←emp[pos].adresse;

arch[j].salaire←emp[pos].salaire;

arch[j].tel←emp[pos].tel;

Répéter

Afficher("Entrer le numéro de l'archive");

Lire(arch[j].numArch);

FRJ

Jusqu'à(testChampVide(arch[j].numArch)=1 ou testDoublonCode(arch[j].numArch)=1);

emp[pos].numEmp←"";

emp[pos].nom←"";

emp[pos].prenom←"";

emp[pos].adresse←"";

emp[pos].sexe←"";

emp[pos].tel←"";

emp[pos].salaire←0;

FinSi

Sinon

Afficher("Ce numéro n'existe pas");

FinSinon

j←j+1;

nbarch← nbarch+1;

Afficher("Pressez : 1. Menu Gestion Employés, 2. Menu Principal");

Lire(rep);

FRJ

Jusqu'à(testChampVide(rep)=1 ou rep=1 ou rep=2);

Selon(rep) faire

Cas 1: {

gestionEmployes();

};

Cas 2: {

menuPrincipal();

};

Sinon

Afficher("");

FinSinon

FinSelon

FinProcédure

**Procédure depot()**

Var i←nbt; pos← -1; posa← -1; r←0; t←0, rep←0; j: entiers;

Var num[11]: chaîne;

Var montant: reel;

Répéter

Afficher("Entrer le numéro du compte");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1 ou testLongueurComptes(num)=1);

Pour j allant de 0 à nbC-1 faire

Si(Cp[i].numeroCpte=num) alors

pos← 1;

r← 1;

FinSi

FinPour

Si(r=1) alors

Répéter

Afficher("Vous avez quelle monnaie en votre possession ? (1. HTG 2. USD)");

Lire(rep);

FRJ

Jusqu'à(testChampVide(rep)=1 ou rep=1 ou rep=2);

Si (rep=1) alors

Répéter

Afficher("Entrer le montant à déposer");

Lire(montant);

Si(montant<500) alors

Afficher("Vous ne pouvez pas déposer un montant inférieur à 500 HTG");

FinSi

FRJ

Jusqu'à(testChampVide(montant)=1 ou montant<500);

Si(Cp[pos].devise="HTG") alors

Cp[pos].solde ← Cp[pos].solde + montant;

FinSi

Sinon

montant←montant/180;

Cp[pos].solde← Cp[pos].solde+montant;

FinSinon

FinSi

Sinon

Répéter

Afficher("Entrer le montant à déposer");

Lire(montant);

Si(montant<5) alors

Afficher("Vous ne pouvez pas déposer moins que 5 $");

FRJ

Jusqu'à(testChampVide(montan)=1 ou montant<5);

Si(Cp[pos].devise="USD") alors

Cp[pos].solde←Cp[pos].solde + montant;

FinSi

Sinon

montant←montant\*180;

Cp[pos].solde← Cp[pos].solde + montant;

FinSinon

FinSi

Sinon

Afficher("Ce numéro n'existe pas");

FinSinon

Pour j allant de 0 à nbA-1 faire

Si(ab[j].numCpte=num) alors

posa←1;

FinSi

FinPour

tr[i].numCompte←Cp[pos].numCpte;

tr[i].typeTransaction←"Dépôt";

tr[i].nomAb←ab[posa].nom;

tr[i].montant← montant;

i←i+1;

nbt←nbt+1;

FinProcédure

**Procédure retrait()**

Début

Var num[11]: chaîne;

Var i←nbt, j, pos← -1, pasA← -1, posa: entiers;

Var montant: réel;

Répéter

Afficher ("Entrer le numéro du compte");

Lire(num);

FRJ

Jusqu'à(testChampVide (num)=1 ou testLongueurComptes(num)=1)

Pour j allant de 0 à nbC-1 faire

Si(Cp[j].numeroCpte=num) alors

pos←1;

r←1;

FinSi

FinPour

Si(r=1) alors

Si(Cp[pos].devise="HTG") alors

Répéter

Répéter

Afficher("Entrer le montant");

Lire(montant);

Si(montant<500) alors

Afficher("Vous ne pouvez pas retirer moins que 500 gourdes")

FinSi

FRJ

Jusqu'à(montant<500);

Si(montant+500>Cp[pos].solde) alors

Afficher("Le compte est insuffisant, après le retrait le compte ne doit pas être à moins que 500 gourdes");

FRJ

Jusqu'à(montant+500<Cp[pos].solde);

Cp[pos].solde←Cp[pos].solde - montant;

FinSi

Sinon

Si(Cp[pos].devise="USD") alors

Répéter

Répéter

Afficher("Entrer le montant en dollars");

Lire(montant);

Si(montant<5) alors

Afficher("Vous ne pouvez pas retire moins que 500 USD");

FinSi

FRJ

Jusqu'à(montant<5);

Si(montant+5<Cp[pos].solde) alors

Afficher("Le compte est insuffisant, après le retrait le compte ne doit pas être plus à moins que 5 USD");

FRJ

Jusqu'à(montant+5<Cp[pos].solde)

Cp[pos].solde←Cp[pos].solde - montant;

FinSi

FinSinon

FinSi

Sinon

Afficher("Ce compte n'existe pas");

FinSinon

Pour j allant de 0 à nbA-1 faire

S(Cp[pos].numCpte=num) alors

posA←1;

FinSi

FinPour

tr[i].numCompte←Cp[pos].numCpte;

tr[i].typeTransaction←"Retrait";

tr[i].nomAb←ab[posa].nom;

tr[i].montant← montant;

i←i+1;

nbt←nbt+1;

FinProcédure

**Procédure transfert()**

Début

Var num1[15], num2[15]: chaîne;

Var i←nbt, j, k, posC1← -1, posC2← -1, posA1← -1, posA2← -1, t←0, r←0:entiers;

Var montant: réel;

Répéter

Afficher("Entrer le numéro du premier Compte")

Lire(num1);

FRJ

Jusqu'à(testChampVide(num1)=1 ou testLongueurComptes(num1)=1);

Pour j allant de 0 à nbC-1 faire

Si(Cp[j].numCpte=num1) alors

posC1←1;

r← 1;

FinSi

FinPour

Si(r=1) alors

Répéter

Afficher("Entrer le numéro du second compte");

Lire(num2);

FRJ

Jusqu'à(testChampVide(num2)=1 ou testLongueurComptes(num2)=1);

Pour j allant de 0 à nbC-1 faire

Si[Cp[j].numCpte=num2) alors

posC2←1;

t← 1;

FinSi

FinPour

Si(t=1) alors

Si(Cp[posC1).devise = "HTG") alors

Répéter

Répéter

Afficher("Entrer le montant à enlever");

Lire(montant);

Si(montant<500) alors

Afficher("Vous pouvez pas transférer moins que 500 HTG");

FinSi

FRJ

Jusqu'à(testChampVide(montant)=1 ou montant<500);

Si(Cp[posC1].solde<montant+500) alors

Afficher("Vous ne pouvez pas transférer cette somme, le compte est insuffisant");

FinSi

FRJ

Jusqu'à(Cp[posC1].solde<montant+500);

Cp[posC1].solde←Cp[posC1].solde - montant;

Si(Cp[posC2].devise="HTG") alors

Cp[posC2].solde← Cp[posc2] + montant;

FinSi

Sinon

Si(Cp[posC2].devise="USD") alors

montant←montant/180;

Cp[posC2].solde←Cp[posC2] + montant;

FinSi

FinSinon

FinSi

Si(Cp[posC1].devise="USD") alors

Répéter

Répéter

Afficher("Entrer le montant à enlever");

Lire(montant);

Si(montant<5) alors

Afficher("Vous pouvez pas transférer moins que 5 USD");

FinSi

FRJ

Jusqu'à(testChampVide(montant)=1 ou montant<5);

Si(Cp[posC1].solde<montant+5) alors

Afficher("Vous ne pouvez pas transférer cette somme, le compte est insuffisant");

FinSi

FRJ

Jusqu'à(Cp[posC1].solde<montant+5);

Cp[posC1].solde←Cp[posC1].solde - montant;

Si(Cp[posC2].devise="USD") alors

Cp[posC2].solde←Cp[posC2].solde + montant;

FinSi

Sinon

Si(Cp[posC2].devise="HTG") alors

montant←montant\*180;

Cp[posC2].solde←Cp[posC2].solde + montant;

FinSi

FinSinon

FinSi

FinSi

Sinon

Afficher("Ce deuxième compte n'existe pas");

FinSinon

FinSi

Sinon

Afficher("Ce premier compte n'existe pas");

FinSinon

Pour j allant de 0 à nbA-1 faire

Si(Cp[j].numCpte=num1) alors

posA1← j;

FinSi

FinPour

Pour j allant de 0 à nbA-1 faire

Si(Cp[k].numCpte=num2) alors

posA2←1;

FinSi

FinPour

tr[i].numCompte←Cp[posC1].numCpte;

tr[i].typeTransaction←"Transfert";

tr[i].nomAb←ab[posA1].nom;

tr[i].montant← montant;

i←i+1;

nbt←nbt+1;

tr[i].numCompte←Cp[pos].numCpte;

tr[i].typeTransaction←"Dépôt";

tr[i].nomAb←ab[posA2].nom;

tr[i].montant← montant;

i←i+1;

nbt←nbt+1;

FinProcédure

**Procédure listerTransactions()**

Début

Var i: entier;

Pour i allant de 0 à nbt-1 faire

Afficher(tr[i].numCpte);

Afficher(tr[i].typeTransaction);

Afficher(tr[i].nomAb);

Afficher(tr]i].montant);

FinPour

FinProcédure

**Procédure listerParType()**

Début

Var i: entier;

Var type[20]: chaîne;

Répéter

Afficher("Entrer le type de transaction(Dépôt, Retrait, Transfert)");

Lire(type);

FRJ

Jusqu'à(type<>"Dépôt" ou type<>"Retrait" ou type<>"Transfert");

Pour i allant de 0 à nbt-1 faire

Si(tr[i].typeTransaction=type) alors

Afficher(tr[i].numCpte);

Afficher(tr[i].typeTransaction);

Afficher(tr[i].nomAb);

Afficher(tr]i].montant);

FinSi

FinPour

FinProcédure

Procédure listerArchives()

Début

Var i: entier;

Pour i allant de 0 à nbarch-1 faire

Afficher(arch[i].numEmp);

Afficher(arch[i].numaArchr);

Afficher(arch[i].nom);

Afficher(arch[i].prenom);

Afficher(arch[i].sexe);

Afficher(arch[i].adresse);

Afficher(arch[i].tel);

Afficher(arch[i].salaire);

FinPour

FinProcédure

**Procédure rechercherArchives()**

Début

Var i, r←0, pos← -1: entiers;

Var num[15]: chaîne;

Répéter

Afficher("Entrer le numéro numéro archive à rechercher");

Lire(num);

FRJ

Jusqu'à(testChampVide(num)=1);

Pour i allant de 0 à nbarch-1 faire

Si(arch[i].numArch=num) alors

pos← i;

r←1;

FinSi

FinPour

Si(r=1) alors

Afficher(arch[pos].numEmp);

Afficher(arch[pos].numaArchr);

Afficher(arch[pos].nom);

Afficher(arch[pos].prenom);

Afficher(arch[pos].sexe);

Afficher(arch[pos].adresse);

Afficher(arch[pos].tel);

Afficher(arch[pos].salaire);

FinSi

Sinon

Afficher("Ce numéro n'existe pas");

FinSinon

FinProcédure

Début

menuPrincipal();

Fin