|  |  |
| --- | --- |
| ***OFPPT*** | 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 Juin 2016 (Correction)** |

|  |  |
| --- | --- |
| ***Filière : Techniques de Développement Informatique*** | ***Epreuve : Synthèse*** |
| ***Niveau: TS*** | ***Variante : V1*** |
| ***Durée : 5 heures*** | ***Barème : / 120*** |

* **Partie I : Théorie** (40 pts)
* **Dossier 1: Notions de mathématiques appliquées à l’informatique (12 pts)**

1. Convertir en binaire les nombres suivants  ***(06 pts)***

(145)8: 1100101

(A4BE)16 : 1010010010111110

(59)10: 111011

1. Effectuer  en binaire l’opération suivante ***(02 pts)***

1011110\*11 = 100011010

1. A l’aide du tableau de Karnaugh, simplifier la fonction F définie par sa table de vérité suivante : **(*04* pts)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EF  G | 00 | 01 | 11 | 10 |
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |

H= E/F+ FG+/F /G

* **Dossier 2: Techniques de programmation structurée (8 pts)**

|  |
| --- |
| constante A =12  constante B=10  **debut**  i,j,T1[ A],T2[B],res[B][A] : entier ;  i=0;  pour i allant de 0 à A  lire (T1[i]);  pour i allant de 0 à B  lire (T2[j]);  max= T1[j] ;  pour i allant de 0 à A  faire  if(max > T1[i] );  max= T1[i]  fin pour  pour i allant de 0 à B  faire  if(max > T2[i] );  max= T2[i] ;  fin pour  pour i allant de 0 à B  pour j allant de 0 à A  res[i][j]=T1[i]\*T2[j]\*max;  pour i allant de 0 à B  pour j allant de 0 à A  ecrire(res[i][j]);    **Fin** |

* **Dossier 3: Conception et modélisation d'un système d'information (20 pts)**

1. Etablir le modèle conceptuel des données correspondant 
2. Etablir le modèle logique des données associé.



* **Partie II: Pratique** (80 pts)
* **Dossier 1: Langage de programmation structurée (25 pts)**

1. Définir une structure chargeDéplacement pouvant contenir ces informations

|  |
| --- |
| typedef struct chargeDéplacement {  int identifiant;  char mission[30];  char lieu[30];  int duree;  int montant;  char statut[30];  }nf; |

2-

***chargeDéplacement HistoriqueChargeDéplacement [20];***

3-

|  |
| --- |
| scanf("%d",&n);  for(j=0;j<n;j++){  printf("info du chargeDéplacement %d\n",j+1);  printf("donner le identifiant de note de frais\n");  scanf("%d",&HistoriqueChargeDéplacement[j].identifiant);  printf("donner la mission\n");  scanf("%s",HistoriqueChargeDéplacement[j].mission);  printf("donner le lieu\n");  scanf("%s",HistoriqueChargeDéplacement[j].lieu);  printf("donner la duree de note de frais\n");  scanf("%d",&HistoriqueChargeDéplacement[j].duree);  printf("donner le montant de note de frais\n");  scanf("%d",&HistoriqueChargeDéplacement[j].montant);  printf("donner le statut de chargeDéplacement \n");  scanf("%s",HistoriqueChargeDéplacement[j].statut);  } |

4-

|  |
| --- |
| for(j=0;j<n;j++){  printf("info du chargeDéplacement %d\n",j+1);  printf("le nom "%s ",HistoriqueChargeDéplacement[j].mission);  printf("le lieu %s", HistoriqueChargeDéplacement[j].lieu);  printf("le identifiant de chargeDéplacement %d", HistoriqueChargeDéplacement [j].identifiant);  printf("la duree %d", HistoriqueChargeDéplacement[j].duree);  printf("le montan %d", HistoriqueChargeDéplacement[j].montant);  printf("le type de chargeDéplacement %s", HistoriqueChargeDéplacement[j].statut);  } |

5-

|  |
| --- |
| for(j=0;j<n;j++){  if(HistoriqueChargeDéplacement[j].statut =="en cours")  {  printf("info du chargeDéplacement %d\n",j+1);  printf("le nom "%s ",HistoriqueChargeDéplacement[j].mission);  printf("le lieu %s", HistoriqueChargeDéplacement[j].lieu);  printf("le identifiant de chargeDéplacement %d", HistoriqueChargeDéplacement [j].identifiant);  printf("la duree %d", HistoriqueChargeDéplacement[j].duree);  printf("le montan %d", HistoriqueChargeDéplacement[j].montant);  printf("le type de chargeDéplacement %s", HistoriqueChargeDéplacement[j].statut);  }  } |

6-

|  |
| --- |
| int totmontant=0 ;  for(j=0;j<n;j++){  totmontant+= HistoriqueChargeDéplacement[j].montant;  }  printf(" le montant totale des chargeDéplacement est: %d\n", totmontant); |

7-

|  |
| --- |
| printf("donner le numero et le montant de note de frais \n");  scanf("%d%d",&num,&mnt);  for(j=0;j<n;j++){  if((HistoriqueChargeDéplacement [j].mission,identifiant)==num)  HistoriqueChargeDéplacement [j].montant=mnt;  } |

8-

|  |
| --- |
| printf("donner le identifiant de note de frais\n");  scanf("%d",&HistoriqueChargeDéplacement[j].identifiant);  printf("donner la mission\n");  scanf("%s",HistoriqueChargeDéplacement[j].mission);  printf("donner le lieu\n");  scanf("%s",HistoriqueChargeDéplacement[j].lieu);  printf("donner la duree de note de frais\n");  scanf("%d",&HistoriqueChargeDéplacement[j].duree);  printf("donner le montant de note de frais\n");  scanf("%d",&HistoriqueChargeDéplacement[j].montant);  printf("donner le statut de chargeDéplacement \n");  scanf("%s",HistoriqueChargeDéplacement[j].statut);  i=0;  while(HistoriqueChargeDéplacement[i].identifiant<ident) i++;  for(j=n-1;j>=i;j--){  HistoriqueChargeDéplacement [j+1]= HistoriqueChargeDéplacement [j];  }  strcpy(HistoriqueChargeDéplacement [i].mission,mission);  strcpy(HistoriqueChargeDéplacement [i].lieu,lieu);  HistoriqueChargeDéplacement [i].identifiant=identifiant;  printf("le nouveau tableau des notes de frais est:\n");  n++;  for(j=0;j<n;j++){  printf("info du chargeDéplacement %d\n",j+1);  printf("le nom "%s ",HistoriqueChargeDéplacement[j].mission);  printf("le lieu %s", HistoriqueChargeDéplacement[j].lieu);  printf("le identifiant de chargeDéplacement %d", HistoriqueChargeDéplacement [j].identifiant);  printf("la duree %d", HistoriqueChargeDéplacement[j].duree);  printf("le montan %d", HistoriqueChargeDéplacement[j].montant);  printf("le type de chargeDéplacement %s", HistoriqueChargeDéplacement[j].statut);  } |

9-

|  |
| --- |
| do{  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 1\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 2\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 3\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 4\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 5\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 6\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 7\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 8\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUESTION 9\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*QUITTER 0\*\*\*\*\*\*\*\*\*\*\*\*\*\*:\n");  scanf("%d",&choix);  switch(choix){  case 1:  case 2:  etc…  }while(choix!=0); |

* **Dossier 2: Langage de programmation Orientée Objet (30 pts)**

1. Classe **Salarié**. **(4 pts)**

class Salarie

{

int identifiant { get; set; }

public string Nom { get; set; }

public string Prenom { get; set; }

public string Adresse { get; set; }

public string Genre { get; set; }

public float Age { get; set; }

private string service { get; set; }

private string departement { get; set; }

public string Ville { get; set; }

public Salarie() { }

public Salarie(int id, string nom, string prenom, string adresse, string Genre, float Age, string service, string dep)

{

this.identifiant = id;

this.Nom = nom;

this.Prenom = prenom;

this.Adresse = adresse;

this.Genre = Genre;

this.Age = Age;

this.service = service;

this.departement = dep;

}

public override string ToString()

{

return "id:"+ this.identifiant + " Nom\n" + this.Nom + "Prenom\n" + this.Prenom+"Adresse:"+ this.Adresse + " Genre\n" + this.Genre + "Age\n" +this.Age+ "service:"+ this.service + " Departement\n" + this.departement ;

}

}

1. Classe **Dépense**: **(4 pts)**

class Dépense

{

int Numero;

string Libellé;

string Lieu;

string Commentaire;

float Montant;

public Dépense() { }

public Dépense(int num, string libelle, string lieu, string commentaire, float montant)

{

this.Numero = num;

this.Libellé = libelle;

this.Lieu = lieu;

this.Commentaire = commentaire ;

this.Montant = montant ;

}

public virtual int CalculerCharge ()

{ return montant \* taux ; }

public override string ToString()

{

return "Numero:" + this.Numero + " Libellé\n" + this.Libellé + "Lieu\n" + this.Lieu + "Commentaire:" + this.Commentaire + " Montant\n" + this.Montant;

}

}

1. Classe **ChargeDéplacementVoiture**: **(7 pts)**

class cheveauException : Exception

{

public cheveauException()

: base("le nombre de de chevaux ne doit pas etre inférieur à 6 ou supérieur à 14")

{

}

}

class ChargeDéplacementVoiture : Dépense

{

string marque;

int nombrechevaux;

string typecarburant;

string numplaque;

public ChargeDéplacementVoiture():base() { }

public ChargeDéplacementVoiture(int num, string libelle, string lieu, string commentaire, float montant , string marque, int nombrechevaux, string typecarburant,string numplaque): base ( num, libelle, lieu, commentaire, montant )

{

if ((nombrechevaux < 6) || (nombrechevaux > 14))

{ throw new cheveauException (); }

else

{

this.marque = marque ;

this.nombrechevaux = nombrechevaux ;

this.typecarburant = typecarburant ;

this.numplaque = numplaque ;

}

}

public override int CalculerCharge ()

{

return Kilométrage \* 11 ;

}

public override string ToString()

{

return base.ToString() + "marque:" + this.marque + " nombrechevaux\n" + this.nombrechevaux + "typecarburant\n" + this.typecarburant + "numplaque:" + this.numplaque;

}

}

1. Classe **ListeChargeDéplacement**: **(10 pts)**

class ListeChargeDéplacement

{

public List<Dépense> lp;

public ListeChargeDéplacement()

{

lp = new List<Dépense>();

}

public void ajouter(Dépense dep)

{

Console.WriteLine("Confirmer l'ajout en tappant le chiffre 1");

int rep=Convert.ToInt32( Console.ReadLine());

if (rep==1)

lp.Add(dep);

}

public void Afficher()

{

foreach (Dépense dep in lp)

dep.ToString();

}

public void supprimer(Dépense dep)

{

foreach (Dépense d in lp)

{

if (d.Equals(dep))

{

Console.WriteLine("Confirmer la suppression en tappant le chiffre 1");

int rep = Convert.ToInt32(Console.ReadLine());

if (rep == 1)

lp.Remove(dep); break;

}

}

}

public void Rechercher()

{

foreach (Dépense d in lp)

{

if (d.Montan > 1000 )

{

d.ToString();

}

}

}

}

***Dossier 2 : (25 Pts)***

1. Ecrire le code du bouton ***Enregistrer*** permettant d’enregistrer la liste des **charges de déplacement** dans un fichier texte. **(6 pts)**

private void buttonEnregistrer\_Click(object sender, EventArgs e)

{

FileStream fichier = File.Open("fich.xml", FileMode.Create);

XmlSerializer s = new XmlSerializer(typeof(ListeChargeDéplacement));

s.Serialize(fichier, this);

fichier.Close();

}

1. Ecrire le code du bouton ***Afficher*** permettant d’afficher dans la grille les charges de déplacement d’un Lieu sélectionnée à partir de la zone de liste. **(6 pts)**

public List<Dépense> lp;

private void buttonAfficher\_Click(object sender, EventArgs e)

{

foreach (Dépense d in lp)

{

if (d.Lieu1 == comboBox1.Text)

{

dataGridView1.Rows.Add(d.Numero1,d.Montan,d.Lieu1, d.Libellé1 ,d.Commentaire1);

}

}

}

1. Ecrire le code nécessaire pour le bouton **Supprimer** qui permet de supprimer la charge de déplacement dont le numéro est saisie dans le textbox, la suppression doit être effectuée à la fois dans la liste et dans la grille et un message de confirmation doit être affiché avant de procéder à la suppression. **(7 pts)**

private void ButtonSupprimer\_Click(object sender, EventArgs e)

{

bool t = false;

for (int i = 0; i < lp.Count; i++)

{

if (lp[i].Numero1 == Convert .ToInt32 ( textBox1.Text))

{

t = true;

DialogResult res = MessageBox.Show("Etes vous sur de vouloir effectuer la suppression?", "Confirmation", MessageBoxButtons.YesNo);

if (res == DialogResult.Yes)

{

for (int j = 0; j < dataGridView1.RowCount; j++)

{

if (Convert .ToInt32 ( dataGridView1.Rows[j].Cells[0].Value) == Convert .ToInt32 ( textBox1.Text))

dataGridView1.Rows.RemoveAt(j);

}

//dataGridView1.Rows.RemoveAt(i);

listBox1.Items.Remove(lp[i]);

lp.RemoveAt(i);

clear(); break;

}

}

}

if (t == false) MessageBox.Show("note de frais inexistante");

}

1. Ecrire le code nécessaire pour le bouton **Total** permettant d’afficher le montant total des charges de déplacement enregistrées. **(6 pts)**

private void buttonTotal\_Click(object sender, EventArgs e)

{

float total = 0.0;

foreach (Dépense d in lp)

{

total += d.Montan;

}

TextBoxTotal = total.ToString();

}