République Tunisienne

Ministère de l'Enseignement Supérieur et
de la RechercheScientifique

Université de Carthage

Ecole Nationale d'Ingénieurs de Carthage

RAPPORT DE PROJET DE
C++

Réalisé Par

Bouebsa Abed Bouebsa

Group A

Gestion du Centre de Visite des Voitures

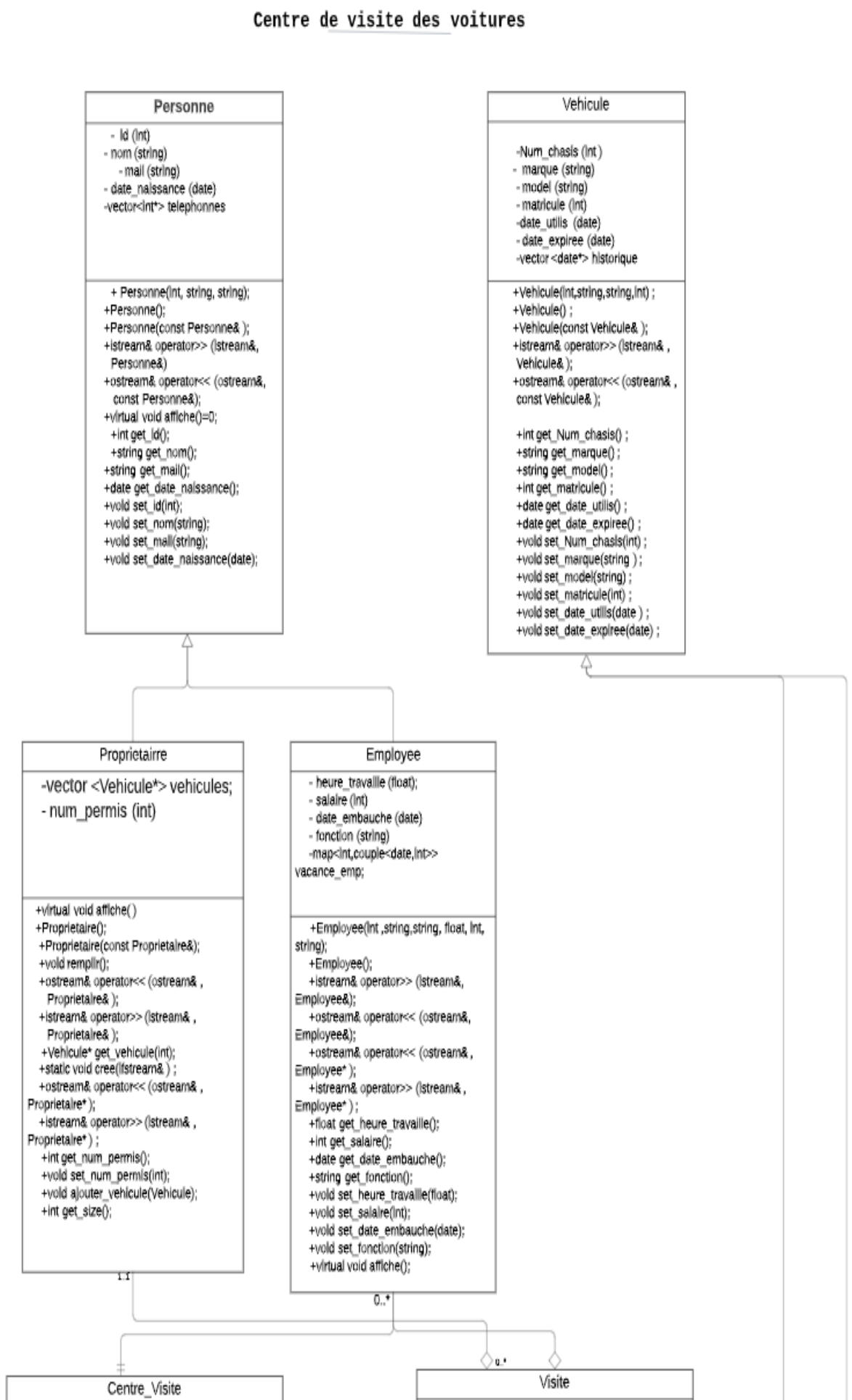
Année Universitaire : 2022-2023

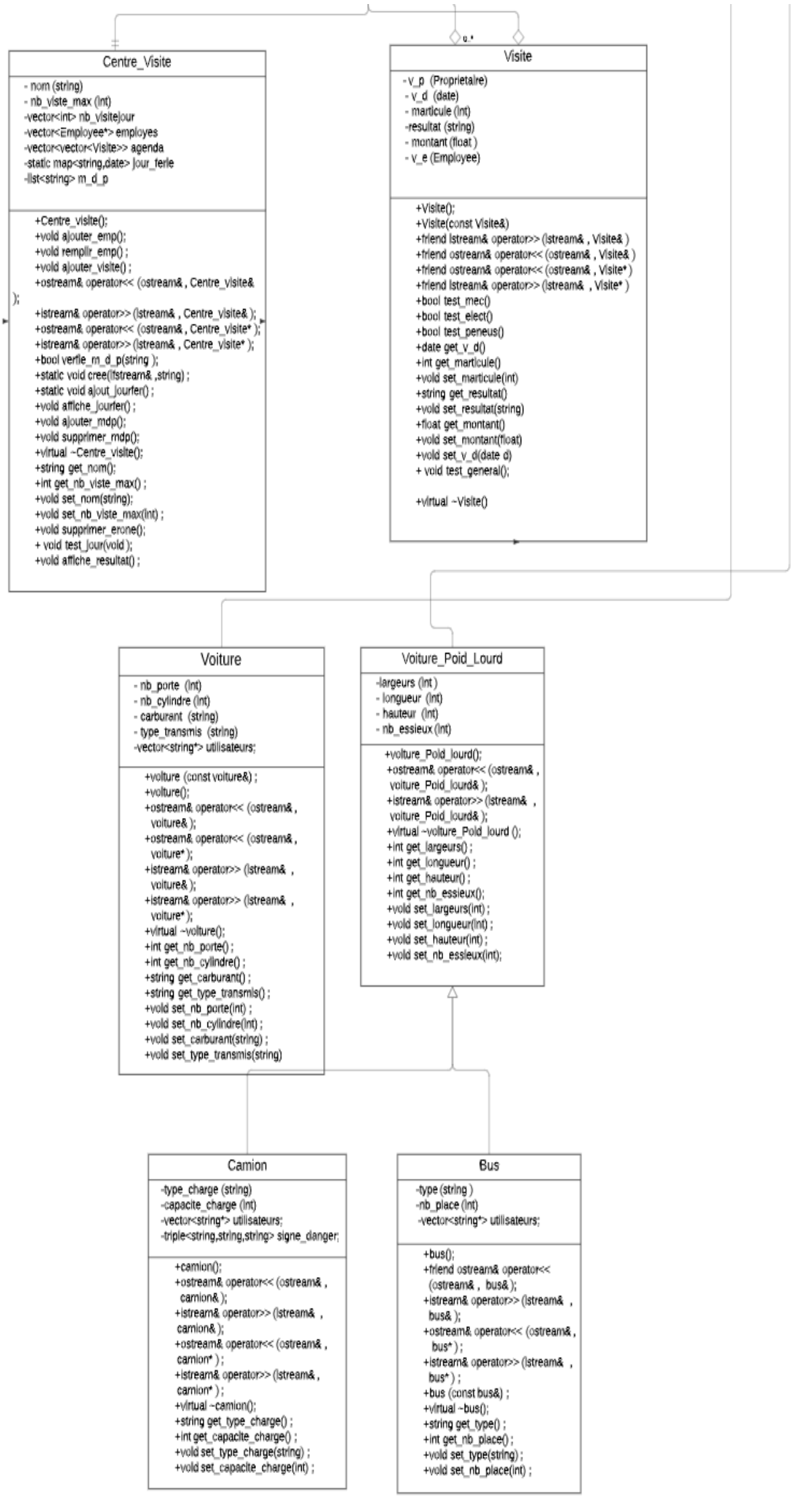
Table des matières

Introduction

Le projet consiste à créer un système de réservation de visites de voitures pour les utilisateurs. Il faudra créer une interface utilisateur pour choisir une date pour la visite et de consulter le résultat du visite qu'il l'a passée ,Les informations sur les voitures,les employées , les propriétaires , les dates et heures disponibles, et les réservations seront stockées dans un fichier texte. Le traitement des réservations, la validation des visites , l'ajout des employees , l'ajout des mots de passe pour l'espace administrateur seront gérés par l'administrateur. Aussi ce dernier est capable de supprimer les visites archivées a partir d'une date donnée.

1) digramme de classe :




Centre_Visite
- nom (string)
- nb_viste_max (int)
-vector<int> nb_visitejour
-vector<Employee*> employes
-vector<vector<Visite>> agenda
-static map<string,date> jour_ferie
-list<string> m_d_p
+Centre_visite();
+void ajouter_emp();
+void remplir_emp() ;
+void ajouter_visite() ;
+ostream& operator<< (ostream& , Centre_visite&
);
+istream& operator>> (istream& , Centre_visite&);
+ostream& operator<< (ostream& , Centre_visite*);
+istream& operator>> (istream& , Centre_visite*);
+bool verfie_m_d_p(string);
+static void cree(ifstream& ,string) ;
+static void ajout_jourfer() ;
+void affiche_jourfer() ;
+void ajouter_mdp();
+void supprimer_mdp();
+virtual ~Centre_visite();
+string get_nom();
+int get_nb_viste_max() ;
+void set_nom(string);
+void set_nb_viste_max(int) ;
+void supprimer_erone();
+ void test_jour(void);
+void affiche_resultat() ;
0..*
Visite
- v_p (Proprietaire)
- v_d (date)
- marticule (int)
-resultat (string)
- montant (float)
- v_e (Employee)
+Visite();
+Visite(const Visite&)
+friend istream& operator>> (istream& , Visite&)
+friend ostream& operator<< (ostream& , Visite&)
+friend ostream& operator<< (ostream& , Visite*)
+friend istream& operator>> (istream& , Visite*)
+bool test_mec()
+bool test_elect()
+bool test_peneus()
+date get_v_d()
+int get_marticule()
+void set_marticule(int)
+string get_resultat()
+void set_resultat(string)
+float get_montant()
+void set_montant(float)
+void set_v_d(date d)
+ void test_general();
+virtual ~Visite()
Voiture
- nb_porte (int)
- nb_cylindre (int)
- carburant (string)
- type_transmis (string)
-vector<string*> utilisateurs;
+voiture (const voiture&) ;
+voiture();
+ostream& operator<< (ostream& ,
voiture&);
+ostream& operator<< (ostream& ,
voiture*);
+istream& operator>> (istream& ,
voiture&);
+istream& operator>> (istream& ,
voiture*);
+virtual ~voiture();
+int get_nb_porte() ;
+int get_nb_cylindre() ;
+string get_carburant() ;
+string get_type_transmis() ;
+void set_nb_porte(int) ;
+void set_nb_cylindre(int) ;
+void set_carburant(string) ;
+void set_type_transmis(string)
Voiture_Poid_Lourd
-largeurs (int)
- longueur (int)
- hauteur (int)
- nb_essieux (int)
+voiture_Poid_lourd();
+ostream& operator<< (ostream& ,
voiture_Poid_lourd&);
+istream& operator>> (istream& ,
voiture_Poid_lourd&);
+virtual ~voiture_Poid_lourd ();
+int get_largeurs() ;
+int get_longueur() ;
+int get_hauteur() ;
+int get_nb_essieux();
+void set_largeurs(int) ;
+void set_longueur(int) ;
+void set_hauteur(int) ;
+void set_nb_essieux(int);
Camion
-type_charge (string)
-capacite_charge (int)
-vector<string*> utilisateurs;
-triple<string,string,string> signe_danger;
+camion();
+ostream& operator<< (ostream& ,
camion&);
+istream& operator>> (istream& ,
camion&);
+ostream& operator<< (ostream& ,
camion*) ;
+istream& operator>> (istream& ,
camion*) ;
+virtual ~camion();
+string get_type_charge() ;
+int get_capacite_charge() ;
+void set_type_charge(string) ;
+void set_capacite_charge(int) ;
Bus
-type (string)
-nb_place (int)
-vector<string*> utilisateurs;
+bus();
+friend ostream& operator<<
(ostream& , bus&);
+istream& operator>> (istream& ,
bus&);
+ostream& operator<< (ostream& ,
bus*) ;
+istream& operator>> (istream& ,
bus*) ;
+bus (const bus&) ;
+virtual ~bus();
+string get_type() ;
+int get_nb_place() ;
+void set_type(string) ;
+void set_nb_place(int) ;

2) Code source :

1) Personne.h :

```
#ifndef PERSONNE_H
#define PERSONNE_H
#include <vector>

#include <iostream>
using namespace std;
#include "string"
#include "date.h"

class Personne
{
protected:
    int id;
    string nom;
    string mail;
    date date_naissance;
    vector<int*> telephonnes;

public :
    Personne(int, string, string);
    Personne();
    Personne(const Personne& );
    friend istream& operator>> (istream&, Personne&);
    friend ostream& operator<< (ostream&, const Personne&);
    virtual void affiche()=0;

    int get_id();
    string get_nom();
    string get_mail();
    date get_date_naissance();

    void set_id(int);
    void set_nom(string);
    void set_mail(string);
    void set_date_naissance(date);

    virtual ~Personne();
};

#endif // PERSONNE_H
```

2) Personne.cpp :

```
#include "Personne.h"

Personne::Personne (int c, string n, string m){
    id = c;
    nom = n;
    mail = m;
    date_naissance.saisir_date("date de naissance");
}

Personne::Personne() {
    date d;
    id = 0;
    nom = "*";
    mail = "*";
    date_naissance = d;
}
Personne::Personne(const Personne& a)
{
      id =a.id;
    nom = a.nom;
    mail = a.mail;
    date_naissance=a.date_naissance;

    int i;
    for (i=0;i< (a.telephonnes).size();i++)
    {   int* d=new int[1];
        *d=*a.telephonnes[i];

        telephonnes.push_back(d);

    }









}

istream& operator>> (istream& in, Personne& p) {
    cout << "donner le id personne " << endl;
    in >> p.id;
    cout << "donner le nom" << endl;
    in >> p.nom;
    cout << "donner le mail " << endl;
    in >> p.mail;
```

```
    cout << "donner la date de naissance " << endl;
    in >> p.date_naissance;
     int* S;
     int choix;

     do
     {
        cout<<"taper 1 pour ajouter un nemero de telphone"<<endl;
        cin>>choix;

        if(choix==1 )
        {   S =new int[1];
            cin>>*S;

            (p.telephonnes).push_back(S);

        }
     }
     while(choix==1);


}

ostream& operator<< (ostream& out, const Personne& p) {
    out << "id est " << p.id<<endl;
    out << "nom est " << p.nom << endl;
    out << "mail est" << p.mail << endl;
    out << "date de naissance est" << p.date_naissance << endl;
    cout<<"le nombre de telphone sont"<<endl;
    for (int i=0; i<p.telephonnes.size();i++)
    {
        out<<*p.telephonnes[i]<<endl;

    }
    return out;
}

void Personne::affiche() {
    cout << "id est " << id << endl;
    cout << "nom est " << nom << endl;
    cout << "maail est " << mail << endl;
    cout << "date de naissance est" << date_naissance << endl;
    cout<<"le nombre de telphone sont"<<endl;
    for (int i=0; i<telephonnes.size();i++)
    {
        cout<<*telephonnes[i]<<"\t";
    }
   cout<<endl;
```

```
}

int Personne::get_id() {
    return id;
}

string Personne::get_nom() {
    return nom;
}
string Personne::get_mail() {
    return mail;
}
date Personne::get_date_naissance() {
    return date_naissance;
}

void Personne::set_id(int a) {
    id = a;
}
void Personne::set_nom(string a) {
    nom = a;
}
void Personne::set_mail(string b) {
    mail = b;
}
void Personne::set_date_naissance(date d) {
    date_naissance = d;
}
Personne::~Personne()
{
    //dtor
}
```

3) Vehicule.h :

```
#include "date.h"
#include "triple.h"

class Vehicule

{
        friend class date;
        protected :
        int Num_chasis ;
        string marque ;
        string model ;
        int matricule ;
        date date_utilis ;

        date date_expiree ;
        vector <date*> historique;
    public:

       //  Vehicule(const Vehicule &);
        Vehicule(int,string,string,int) ;
       Vehicule() ;
       Vehicule(const Vehicule& );
        friend istream& operator>> (istream& , Vehicule& );
        friend ostream& operator<< (ostream& , const Vehicule& );




        int get_Num_chasis() ;
        string get_marque() ;
        string get_model() ;
        int get_matricule() ;
        date get_date_utilis() ;
        date get_date_expiree() ;



        void set_Num_chasis(int) ;
        void set_marque(string ) ;
        void set_model(string) ;
        void set_matricule(int) ;
        void set_date_utilis(date ) ;
        void set_date_expiree(date) ;
```

```
        virtual ~Vehicule();

    protected:

    private:
};

#endif // VEHICULE_H
```

4) Vehicule.cpp :

```
#include "Vehicule.h"

Vehicule::Vehicule(int a,string b,string c,int d) : Num_chasis(a) ,marque(b)
,model(c) ,matricule(d)
{
{
    date_utilis.saisir_date("date_utilis");
    date_expiree.saisir_date("date_expiree");
}



    //ctor
}

ostream& operator<< (ostream& out, const Vehicule& a)
{

    out<<"Num_chasis est "<<a.Num_chasis <<endl;
    out<<"marque est "<<a.marque <<endl;
    out<<"model est "<<a.model <<endl;
    out<<"matricule est "<<a.matricule <<endl;
    out<<"date_utilis est "<<a.date_utilis <<endl;
    out<<"date_expiree est "<<a.date_expiree<<endl;
    int i;
    for (i=0;i< (a.historique).size();i++)
    {
        out<<"le "<<i<<" historique date est "<<*((a.historique)[i])<<endl;

    }

return out;
}

istream& operator>> (istream& in, Vehicule& a)
```

```
{
    cout<<"donnez le Num_chasis"<<endl;
    in>>a.Num_chasis ;
    cout<<"donnez la marque " <<endl;
    in>>a.marque ;
    cout<<"donnez la model "<<endl;
    in>>a.model ;
    cout<<"donnez le matricule " <<endl;
    in>>a.matricule ;
    cout<<"donnez le date_utilis "<<endl;
    in>>a.date_utilis ;
    cout<<"donnez le date_expiree "<<endl;
    in>>a.date_expiree;
    date S;
    int choix;

     do
     {
        cout<<"\n il ya un date  de historique_conduite te veux ajouter ,taper
1 pour ajouter"<<endl;
        cin>>choix;

        if(choix==1 )
        {   date* S ;
            S=new date[1];
            cin>>*S;

            (a.historique).push_back(S);

        }
     }
     while(choix==1);
return in;
}
Vehicule::Vehicule()
{
    date a ;
    this->matricule=0;
    this->date_expiree=a;
    this->date_utilis=a;
    this->model="*";
    this->marque="*";
    this->Num_chasis=0;
}
Vehicule::Vehicule(const Vehicule& a)

{
    Num_chasis=a.Num_chasis ;
    marque=a.marque;
```

```
    model=a.model;
    matricule=a.matricule ;
    date_utilis=a.date_utilis ;
    date_expiree=a.date_expiree ;
    int i;
    for (i=0;i< (a.historique).size();i++)
    {   date* d;
        d=new date(*(a.historique)[i]);

        historique.push_back(d);

    }
}

        int Vehicule::get_Num_chasis()
        {
            return(Num_chasis);
        }
        string Vehicule::get_marque()
        {
            return(marque);
        }
        string Vehicule::get_model()
        {
            return(model);
        }
        int Vehicule::get_matricule()
        {
            return(matricule);
        }
        date Vehicule::get_date_utilis()
        {
            return(date_utilis);
        }
        date Vehicule::get_date_expiree()
        {
            return(date_expiree);
        }
        void Vehicule::set_Num_chasis(int a)
        {
            Num_chasis=a;
        }
        void Vehicule::set_marque(string a)
        {
            marque=a;
        }
        void Vehicule::set_model(string a)
        {
           model=a;
```

```
        }
        void Vehicule::set_matricule(int a)
        {
            matricule=a;
        }
        void Vehicule::set_date_utilis(date a )
        {
            date_utilis=a;
        }
        void Vehicule::set_date_expiree(date a)
        {
          date_expiree=a;
        }

Vehicule::~Vehicule()
{
    //dtor
}
```

5) voiture_Poid_lourd .h:

```
#ifndef VOITURE_POID_LOURD _H
#define VOITURE_POID_LOURD _H

#include <Vehicule.h>


class voiture_Poid_lourd  : public Vehicule
{
    protected :
    int largeurs ;
    int longueur ;
    int hauteur ;
    int nb_essieux;

    public:

        voiture_Poid_lourd (Vehicule,int=0,int=0,int=0,int=0);
        voiture_Poid_lourd();
        friend ostream& operator<< (ostream& ,  voiture_Poid_lourd& );
        friend istream& operator>> (istream&  ,  voiture_Poid_lourd& );
        virtual ~voiture_Poid_lourd ();
        int get_largeurs() ;
        int get_longueur() ;
        int get_hauteur() ;
        int get_nb_essieux();

        void set_largeurs(int) ;
```

```
        void set_longueur(int) ;
        void set_hauteur(int) ;
        void set_nb_essieux(int);

    protected:

    private:
};

#endif // VOITURE_POID_LOURD _H
```

6) voiture_Poid_lourd .cpp :

```
voiture_Poid_lourd ::voiture_Poid_lourd (Vehicule a,int b,int c,int d,int
e):Vehicule(a),largeurs(b),longueur(c),hauteur (d),nb_essieux(e)
{
    //ctor
}
voiture_Poid_lourd::voiture_Poid_lourd()
{
        date a ;
    this->matricule=0;
    this->date_expiree=a;
    this->date_utilis=a;
    this->model="*";
    this->marque="*";
    this->Num_chasis=0;
    this->hauteur=0;
    this->largeurs=0;
    this->longueur=0;
    this->nb_essieux=0;
    this->largeurs=0;
}
ostream& operator<< (ostream& out ,  voiture_Poid_lourd& a)
{
    Vehicule *c=&a;
    out<<*c;
    out<<"largeurs est "<<a.largeurs<<endl ;
    out<<"longueur est "<<a.longueur<<endl ;
    out<<"hauteur est "<<a.hauteur<<endl ;
    out<<"nb_essieux est "<<a.nb_essieux<<endl;
}
istream& operator>> (istream& in ,  voiture_Poid_lourd& a)
{
    Vehicule *c=&a;
    in>>*c;
    cout<<"donnez le largeurs"<<endl;
    in>>a.largeurs ;
```

```
    cout<<"donnez le longueur"<<endl;
    in>>a.longueur ;
    cout<<"donnez le hauteur"<<endl;
    in>>a.hauteur ;
    cout<<"donnez le nb_essieux"<<endl;
    in>>a.nb_essieux;
}
        int voiture_Poid_lourd::get_largeurs()
        {
           return(largeurs);
        }
        int voiture_Poid_lourd::get_longueur()
        {
            return(longueur);
        }
        int voiture_Poid_lourd::get_hauteur()
        {
            return(hauteur);
        }
        int voiture_Poid_lourd::get_nb_essieux()
        {
            return(nb_essieux);
        }
        void voiture_Poid_lourd::set_largeurs(int a)
        {
            largeurs=a;
        }
        void voiture_Poid_lourd::set_longueur(int a)
        {
            longueur=a;
        }
        void voiture_Poid_lourd::set_hauteur(int a)
        {
            hauteur=a;
        }
        void voiture_Poid_lourd::set_nb_essieux(int a)
        {
            nb_essieux=a;
        }
voiture_Poid_lourd ::~voiture_Poid_lourd ()
{
    //dtor
}
```

7) voiture.h :

```
#ifndef VOITURE_H
#define VOITURE_H

#include <Vehicule.h>


class voiture : public Vehicule
{   int nb_porte ;
    int nb_cylindre ;
    string carburant ;
    string type_transmis ;
    vector<string*> utilisateurs;
    public:
        voiture (Vehicule,int=0,int=0,string=" ",string=" ");
        voiture (const voiture&) ;
        voiture();
        friend ostream& operator<< (ostream& ,  voiture& );
        friend ostream& operator<< (ostream& ,  voiture* );
        friend istream& operator>> (istream&  ,  voiture& );
        friend istream& operator>> (istream&  ,  voiture* );
        virtual ~voiture();
        int get_nb_porte() ;
        int get_nb_cylindre() ;
        string get_carburant() ;
        string get_type_transmis() ;

        void set_nb_porte(int) ;
        void set_nb_cylindre(int) ;
        void set_carburant(string) ;
        void set_type_transmis(string) ;

    protected:

    private:
};

#endif // VOITURE_H
```

8) voiture.cpp :

```
#include "voiture.h"


voiture ::voiture (Vehicule a,int b,int c,string d,string
e):Vehicule(a),nb_porte(b),nb_cylindre(c),carburant (d),type_transmis(e)
{
    //ctor
}
voiture::voiture()
{
        date a ;
    this->matricule=0;
    this->date_expiree=a;
    this->date_utilis=a;
    this->model="*";
    this->marque="*";
    this->Num_chasis=0;
    this->nb_porte=0 ;
    this->nb_cylindre=0 ;
    this->carburant="*" ;
    this->type_transmis="*" ;
}
ostream& operator<< (ostream& out,  voiture* a)
{

    out<<" "<<(a->matricule);
    out<<" "<<(&a->date_expiree);
    out<<" "<<(&a->date_utilis);
    out<<" "<<(a->model);
    out<<" "<<(a->marque);
    out<<" "<<(a->Num_chasis);
    out<<" "<<(a->nb_porte) ;
    out<<" "<<(a->nb_cylindre) ;
    out<<" "<<(a->carburant) ;
    out<<" "<<(a->type_transmis) ;
    out<<" "<<(a->historique).size();

    for (int i=0;i< (a->historique).size();i++)
    {
        out<<" "<<((a->historique)[i])<<endl;
    }

    out<<" "<<(a->utilisateurs).size();
    for (int i=0;i< (a->utilisateurs).size();i++)
    {
        out<<" "<<*((a->utilisateurs)[i])<<endl;
    }
```

```
    return(out);

}
ostream& operator<< (ostream& out ,  voiture& a)
{
    Vehicule *c=&a;
    out<<*c;
    out<<"nb_porte "<<a.nb_porte<<endl ;
    out<<"nb_cylindre est "<<a.nb_cylindre<<endl ;
    out<<"carburant est "<<a.carburant<<endl ;
    out<<"type_transmis est "<<a.type_transmis<<endl;
    int i;
    for (i=0;i< (a.utilisateurs).size();i++)
    {
        out<<"le "<<i<<" conducteur est "<<*((a.utilisateurs)[i])<<endl;

    }
}
istream& operator>> (istream& in ,  voiture& a)
{
    Vehicule *c=&a;
    in>>*c;
    cout<<"donnez le nb_porte"<<endl;
    in>>a.nb_porte ;
    cout<<"donnez le nb_cylindre"<<endl;
    in>>a.nb_cylindre ;
    cout<<"donnez le carburant"<<endl;
    in>>a.carburant ;
    cout<<"donnez le type_transmis"<<endl;
    in>>a.type_transmis;
    int choix;
    do
     {
        cout<<"\n il ya un conduteur   te veux ajouter ,taper 1 pour
ajouter"<<endl;

        cin>>choix;

        if(choix==1 )
        {   string* S ;
            S=new string[1];
            cin>>*S;

            (a.utilisateurs).push_back(S);

        }
     }
     while(choix==1);
```

```
    return(in);
}
        int voiture::get_nb_porte()
        {
            return(nb_porte);
        }
        int voiture::get_nb_cylindre()
        {
           return(nb_cylindre);
        }
        string voiture::get_carburant()
        {
            return(carburant);
        }
        string voiture::get_type_transmis()
        {
            return(type_transmis);
        }

        void voiture::set_nb_porte(int a)
        {
            nb_porte=a;
        }
        void voiture::set_nb_cylindre(int a)
        {
            nb_cylindre=a;
        }
        void voiture::set_carburant(string a)
        {
            carburant=a;
        }
        void voiture::set_type_transmis(string a)
        {
            type_transmis=a;
        }
istream& operator>> (istream&  in,  voiture* a)
{

    in>>(a->matricule);
    in>>&(a->date_expiree);
    in>>&(a->date_utilis);
    in>>(a->model);
    in>>(a->marque);
    in>>(a->Num_chasis);
    in>>(a->nb_porte) ;
    in>>(a->nb_cylindre) ;
    in>>(a->carburant) ;
    in>>(a->type_transmis) ;
     int taille;
```

```
    in>>taille;
    for (int i=0;i< taille;i++)

    {
        date* d ;
        d=new date[1];
        in>>d;

        a->historique.push_back(d);
    }
    in>>taille;
    for (int i=0;i< taille;i++)

    {
        string* d ;
        d=new string[1];
        in>>*d;

        a->utilisateurs.push_back(d);
    }




    return(in);
}
voiture::voiture(const voiture& a)
{
    Num_chasis=a.Num_chasis ;
    marque=a.marque;
    model=a.model;
    matricule=a.matricule ;
    date_utilis=a.date_utilis ;
    date_expiree=a.date_expiree ;
    int i;
    for (i=0;i< (a.historique).size();i++)
    {   date* d;
        d=new date(*(a.historique)[i]);

        historique.push_back(d);

    }
    for (i=0;i< (a.utilisateurs).size();i++)
    {   string* d;
        d=new string(*(a.utilisateurs)[i]);

        utilisateurs.push_back(d);

    }
```

```
    nb_porte=a.nb_porte ;
    nb_cylindre=a.nb_cylindre ;
    carburant=a.carburant ;
    type_transmis=a.type_transmis ;
}
voiture ::~voiture ()
{
    //dtor
}
```

9) bus.h :

```
#ifndef BUS_H
#define BUS_H

#include <voiture_Poid_lourd .h>


class bus : public voiture_Poid_lourd
{   string type ;
    int nb_place ;
    vector<string*> utilisateurs;

    public:
        bus (voiture_Poid_lourd,string=" ",int=0);
        bus();
        friend ostream& operator<< (ostream& ,  bus& );
        friend istream& operator>> (istream&  ,  bus& );
        friend ostream& operator<< (ostream& ,  bus* ) ;
        friend istream& operator>> (istream&  ,  bus* ) ;
        //bus (const bus&) ;
        virtual ~bus();
        string get_type() ;
        int get_nb_place() ;
        void set_type(string) ;

        void set_nb_place(int) ;
    protected:

    private:
};

#endif // BUS_H
```

10) bus.cpp :

```
#include "bus.h"

bus::bus(voiture_Poid_lourd a,string b,int
c):voiture_Poid_lourd(a),type(b),nb_place (c)
{
    //ctor
}
bus::bus()
{
        date a ;
    this->matricule=0;
    this->date_expiree=a;
    this->date_utilis=a;
    this->model="*";
    this->marque="*";
    this->Num_chasis=0;
    this->hauteur=0;
    this->longueur=0;
    this->type="*";
    this->nb_place=0;
    this->largeurs=0;

}
istream& operator>> (istream&  in,  bus* a)
{

    in>>(a->matricule);
    in>>&(a->date_expiree);
    in>>&(a->date_utilis);
    in>>(a->model);
    in>>(a->marque);
    in>>(a->Num_chasis);
    in>>(a->hauteur) ;
    in>>(a->largeurs) ;
    in>>(a->longueur) ;
    in >> a->nb_essieux ;
    in>>(a->type) ;
    in>>(a->nb_place) ;
    int taille;
    in>>taille;
    for (int i=0;i< taille;i++)

    {
        date* d ;
        d=new date[1];
        in>>d;
```

```
        a->historique.push_back(d);
    }
    in>>taille;
    for (int i=0;i< taille;i++)

    {
        string* d ;
        d=new string[1];
        in>>*d;

        a->utilisateurs.push_back(d);
    }
    return(in);
}
ostream& operator<< (ostream& out ,  bus& a)
{
    Vehicule *c=&a;
    out<<static_cast<voiture_Poid_lourd&>(*c);
    out<<"type est "<<a.type<<endl ;
    out<<"nb_place est "<<a.nb_place<<endl ;
    int i;
    for (i=0;i< (a.utilisateurs).size();i++)
    {
        out<<"le "<<i<<" conducteur est "<<*((a.utilisateurs)[i])<<endl;

    }
    return(out);

}

istream& operator>> (istream& in ,  bus& a)
{
    Vehicule *c=&a;
    in>>static_cast<voiture_Poid_lourd&>(*c);
    cout<<"donnez le type"<<endl;
    in>>a.type ;
    cout<<"donnez le nb_place "<<endl;
    in>>a.nb_place ;
    int choix;
    do
     {
        cout<<"\n il ya un conduteur   te veux ajouter ,taper 1 pour
ajouter"<<endl;

        cin>>choix;

        if(choix==1 )
        {   string* S ;
```

```
            S=new string[1];
            cin>>*S;

            (a.utilisateurs).push_back(S);

        }
     }
     while(choix==1);
    return(in);

}
         string bus::get_type()
         {
             return(type);
         }
         int bus::get_nb_place()
         {
             return(nb_place);
         }
         void bus::set_type(string a)
         {
             type=a;
         }
         void bus::set_nb_place(int a)
         {
             nb_place=a;
         }
ostream& operator<< (ostream& out,  bus* a)
{

    out<<" "<<(a->matricule);
    out<<" "<<(&a->date_expiree);
    out<<" "<<(&a->date_utilis);
    out<<" "<<(a->model);
    out<<" "<<(a->marque);
    out<<" "<<(a->Num_chasis);
    out<<" "<<(a->hauteur);
    out<<" "<<(a->largeurs) ;
    out<<" "<<(a->longueur) ;
    out <<" "<< a->nb_essieux ;
    out<<" "<<(a->type) ;
    out<<" "<<(a->nb_place) ;
    out<<" "<<(a->historique).size();

    for (int i=0;i< (a->historique).size();i++)
    {
        out<<" "<<((a->historique)[i])<<endl;

    }
```

```
    out<<" "<<(a->utilisateurs).size();
    for (int i=0;i< (a->utilisateurs).size();i++)
    {
        out<<" "<<*((a->utilisateurs)[i])<<endl;
    }


    return(out);
}

bus ::~bus ()
{
    //dtor
}
```

11) camion.h :

```
#ifndef CAMION_H
#define CAMION_H

#include <voiture_Poid_lourd .h>


class camion : public voiture_Poid_lourd
{  string type_charge ;
   int capacite_charge ;
   vector<string*> utilisateurs;
   triple<string,string,string> signe_danger;
    public:
        camion (voiture_Poid_lourd,string=" ",int=0);
        camion();
        friend ostream& operator<< (ostream& ,  camion& );
        friend istream& operator>> (istream&  ,  camion& );
        friend ostream& operator<< (ostream& ,  camion* ) ;
        friend istream& operator>> (istream& ,  camion* ) ;
        virtual ~camion();
         string get_type_charge() ;

         int get_capacite_charge() ;
         void set_type_charge(string) ;
         void set_capacite_charge(int) ;


    protected:

    private:
};

#endif // CAMION_H
```

12) employee.h :

```
#ifndef EMPLOYEE_H
#define EMPLOYEE_H

#pragma once
#include <iostream>
#include "Personne.h"
#include"date.h"
#include <map>
#include "couple.h"
using namespace std;
#include "string"
class Employee : public Personne
{
protected:
    float heure_travaille;
    int salaire;
    date date_embauche;
    string fonction;
    map<int,couple<date,int>> vacance_emp;



public:
    Employee(int ,string,string, float, int, string);
    Employee();
    virtual void affiche();
    friend istream& operator>> (istream&, Employee&);
    friend ostream& operator<< (ostream&, Employee&);
    friend ostream& operator<< (ostream& , Employee* );
    friend  istream& operator>> (istream& , Employee* ) ;

    float get_heure_travaille();
    int get_salaire();
    date get_date_embauche();
    string get_fonction();
    void set_heure_travaille(float);
    void set_salaire(int);
    void set_date_embauche(date);
    void set_fonction(string);

    virtual ~Employee();
};

#endif // EMPLOYEE_H
```

13) employee.cpp :

```
#include "Employee.h"



Employee:: Employee(int c, string n, string m, float h, int s, string f)
{   id=c;
    nom=n;
    mail=m;

    heure_travaille = h;
    salaire = s;
    date_embauche.saisir_date("date de l'embauche");
    fonction = f;
}

Employee::Employee() {
    date d;
    id = 0;
    nom = "*";
    mail = "*";
    date_naissance = d;
    heure_travaille = 0;
    salaire = 0;
    date_embauche = d;
    fonction = "*";
}

istream& operator>> (istream& in, Employee& emp) {
    Personne* p;
    p = &emp;
    in >> *p;
    cout << "donner les heures de travailles" << endl;
    in >> emp.heure_travaille;
    cout << "donner le saliare" << endl;
    in >> emp.salaire;
    cout << "donner la date de l'embauche" << endl;
    in >> emp.date_embauche;
    cout << "donner la fonction" << endl;
    in >> emp.fonction;
    int choix;
    do
     {
        cout<<"\n taper 1 pour ajouter pour ajouter vacance"<<endl;

        cin>>choix;

        if(choix==1 )
```

```
        {   date S ;

            cin>>S;
   int z;
   do
   {
       in>>z;
       if (z<1)
        cout<<"les nombres de jour de vacance doit  superiur 0 "<<endl;
   }
   while ((z<1));
   int id;
   do
   {
       in>>id;
       if (id<1)
        cout<<"les id de vacance doit  superiur 0 "<<endl;
   }
   while ((id<1));

          couple<date,int> c(S,z);

           emp.vacance_emp[id]=c;

        }
     }
     while(choix==1);
    return in;
}
 ostream& operator<< (ostream& out, Employee* w)
 {
    out<<" "<<w->id<<endl;
    out<<" "<<w->nom<<endl;
    out<<" "<<w->mail<<endl;
    out<<" "<<&(w->date_naissance)<<endl;
    out<<" "<<(w->telephonnes).size();

    for (int i=0;i< (w->telephonnes).size();i++)
    {
        out<<" "<<*((w->telephonnes)[i])<<endl;
    }
    out<<" "<<w->heure_travaille<<endl;
    out<<" "<<w->salaire<<endl;
    out<<" "<<&(w->date_embauche)<<endl;
    out<<" "<<w->fonction<<endl;
    out<<" "<<(w->vacance_emp).size();
    map<int,couple<date,int>> ::iterator it ;
    for (it = w->vacance_emp.begin(); it != w->vacance_emp.end(); ++it)
    {
```

```
        out<<" "<<it->first<<endl;
        couple<date,int> c =it->second;
        date x1=c.get_x1();
        out<<" "<<&x1<<endl;
        int x2=c.get_x2();
        out<<" "<<x2<<endl;



    }
    return(out);



}
void Employee::affiche() {
    cout << "id est " << id << endl;
    cout << "nom est " << nom << endl;
    cout << "maail est " << mail << endl;
    cout << "date de naissance est" << date_naissance << endl;
    cout<<"le nombre de telphone sont"<<endl;
    for (int i=0; i<telephonnes.size();i++)
    {
        cout<<*telephonnes[i]<<"\t";
    }
   cout<<endl;
   Employee emp = *this;

    cout << "les heures de travaille sont" << emp.heure_travaille << endl;
    cout << "le salaire est" << emp.salaire << endl;
    cout << "la date d'embauche est" << emp.date_embauche << endl;
    cout << "la fonction est" << emp.fonction << endl;
    map<int,couple<date,int>> ::iterator it ;
    for (it = emp.vacance_emp.begin(); it != emp.vacance_emp.end(); ++it)
    {
       cout  << "l'identifier de vacance "<< it->first<<endl;
       cout  << "la date et nombre de jour"<< it->second << endl;
    }


}
istream& operator>> (istream& in, Employee* w)
{
    in>>w->id;
    in>>w->nom;
    in>>w->mail;
    in>>&(w->date_naissance);
    int c;
    in>>c;
```

```
    for (int i=0;i< c;i++)

    {
        int* d ;
        int k;
        in>>k;

        d=new int(k);

        w->telephonnes.push_back(d);
    }
    in>>w->heure_travaille;
    in>>w->salaire;
    in>>&(w->date_embauche);
    in>>w->fonction;

    in>>c;
    for (int i=0;i< c;i++)

    {
        int cle,jour;
        date d;
        in>>cle;
        in>>&d;
        in>>jour;
        couple<date,int> f(d,jour);
        w->vacance_emp[cle]=f;


    }
    return(in);



 }
ostream& operator<< (ostream& out, Employee& emp) {
    Personne* p;
    p = &emp;
    out << *p;
    out << "les heures de travaille sont" << emp.heure_travaille << endl;
    out << "le salaire est" << emp.salaire << endl;
    out << "la date d'embauche est" << emp.date_embauche << endl;
    out << "la fonction est" << emp.fonction << endl;
    map<int,couple<date,int>> ::iterator it ;
    for (it = emp.vacance_emp.begin(); it != emp.vacance_emp.end(); ++it)
    {
       cout  << "l'identifier de vacance "<< it->first<<endl;
       cout  << "la date et nombre de jour"<< it->second << endl;    }
```

```
    return out;
}
float Employee::get_heure_travaille()
{
    return heure_travaille;
}
int Employee::get_salaire() {
    return salaire;
}
date Employee::get_date_embauche() {
    return date_embauche;
}
string Employee::get_fonction() {
    return fonction;
}
void Employee::set_heure_travaille(float a) {
    id = a;
}
void Employee::set_salaire(int a) {
    salaire = a;
}
void Employee::set_date_embauche(date d) {
    date_embauche = d;
}
void Employee::set_fonction(string f) {
    fonction = f;
}

Employee::~Employee()
{
    //dtor
}
```

14) Proprietaire.h :

```
#ifndef PROPRIETAIRE_H
#define PROPRIETAIRE_H
#include "Personne.h"
#include "bus.h"
#include "voiture.h"
#include "camion.h"
#include <fstream>
using namespace std;
#include <vector>
class Proprietaire :public Personne
{
    vector <Vehicule*> vehicules;
```

```
    int num_permis;


public:

    Proprietaire();
    Proprietaire(const Proprietaire&);
    void remplir();
    friend ostream& operator<< (ostream& , Proprietaire& );
    friend istream& operator>> (istream& , Proprietaire& );
    Vehicule* get_vehicule(int);
    static void cree(ifstream& ) ;
    friend ostream& operator<< (ostream& , Proprietaire* );
    friend istream& operator>> (istream& , Proprietaire* ) ;
    virtual void affiche();
    int recharhe(int ) ;

    int get_num_permis();
    void set_num_permis(int);
    void ajouter_vehicule(Vehicule);
    int get_size();
    virtual ~Proprietaire();
};

#endif // PROPRIETAIRE_H
```

15) Propriétaire.cpp :

```
#include "Proprietaire.h"

Proprietaire::Proprietaire()
{
    date d;
    id = 0;
    nom = "*";
    mail = "*";
    date_naissance = d;
    num_permis=0;

}
void Proprietaire::remplir()
   { char rep;
     Vehicule *q;
     int choix;
    do
    {
        cout<<"\n taper 1: voiture, 2:camion, 3:bus "<<endl;
        cin>>choix;
```

```
        if(choix==1)
            {
                 q=new voiture();
                 cin>>static_cast<voiture&> (*q);
            }
        else if(choix==2)
        {
              q=new camion();
              cin>>static_cast<camion&> (*q);
        }


        else if(choix ==3)
        {
            q=new bus();
            cin>>static_cast<bus&> (*q);
        }
        else break;

        vehicules.push_back(q);
        cout<<"\n rajouter ?"<<endl;
        cin>>rep;
    }
    while(rep=='o' || rep=='0');

   }

void Proprietaire::affiche() {
    cout << "id est " << id << endl;
    cout << "nom est " << nom << endl;
    cout << "maail est " << mail << endl;
    cout << "date de naissance est" << date_naissance << endl;
    cout<<"le nombre de telphone sont"<<endl;
    for (int i=0; i<telephonnes.size();i++)
    {
        cout<<*telephonnes[i]<<"\t";
    }
   cout<<endl;
  Proprietaire a=*this;
       cout << "num_permis est " << a.num_permis<<endl;
    int i;

    for (i=0;i< (a.vehicules).size();i++)
    {


        if(typeid(*((a.vehicules)[i]))==typeid(voiture))
            {
                 cout<<"le "<<i<<" vehicule est ";
```

```
                cout<<static_cast<voiture&> (*((a.vehicules)[i]));
                cout<<((a.vehicules)[i]);
                cout<<endl;
            }
        else if (typeid(*((a.vehicules)[i]))==typeid(camion))
            {     cout<<"le "<<i<<" vehicule est ";

                cout<<static_cast<camion&> (*((a.vehicules)[i]));
                cout<<endl;
            }

        else if (typeid(*((a.vehicules)[i]))==typeid(bus))
            {
                cout<<"le "<<i<<" vehicule est ";

                cout<<static_cast<bus&> (*((a.vehicules)[i]));
                cout<<endl;
            }
    }


}
istream& operator>> (istream& in, Proprietaire& a)
{   Personne *c=&a;
    in>>*c;
    cout << "donnez num_permis svp " <<endl;
    in>> a.num_permis;
    a.remplir();


}
ostream& operator<< (ostream& out, Proprietaire& a)
{
    Personne *c=&a;
    out<<*c;
    out << "num_permis est " << a.num_permis<<endl;
    int i;

    for (i=0;i< (a.vehicules).size();i++)
    {


        if(typeid(*((a.vehicules)[i]))==typeid(voiture))
            {
                out<<"le "<<i<<" vehicule est ";
                out<<static_cast<voiture&> (*((a.vehicules)[i]));
                //out<<((a.vehicules)[i]);
                out<<endl;
            }
```

```
        else if (typeid(*((a.vehicules)[i]))==typeid(camion))
            {     out<<"le "<<i<<" vehicule est ";

                  out<<static_cast<camion&> (*((a.vehicules)[i]));
                  out<<endl;
            }

        else if (typeid(*((a.vehicules)[i]))==typeid(bus))
            {
                  out<<"le "<<i<<" vehicule est ";

                  out<<static_cast<bus&> (*((a.vehicules)[i]));
                  out<<endl;
            }
    }


}
Proprietaire::Proprietaire( const Proprietaire& a)
{
    id =a.id;
    nom = a.nom;
    mail = a.mail;
    date_naissance=a.date_naissance;
    num_permis=a.num_permis;
    int i;


    for (i=0;i< (a.telephonnes).size();i++)
    {   int* d=new int[1];
        *d=*a.telephonnes[i];

        telephonnes.push_back(d);

    }

    Vehicule* v;
    for (i=0;i< (a.vehicules).size();i++)
    {


        if(typeid(*((a.vehicules)[i]))==typeid(voiture))
            {
                  v =new voiture(static_cast<const
voiture&>(*((a.vehicules)[i])));
                  vehicules.push_back(v);
                  cout<<((a.vehicules)[i]);
```

```
            }
        else if (typeid(*((a.vehicules)[i]))==typeid(camion))
            {
                v =new camion(static_cast<const
camion&>(*((a.vehicules)[i])));
                vehicules.push_back(v);
            }

        else  if (typeid(*((a.vehicules)[i]))==typeid(bus))
            {
                v =new bus(static_cast<const bus&>(*((a.vehicules)[i])));
                vehicules.push_back(v);
            }
    }



}

void Proprietaire::cree(ifstream& f)
{
    f.open("D:\fff.txt",ios::out|ios::in|ios::trunc);
    if(!f.is_open())
        exit(-1);
}
 ostream& operator<< (ostream& out, Proprietaire* w)
 {
    out<<" "<<w->id<<endl;
    out<<" "<<w->nom<<endl;
    out<<" "<<w->mail<<endl;
    out<<" "<<&(w->date_naissance)<<endl;
     out<<" "<<(w->telephonnes).size();

    for (int i=0;i< (w->telephonnes).size();i++)
    {
        out<<" "<<*((w->telephonnes)[i])<<endl;
    }

    out<<" "<<w->num_permis<<endl;




out<<" "<<(w->vehicules).size();
for(int i=0; i<w->vehicules.size(); i++)
{if (typeid(*w->vehicules[i])==typeid(voiture))
```

```
out<<" "<<"voiture "<<&(static_cast< voiture&>(*w->vehicules[i]))<<endl;
else if (typeid(*w->vehicules[i])==typeid(bus))
out<<" "<<"bus "<< &(static_cast< bus&>(*w->vehicules[i]))<<endl;
else if(typeid(*w->vehicules[i])==typeid(camion))
 out<<" "<<"camion "<< &(static_cast< camion&>(*w->vehicules[i]))<<endl;

 }



 return out;
 }
 istream& operator>> (istream& in, Proprietaire* w)
 {

    in>>w->id;
    in>>w->nom;
    in>>w->mail;
    in>>&(w->date_naissance);
    int c;
in>>c;
for (int i=0;i< c;i++)

    {
        int* d ;
        int k;
        in>>k;

        d=new int(k);

        w->telephonnes.push_back(d);
    }

    in>>w->num_permis;
string choix,b;



in>>b;
int x = std::stoi(b);

if(b!="0")
{


 int i=0;
```

```
 do


{in>>choix;


if ((choix)=="voiture")

  {
      voiture* v;
       v =new voiture;
        in>>((v));
         (w->vehicules).push_back(v);

  }

else if (choix=="bus")
{
      bus* v;
       v =new bus;
        in>>(v);
         (w->vehicules).push_back(v);

}


else if(choix=="camion")


 {
      camion* v;
      v =new camion;
      in>>(v);
      (w->vehicules).push_back(v);
}
i+=1;


}
while(i<x);

}



 }

int Proprietaire::get_size() {

        return(vehicules.size() ) ;
```

```
}


Vehicule* Proprietaire::get_vehicule(int i) {

        return(vehicules[i] ) ;
}


int Proprietaire::get_num_permis(){
    return num_permis;
}
void Proprietaire::set_num_permis(int a) {
    num_permis = a;
}
int Proprietaire::recharhe(int c)
{   int i;
    for (i=0;i< (vehicules).size();i++)
    {
        if (c== (vehicules[i])->get_matricule())
        return(i);
    }
    return(-1);

}
Proprietaire::~Proprietaire()
{
     vehicules.clear();

}
```

16) Visite.h :

```
#ifndef VISITE_H
#define VISITE_H
#include "Employee.h"
#include "Proprietaire.h"

using namespace std;

class Visite
{   Proprietaire v_p ;
    date v_d ;
    int marticule ;
    string resultat ;
    float montant  ;
    Employee v_e  ;
    public:
        Visite();
        Visite(const Visite&);
        friend istream& operator>> (istream& , Visite& );
        friend ostream& operator<< (ostream& , Visite& ) ;
        friend ostream& operator<< (ostream& , Visite* ) ;
        friend istream& operator>> (istream&  , Visite* ) ;
        static bool test_mec();
        static bool test_elect();
        static bool test_peneus();
        void test_general();

        date get_v_d() ;
        int get_marticule() ;
        void set_marticule(int) ;
        string get_resultat() ;
        void set_resultat(string) ;
        float get_montant()  ;
        void set_montant(float)  ;
        void set_v_d(date d) ;
        virtual ~Visite();

    protected:

    private:
};

#endif // VISITE_H
```

17) Visite.cpp ;

```
#include "Visite.h"

Visite::Visite()
{
    marticule=0 ;
    resultat="en_attend" ;
    montant=0  ;
    date d;
    v_d=d;
    Proprietaire  p;
    v_p=p;
    Employee e ;
    v_e =e;
}
istream& operator>> (istream& in, Visite& v)
{

     cout << "donner visite date " << endl;
    in>>v.v_d;

    cout << "donner montant " << endl;
    in>>v.montant;

    cout << "donner le donnes de Proprietaire " << endl;
    in>>v.v_p;
    cout << "donner le donnes de  Employee" << endl;
    in>>v.v_e;
    /*do
    {

        int i;
        for ( i=0;i<(v.v_p).get_size();i++)

            {
                if ((((v.v_p).get_vehicule(i))->get_matricule())==v.marticule)


            }

             if ((((v.v_p).get_vehicule(i-1))->get_matricule())==v.marticule)
                    break;

    }
    while(1);*/
    cout << "donner marticule " << endl;
    in>>v.marticule;
    return(in);
```

```
}
ostream& operator<< (ostream& out, Visite& v)
{   cout<<"***************les données de la visite*****************"<<endl;
    out << "le  marticule est" <<v.marticule<< endl;
    out << "le  montant est" <<v.montant<< endl;
    out << "le  date visite  est" <<v.v_d<< endl;


    out << "le donnes de Employee est :" <<v.v_e<< endl;
    out << " le donnes de Proprietaire est :" ;
    out<<static_cast<Proprietaire&> (v.v_p);
    return(out);




}
        bool Visite::test_mec()
        {
            cout<<"taper oui si test mecanique est valide non si non"<<endl;
            {
              string a;
              do

              {
               cin>>a;
               if (a=="oui" )
                return(true);
              }
              while(a!="non");
            }
            return(false);
        }
        bool Visite::test_elect()
        {
            cout<<"taper oui si test elecrique est valide non si non"<<endl;
            {
              string a;
              do

              {
               cin>>a;
               if (a=="oui" )
                return(true);
              }
              while(a!="non");
            }
```

```
        return(false);
    }
    bool Visite::test_peneus()
    {
        cout<<"taper oui si test peneus est valide non si non"<<endl;
        {
          string a;
          do

          {
           cin>>a;
           if (a=="oui" )
            return(true);
          }
          while(a!="non");
        }
        return(false);
    }
    void Visite::test_general()
    {
        bool a,b,c;
        a=this->test_mec();
        b=this->test_elect();
        c=this->test_peneus();
        if (a&&b&&c)
            {
                resultat="valide";
            int i;
            for ( i=0;i<(v_p).get_size();i++)

            {
               if ((((v_p).get_vehicule(i))->get_matricule())==marticule)

                break;
            }
            date h;
            h= ((v_p).get_vehicule(i))->get_date_expiree();
            cout<<"donnez valide de visite"<<endl;
            date j;
            cin>>j;
            h=h+j;
            ((v_p).get_vehicule(i))->set_date_expiree(h);


            }
        else
            {
                  resultat="refuse";
                  if (a==false)
```

```
                    resultat+="_test_mecanique";
                    if (b==false)
                    resultat+="_test_electrique";
                    if (c==false)
                    resultat+="_test_peneus";

                }

        }

date Visite::get_v_d()
{
    return(v_d);
}
void Visite::set_v_d(date d)
{
    v_d=d;
}
Visite::Visite( const Visite& a)
{
     Proprietaire c(a.v_p) ;
     v_p=c;
     v_d=a.v_d ;

     marticule=a.marticule ;
     resultat=a.resultat ;
     montant=a.montant  ;
    v_e=a.v_e  ;
}
        int Visite::get_marticule()
        {
            return(marticule);
        }
        void Visite::set_marticule(int a)
        {
            marticule=a;

        }
        string Visite::get_resultat()
        {
            return(resultat);
        }
        void Visite::set_resultat(string a)
        {
            resultat=a;
        }
        float Visite::get_montant()
        {
            return(montant);
```

```
        }
        void Visite::set_montant(float a)
        {
            montant=a;
        }
 ostream& operator<< (ostream& out , Visite* w)
 {
     out<<" "<<&(w->v_p)<<endl;
     out<<" "<<&(w->v_d)<<endl;
     out<<" "<<(w->marticule)<<endl;
     out<<" "<<(w->resultat)<<endl;
     out<<" "<<(w->montant)<<endl;
     out<<" "<<&(w->v_e)<<endl;



     return(out);



 }
  istream& operator>> (istream& in , Visite* w)
 {
     in>>&(w->v_p);
     in>>&(w->v_d);
     in>>(w->marticule);
     in>>(w->resultat);
     in>>(w->montant);
     in>>&(w->v_e);
     return(in);



 }

Visite::~Visite()
{
    //dtor
}
```

18) Centre_visite.h :

```
#ifndef CENTRE_VISITE_H
#define CENTRE_VISITE_H
#include "Visite.h"
#include <fstream>
#include <list>
#include <algorithm>
using namespace std;
class Centre_visite
{   string nom;
    int nb_viste_max ;
    vector<Employee*> employes;
    vector<vector<Visite>> agenda;
    static map<string,date> jour_ferie;
    list<string> m_d_p;
    public:

        Centre_visite();
        void ajouter_emp();
        void remplir_emp() ;
        void ajouter_visite() ;
        friend ostream& operator<< (ostream& , Centre_visite& );
        friend istream& operator>> (istream& , Centre_visite& );
        friend ostream& operator<< (ostream& , Centre_visite* );
        friend istream& operator>> (istream& , Centre_visite* );
        bool verfie_m_d_p(string );
        static void cree(ifstream& ,string) ;
        static void ajout_jourfer() ;
        void affiche_jourfer() ;
        void test_jour(void );
        void affiche_resultat() ;
        void ajouter_mdp();
        void supprimer_mdp();
        virtual ~Centre_visite();
        string get_nom();
        int get_nb_viste_max() ;
        void set_nom(string);
        void set_nb_viste_max(int) ;

        void supprimer_erone();

    protected:

    private:
};

#endif // CENTRE_VISITE_H
```

19) Centre_visite.cpp :

```
#include "Centre_visite.h"


Centre_visite::Centre_visite()
{
    nom="*";
    nb_viste_max=2;
    m_d_p.push_back("0000");

}
void Centre_visite::ajouter_emp()
{ Employee* q;
      q=new Employee();
   cin>>static_cast<Employee&> (*q);

  employes.push_back(q);
}
void Centre_visite::remplir_emp()
{   string rep;
    do
    {

      this->ajouter_emp();
      cout<<"voulez vous ajoutes employee si oui taper o sinon taper n"<<endl;
      cin>>rep;
      if (rep=="n")
      {
          break;
      }
    }
    while(rep=="o");
}
void Centre_visite::ajouter_visite()
{
    Visite v;
    cin>>v;

    date d;
   d=v.get_v_d();
   int i;
   i=0 ;
   do
   {if (agenda.size()==0)
        {


          vector<Visite> c;
```

```
          c.push_back(v);
          agenda.push_back(c);




          break ;
          }

     else if (d.egale(((agenda[i])[0]).get_v_d()))
               {
                  if ((agenda[i]).size()<nb_viste_max)
                   {









                      //Visite a(v);

                      (agenda[i]).push_back(v);


                       break;
                   }
                   else
                     {
                          cout<<"date pas valide parce que la date que vous avez
choisie est pleine"<<endl;
                          cin>>d;

                           ((agenda[i])[(agenda[i]).size()-1]).set_v_d(d);

                          i=-1;

                     }
               }
     i++;
     if (agenda.size()==i)
     {


          vector<Visite> c;

          c.push_back(v);

          agenda.push_back(c);
```

```
        break;
     }



   }
   while(true);



}
ostream& operator<< (ostream& out, Centre_visite& c)
{
    out << "le  nom est" <<c.nom<< endl;
    out << "le  nobre de visite  maximal par jour est" <<c.nb_viste_max<<
endl;
    int i,j;
    for (i=0;i< (c.agenda).size();i++)
    {
        for(j=0;j< (c.agenda[i]).size();j++)
            out<<((c.agenda[i])[j]);
    }
    for (i=0;i< (c.employes).size();i++)
    {

            out<<(*(c.employes[i]));
    }
   list<string>::iterator it ;
   cout<<" "<<(c.m_d_p).size();
   for (it = c.m_d_p.begin(); it != c.m_d_p.end(); it++)
    {
       cout <<" " << *it<<endl;
    }
    return(out);
}
ostream& operator<< (ostream& out, Centre_visite* c)
{
    out << " " <<c->nom<< endl;
    out << " " <<c->nb_viste_max<< endl;
    int i,j;
    out<<" "<<(c->agenda).size();
    for (i=0;i< (c->agenda).size();i++)
    {   out<<" "<<((c->agenda)[i]).size();
        for(j=0;j< (c->agenda[i]).size();j++)

            out<<&((c->agenda[i])[j]);
    }
    out<<" "<<(c->employes).size();
```

```
    for (i=0;i< (c->employes).size();i++)
    {

            out<<((c->employes[i]));
    }
  list<string>::iterator it ;
   out<<" "<<(c->m_d_p).size();
   for (it = c->m_d_p.begin(); it != c->m_d_p.end(); ++it)
    {
       out <<" " << *it<<endl;
    }

    return(out);
}
istream& operator>> (istream& in, Centre_visite* c)
{   in.seekg(0);
    in>>c->nom;
    in>>c->nb_viste_max;
    int i,j;
    int k;
    in>>k;
    for (i=0;i< k;i++)
    {   int d;
        in>>d;
        vector<Visite> a;
        for(j=0;j< d;j++)


         {
              Visite V;
           in>>&V;
           a.push_back(V);
         }
         c->agenda.push_back(a);
    }
    in>>k;
    for (i=0;i< k;i++)
    {
          Employee *E ;
            E=new Employee[1];
            in>>E;
            (c->employes).push_back(E);
    }
    in>>k;
    (c->m_d_p).clear();
    for (i=0;i< k;i++)
    {
          string E ;
```

```
            in>>E;
            (c->m_d_p).push_back(E);
    }
    return(in);
}
istream& operator>> (istream& in, Centre_visite& c)
{
   cout << "donnez svp le nom de centre " <<endl;
   in>>c.nom;
   cout << "donnez nbre maximal de visite par jour " <<endl;
   in>>c.nb_viste_max;
   cout << "donnez le data de emploie " <<endl;
   c.remplir_emp() ;

   return(in);

}
string Centre_visite::get_nom()
{
    return(nom);
}
int Centre_visite::get_nb_viste_max()
{
    return(nb_viste_max);
}
void Centre_visite::set_nom(string a )
{
   nom=a;
}

void  Centre_visite::set_nb_viste_max(int a)
{
  nb_viste_max=a;
}
void Centre_visite::cree(ifstream& f,string a)
{
    f.open(a,ios::out|ios::in);
    if(!f.is_open())
        exit(-1);
}

map<string,date> Centre_visite::jour_ferie;
void Centre_visite::ajout_jourfer()
{   date a;
    cout<<"donnez le nom de jour ferie"<<endl;
    string m;
    cin>>m;
    cout<<"donnez le date de cette jour ferie"<<endl;
    cin>>a;
```

```
    jour_ferie[m]=a;
}
void Centre_visite::affiche_jourfer()
{   map<string,date> ::iterator it ;
    for (it = jour_ferie.begin(); it != jour_ferie.end(); ++it)
    {
       cout  << "nom de vacance "<< it->first  << " la date vacance"<< it-
>second << endl;
    }

}

void Centre_visite::ajouter_mdp()
{ string a;
cin>>a;
list<string>::iterator it ;
it=find(m_d_p.begin(),m_d_p.end(),a);
if(it==m_d_p.end())
    m_d_p.push_back(a);
else
    cout<<"ce mot de passe existe deja"<<endl;
}

bool Centre_visite::verfie_m_d_p(string a)
{
list<string>::iterator it ;
it=find(m_d_p.begin(),m_d_p.end(),a);
if(it==m_d_p.end())
    return(false);
else
    return(true);
}




void Centre_visite::supprimer_mdp()
{ string a;
cin>>a;
if(a.size()!=1)
    m_d_p.remove(a);

}

void Centre_visite:: supprimer_erone()
{
    cout <<"donner la date "<<endl;
    date d;
```

```
    cin>> d;
     for (int i = agenda.size() - 1; i >= 0; i--) {
        date a = (agenda[i][0]).get_v_d();
        if (d == a) {
            agenda.erase(agenda.begin() + i);
        }
     }


}
void Centre_visite::test_jour()
{cout<<"donnnez le date"<<endl;
date d;
cin>>d;
 for (int i=0; i<agenda.size();i++)
   { date a;
     a=(((agenda[i])[0]).get_v_d());
     if (d == a)
     {  cout<<"donnez de marticule de la voiture"<<endl;
        int l;
        cin>>l;
        for (int j=0;i<(agenda[i]).size();j++)
        {  int p;
            p=((agenda[i])[j]).get_marticule();
            if (l==p)

                {(agenda[i])[j].test_general();
                break;}
        }


        break;


     }
   }

}
void Centre_visite::affiche_resultat()
{cout<<"donnnez le date"<<endl;
date d;
cin>>d;
 for (int i=0; i<agenda.size();i++)
   { date a;
     a=(((agenda[i])[0]).get_v_d());
     if (d == a)
     {  cout<<"donnez de marticule de la voiture"<<endl;
        int l;
        cin>>l;
```

```
        for (int j=0;i<(agenda[i]).size();j++)
        {  int p;
            p=((agenda[i])[j]).get_marticule();
            if (l==p)

                {string f;
                f=(agenda[i])[j].get_resultat();
                cout<<f;
                break;}
        }


        break;


     }
   }

}
Centre_visite::~Centre_visite()
{
    //dtor
}
```

20) Date.h :

```
#ifndef DATE_H
#define DATE_H

#include <iostream>
using namespace std ;
#include <string>
class date
{
    int jour;
    int mois ;
    int annee;
    public:
        date(int=0,int=0,int=0);
        void affiche();
        void saisir_date(string ="donnez date");
        friend ostream& operator<< (ostream& , const date& );
        friend ostream& operator<< (ostream& , const date* );
        friend istream& operator>> (istream& , date& );
        friend istream& operator>> (istream& , date* );
        date operator + (date& ) ;
        bool operator == (const date& ) ;
        bool operator < (date& ) ;
        bool operator > (date& ) ;
        virtual ~date();
        int get_jour();
        int get_mois() ;
        int get_annee();
        void  set_jour(int);
        void  set_mois(int) ;
        void  set_annee(int);
        bool  egale(date);


    protected:

    private:
};

#endif // DATE_H
```

21) Date.cpp :

```
#include "date.h"

date::date(int a,int b,int c):jour(a),mois(b),annee(c)
{
    //ctor
}
 void date::saisir_date(string msg)
 {
     cout<<msg<<endl;
     cin>>jour;
     cin>>mois;
     cin>>annee;
 }
 void date::affiche()
 {
     cout<<jour<<":"<<mois<<":"<<annee<<endl;
 }
ostream& operator<< (ostream& out, const date& d)
{
out<<d.annee<<":"<<d.mois<<":"<<d.jour<<endl;


return out;
}
ostream& operator<< (ostream& out, const date* a)
{
    out<<" "<<(a->jour);
    out<<" "<<(a->mois);
    out<<" "<<(a->annee);
    return(out);

}
date date::operator + (date& a )
{date y;
  ;

y.jour=jour;
y.mois=(a.mois+mois)%12;
y.annee=annee+(a.mois+mois)/12+a.annee;
return (y);

}

istream& operator>> (istream& in, date& a)
{

 cout<<"donnez le date sous form A||M||J"<<endl;
```

```
      int tab[]={0,31,28,31,30,31,30,31,31,30,31,30,31}  ;
    in>>a.annee;
    do
   {try
           {
               in>>a.mois;
               if ((a.mois>12) || (a.mois<1) ) throw (-1);

           }

      catch (int e)
       {
         cout<<"le mois doit etre infrieure ou egale de 12 et superiur 0 \n";
       }
    }
    while ((a.mois>12) || (a.mois<1) );
    if (((a.annee)%4==0 && (a.annee)%100!=0 || (a.annee)%400==0))
           *(tab+2)=*(tab+2)+1;

   do
   {
       in>>a.jour;
       if (((a.jour)>tab[a.mois])||(a.jour<1))
        cout<<"le jour doit inferiur "<<tab[a.mois]<<" et superiur 0 "<<endl;
   }
   while (((a.jour)>tab[a.mois])||(a.jour<1));


return in;
}
        int date::get_jour()
        {
          return(jour);
        }
        int date::get_mois()
        {
            return(mois);
        }
        int date::get_annee()
        {
            return(annee);
        }
        void  date::set_jour(int a )
        {
            jour=a;
        }
        void  date::set_mois(int a)
        {
            mois=a;
```

```
        }
        void  date::set_annee(int a)
        {
            annee=a;
        }
        bool  date::egale(date d)
        {
            if((jour==d.jour)&&(mois==d.mois)&&(annee==d.annee))
            return(true);
            return(false);

        }
         istream& operator>> (istream& in, date* a)
        {
             in>>(a->jour);
             in>>(a->mois);
             in>>(a->annee);
    return(in);
        }
  bool date::operator == (const date& d)
      {
            if((jour==d.jour)&&(mois==d.mois)&&(annee==d.annee))
            return(true);
            return(false);

        }
  bool date::operator < (date& other)
  {
      if  ((other>*this))
        return(true) ;
      return(false);

  }
  bool date::operator > (date& other)
  {
    if (annee > other.annee)
        {return true;}
    else if (annee == other.annee && mois > other.mois) {
      return true;
    } else if (annee== other.annee&& mois== other.mois && jour > other.jour) {
      return true;
    }
    return false;
  }

date::~date()
{
    //dtor
}
```

22) Couble.h (class template) :

```
#ifndef COUPLE_H
#define COUPLE_H

#include "date.h"
#include <iostream>

using namespace std;

template<class t1, class t2>
class couple
{
protected:
    t1 x1;
    t2 x2;
public:
    couple(t1, t2);
    couple();
    template<class T, class V>
    friend ostream& operator<< (ostream& os, const couple<T, V>& c);
    template<class T, class V>
    friend istream& operator>> (istream& in , couple<T, V>& c);

    t1 get_x1();
    void set_x1(t1);
    void set_x2(t2);
    t2 get_x2();

    virtual ~couple();
};

template<class t, class v>
couple<t, v>::couple(t a, v b)
{
    x1 = a;
    x2 = b;
}

template<class t, class v>
couple<t, v>::couple()
{

}

template<class t, class v>
couple<t, v>::~couple()
{
```

```
    // No explicit clean-up needed
}

template<class a, class b>
istream& operator>> (istream& in, couple<a, b>& c)
{
    in >> c.x1;
    in >> c.x2;
    return in;
}

template<class T, class U>
ostream& operator<<(ostream& out, const couple<T,U>& c)
{
    out << c.x1 << endl;
    out << c.x2 << endl;
    return out;
}

template<class T, class U>
T couple<T, U>::get_x1()
{
    return x1;
}

template<class T, class U>
void couple<T, U>::set_x1(T a)
{
    x1 = a;
}

template<class T, class U>
void couple<T, U>::set_x2(U a)
{
    x2 = a;
}

template<class T, class U>
U couple<T, U>::get_x2()
{
    return x2;
}

#endif // COUPLE_H
```

23) Triple.h (class herité du classe couple) :

```
#ifndef TRIPLE_H
#define TRIPLE_H

#include <couple.h>

#include "date.h"

#include <iostream>

using namespace std;

template<class t1, class t2, class t3>
class triple : public couple<t1, t2>
{

    t3 x3;
    public:
    triple();
    void set_x3(t3);
    t3 get_x3();
    triple(t1,t2,t3);
    template<class a, class b, class c>
    friend ostream& operator<< (ostream& os, const triple<a, b, c>& h);
    template<class T, class U, class V>
    friend istream& operator>> (istream& in, triple<T, U, V>& h);

    virtual ~triple();

    protected:

    private:
};
template<class T, class U, class V>
istream& operator>>(istream& in, triple<T, U, V>& h) {
    in >> h.x1;
    in >> h.x2;
    in >> h.x3;
    return in;
}

template<class T, class U, class V>
ostream& operator<<(ostream& out, const triple<T, U, V>& h) {
    out << h.x1 << endl;
    out << h.x2 << endl;
    out << h.x3 << endl;
    return out;
```

```
}
template<class t, class v,class u>
triple<t, v,u>::triple() {

}
template<class q, class s,class d>
triple<q, s,d>::triple(q a, s b,d c)
 {
    this->x1 = a;
    this->x2 = b;
    this->x3= c;
}
template<class t, class v,class u>
triple<t, v,u>::~triple() {

}
template<class T, class U, class V>
void triple<T, U,V>::set_x3(V a)
{
    x3 = a;
}

template<class T, class U, class V>
V triple<T, U,V>::get_x3()
{
    return x3;
}

#endif // TRIPLE_H
```

24) Main.cpp :

```
#include <iostream>
#include "Vehicule.h"
#include "voiture_Poid_lourd .h"
#include "voiture.h"
#include "Camion.h"
#include "bus.h"
#include "Personne.h"
#include "Proprietaire.h"
#include "Employee.h"
#include "Visite.h"
#include "Centre_visite.h"
#include "couple.h"
#include "triple.h"

#include "date.h"
```

```
#include <fstream>


using namespace std;

int main()
{
fstream f;
Centre_visite b;
  f.open("D:\projet c++.txt",ios::out|ios::in);
  f>>&b;
  cout<<endl;
  cout<<endl;
 cout << "°°°°°°°°Bienvenue dans notre centre de visite de voiture !°°°°°°°°"
<< endl;
  do
  {   menu:
      cout<<endl;
      cout<<"1) mode utilisateur "<<endl;
      cout <<"2) mode administatreur "<<endl;
      cout <<"3) quitter"<<endl;

      int choix;
      cin >>choix;
      if(choix == 3)
        break;
      else if (choix == 1)
      {       menu1:
              cout <<"*****************vous etes en mode
utlisateur**************"<<endl;
              cout<<"1) ajouter une visite  "<<endl;
               cout<<"2) la resultat de la visite  "<<endl;
              cout <<"3) retour "<<endl;
                    int choix1;
                    cin>>choix1;
                    if(choix1 == 1)
                  {      cout<<endl<<"donner les donnes du nouvelle
visite"<<endl;
                         b.ajouter_visite();
                        f.close();
                        f.open("D:\projet c++.txt",ios::out|ios::in |
ios::trunc);
                        f.seekg(0);
                        f<<&b;
                     goto menu1;
                  }
                  if(choix1 == 2)
                  {     cout<<endl<<"donner les coordonnees de la
visite"<<endl;
```

```
                        b.affiche_resultat();
                        cout<<"\n";
                        f.close();
                        f.open("D:\projet c++.txt",ios::out|ios::in |
ios::trunc);
                        f.seekg(0);
                        f<<&b;

                       goto menu1;
                   }
                    else if(choix1 == 3) goto menu;
                    else goto menu1;
      }
            else if (choix == 2)
      {
          kafteji:
          cout<<"saisir un mot de passe"<<endl;
          string x;
          cin>>x;
          if (b.verfie_m_d_p(x)==true)
              {       menu2:
              cout<<"***************vous etes en mode
administrateur*************"<<endl;
              cout<<"1) ajouter une employee  "<<endl;
              cout<<"2)  ajouter mot passe   "<<endl;
              cout<<"3)  supprimer les visites  eronees  "<<endl;
              cout<<"4)  changer la resultat d une visite  "<<endl;
              cout<<"5)  afficher les donnees du centre "<<endl;
              cout<<"6) retour "<<endl;
                    int choix2;
                    cin>>choix2;
                    if(choix2 == 1)
                  {    cout<<endl<<"donner les donnes du nouveau
employee"<<endl;
                       b.ajouter_emp();
                       f.close();
                       f.open("D:\projet c++.txt",ios::out|ios::in |
ios::trunc);
                       f.seekg(0);
                       f<<&b;
                       goto menu2;
                   }
                       else if(choix2 == 2)
                   {     cout<<endl<<"donner le nouveau mot de passe"<<endl;
                        b.ajouter_mdp();
                        f.close();
                        f.open("D:\projet c++.txt",ios::out|ios::in |
ios::trunc);
                        f.seekg(0);
```

```
                        f<<&b;
                       goto menu2;
                  }
                     else if(choix2 == 3)
                  {
                       b.supprimer_erone();
                       f.close();
                       f.open("D:\projet c++.txt",ios::out|ios::in |
ios::trunc);
                       f.seekg(0);
                       f<<&b;
                       goto menu2;
                  }
                     else if(choix2 == 4)
                  {   cout<<endl<<"donner les coordonnees de la visite"<<endl;
                       b.test_jour();
                       f.close();
                       f.open("D:\projet c++.txt",ios::out|ios::in |
ios::trunc);
                       f.seekg(0);
                       f<<&b;
                       goto menu2;
                  }
                 else if(choix2 == 5)
                 {cout<<b;
                 goto menu2;
                 }


                   else if(choix2 == 6) goto menu;
                   else goto menu2;

      }
          else
          {
              cout << "taper 1 pour resaisir le mot de passe "<<endl;
              int k;
              cin>>k;
              if (k== 1)
                goto kafteji ;
          }
     }
  }
  while(1);


    return 0;
}
```

3) Fichier :

projet c++ - Bloc-notes

Fichier Modifier Affichage

```
Centre_Visite_Ariana
6
2 2  11
mohmed_aziz
aziz@gmail.com
 4 5 2000
1 59487362
125
1 voiture  1234  2 5 2023  4 4 2001 benz mercdis 123 4 4 essence manuel 0 1 152


 2 5 2023
1234
en_attend
150
 222
Mohamed
mohamed@gmail.com
 2 2 1980
1 12345678
8
1000
 7 4 2010
technicien
0
 33
slah
slah@gmail.com
 7 12 1970
1 25789614
159875332
1 bus  6969  6 5 2023  5 11 2017 volvo 2 1478 3 2 10 8 touriste 60 1  2 7 2019
3 123
456
789


 2 5 2023
6969
```

```
 2 5 2023
6969
en_attend
300
 333
salah
salah@gmail.com
 7 4 1980
1 45632189
8
1000
 4 9 2018
technicien
1 44
 18 4 2023
3

1  22
yassine
yassine@gmail.com
 2 1 2000
1 95847623
14523698
1 camion  5698  4 5 2023  6 5 2015 x2 clio 154 2 2 4 6 petrole 500 0 2 123
456
losange
rouge
feu


 3 5 2023
5698
en_attend
200
 222
mohamed
mohamed@gmail.com
 4 1 2000
1 98652347
```

```
1 98652347
8
1000
 1 1 2020
technicien
0
4 111
Ali
ali@gmail.com
 6 5 1990
2 28212530
30405060
8
1500
 20 4 2002
Chef
2 1111
 2 5 2023
7
2222
 1 3 2023
3
222
Mohamed
Mohamed@gmail.com
 2 2 1980
1 12345678
8
1000
 7 4 2010
technicien
0 333
Salah
salah@gmail.com
 20 10 1997
2 45127863
52417863
8
1000
 17 8 2019
```

```
Arij
arij@yahoo.fr
 16 8 1998
2 36475921
21457896
6
1600
 14 9 2021
comptable
1 3333
 4 2 2023
10
1 0000
```

4) Menu :

```
        Bienvenue dans notre centre de visite de voiture !

1) mode utilisateur
2) mode administatreur
3) quitter
```

```
******************vous etes en mode utlisateur**************
1) ajouter une visite
2) la resultat de la visite
3) retour
```

```
saisir un mot de passe
0000
***************vous etes en mode administrateur*************
1) ajouter une employee
2)  ajouter mot passe
3)  supprimer les visites  eronees
4)  changer la resultat d une visite
5)  afficher les donnees du centre
6) retour
```

5) Affichage final :

```
le  nom estCentre_Visite_Ariana
le  nobre de visite  maximal par jour est6
***************les donnÚes de la visite******************
le  marticule est1234
le  montant est150
le  date visite  est2023:5:2

le donnes de Employee est :id est 222
nom est Mohamed
mail estmohamed@gmail.com
date de naissance est1980:2:2

le nombre de telphone sont
12345678
les heures de travaille sont8
le salaire est1000
la date d'embauche est2010:4:7

la fonction esttechnicien

 le donnes de Proprietaire est :id est 11
nom est mohmed_aziz
mail estaziz@gmail.com
date de naissance est2000:5:4

le nombre de telphone sont
59487362
num_permis est 125
le 0 vehicule est Num_chasis est 123
marque est mercdis
model est benz
matricule est 1234
date_utilis est 2001:4:4

date_expiree est 2023:5:2

nb_porte 4
nb_cylindre est 4
carburant est essence
type_transmis est manuel
```

```
type_transmis est manuel
le 0 conducteur est 152

***************les donnÚes de la visite******************
le  marticule est6969
le  montant est300
le  date visite  est2023:5:2

le donnes de Employee est :id est 333
nom est salah
mail estsalah@gmail.com
date de naissance est1980:4:7

le nombre de telphone sont
45632189
les heures de travaille sont8
le salaire est1000
la date d'embauche est2018:9:4

la fonction esttechnicien
l'identifier de vacance 44
la date et nombre de jour2023:4:18

3


 le donnes de Proprietaire est :id est 33
nom est slah
mail estslah@gmail.com
date de naissance est1970:12:7

le nombre de telphone sont
25789614
num_permis est 159875332
le 0 vehicule est Num_chasis est 1478
marque est 2
model est volvo
matricule est 6969
date_utilis est 2017:11:5
```

```
date_expiree est 2023:5:6

le 0 historique date est 2019:7:2

largeurs est 2
longueur est 10
hauteur est 3
nb_essieux est 8
type est touriste
nb_place est 60
le 0 conducteur est 123
le 1 conducteur est 456
le 2 conducteur est 789

***************les donnÚes de la visite******************
le  marticule est5698
le  montant est200
le  date visite  est2023:5:3

le donnes de Employee est :id est 222
nom est mohamed
mail estmohamed@gmail.com
date de naissance est2000:1:4

le nombre de telphone sont
98652347
les heures de travaille sont8
le salaire est1000
la date d'embauche est2020:1:1

la fonction esttechnicien

 le donnes de Proprietaire est :id est 22
nom est yassine
mail estyassine@gmail.com
date de naissance est2000:1:2

le nombre de telphone sont
95847623
```

```
num_permis est 14523698
le 0 vehicule est Num_chasis est 154
marque est clio
model est x2
matricule est 5698
date_utilis est 2015:5:6

date_expiree est 2023:5:4

largeurs est 2
longueur est 4
hauteur est 2
nb_essieux est 6
type_charge est petrole
capacite_charge  est 500
le 0 conducteur est 123
le 1 conducteur est 456
la forme de signe est losange
la colore de signe est rouge
le nom de signe est feu

id est 111
nom est Ali
mail estali@gmail.com
date de naissance est1990:5:6

le nombre de telphone sont
28212530
30405060
les heures de travaille sont8
le salaire est1500
la date d'embauche est2002:4:20

la fonction estChef
l'identifier de vacance 1111
la date et nombre de jour2023:5:2

7
```

```
7

l'identifier de vacance 2222
la date et nombre de jour2023:3:1

3

id est 222
nom est Mohamed
mail estMohamed@gmail.com
date de naissance est1980:2:2

le nombre de telphone sont
12345678
les heures de travaille sont8
le salaire est1000
la date d'embauche est2010:4:7

la fonction esttechnicien
id est 333
nom est Salah
mail estsalah@gmail.com
date de naissance est1997:10:20

le nombre de telphone sont
45127863
52417863
les heures de travaille sont8
le salaire est1000
la date d'embauche est2019:8:17

la fonction esttechnicien
id est 444
nom est Arij
mail estarij@yahoo.fr
date de naissance est1998:8:16

le nombre de telphone sont
```

```
le nombre de telphone sont
36475921
21457896
les heures de travaille sont6
le salaire est1600
la date d'embauche est2021:9:14

la fonction estcomptable
l'identifier de vacance 3333
la date et nombre de jour2023:2:4

10

 1 0000
```