```cpp
/*
-----------------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : main.cpp
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Programme principal qui permet de tester les classes Personne, Adresse
              et Date du laboratoire 2. La création de personne via des données
              ainsi que celle via une saisie utilisateur est testée. Ce programme
              va également afficher toutes les informations des personnes saisies,
              les noms et prénoms des personnes possédant un emploi et celles
              dont l'attribut npa de l'adresse de la personne correspondant à une
              valeur saisie par l'utilisateur.

Remarque(s) :   ---

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------------
*/

#include <iostream>
#include <string>
#include <vector>
#include "Adresse.h"
#include "Date.h"
#include "Personne.h"

using namespace std;
using vP = vector<Personne>;

int main() {
    vP personnes;
    Date naissance((short)22, (Mois) 2, (long long) 2000);
    Adresse employeurAdresse(
        "6666", "Hante-Ville", "Danemark",
        "Route de Cheseaux", "18b");
    Personne personne(
        "Marcel-Jean", "Jean", naissance, employeurAdresse, true,
        "Borris Leutard", employeurAdresse, 10000000);
    personnes.push_back(personne);

    Date naissance2((short)23, (Mois)4, (long long)1905);
    Adresse employeurAdresse2(
        "18950", "Amsterdam", "Pays-Bas",
        "Place de la Gare", "1");
    Personne personne2("Marcel-Jean", "Marie", naissance,
        employeurAdresse, true, "Borris Leutard", employeurAdresse, 250);
    personnes.push_back(personne2);

    Personne personne3;
    cin >> personne3;
    personnes.push_back(personne3);

    for (Personne p : personnes) {
        cout << p << endl;
    }

    for (Personne p : personnes) {
        if (p.getIsEmployee()) {
            cout << p.getNom() << " " << p.getPrenom() << endl;
        }
    }

    cout << endl << "Saissiez un NPA a rechercher : ";
    string npaSearched;
```

```
67      getline(cin, npaSearched);
68      cout << "Personnes habitant au NPA : " << npaSearched << endl;
69      cout << "-------------------------" << endl;
70
71      for (Personne p : personnes) {
72          if (!((p.getAdresse().getNpa()).compare(npaSearched))) {
73              cout << "- " << p << endl;
74          }
75      }
76      cout << "-------------------------" << endl << endl;
77
78      cout << "Appuyez sur une touche pour continuer...";
79      cin.get();
80      return EXIT_SUCCESS;
81  }
```

```
1   /*
2   ------------------------------------------------------------------------------------
3   Laboratoire : 02 - Personnes
4   Fichier     : Adresse.h
5   Auteur(s)   : Frédéric Korradi et Adrien Barth
6   Date        : 28.02.2017
7
8   But         : Fichier de définition pour Adresse.cpp
9
10  Remarque(s) :   ---
11
12  Compilateur : MinGW-g++ 4.8.1
13  ------------------------------------------------------------------------------------
14  */
15
16  #ifndef ADRESSE_H
17  #define ADRESSE_H
18
19  #include <string>
20
21  class Adresse;
22
23  std::ostream& operator << (std::ostream&, const Adresse&);
24  std::istream& operator >> (std::istream&, Adresse&);
25
26  class Adresse {
27      friend std::ostream& operator << (std::ostream&, const Adresse&);
28      friend std::istream& operator >> (std::istream&, Adresse&);
29
30  public:
31      Adresse();
32      Adresse( const std::string&,  const std::string&,  const std::string&,  const
        std::string&,  const std::string&);
33      void setRue( const std::string&);
34      void setNo( const std::string&);
35      void setNpa(const std::string&);
36      void setVille(const std::string&);
37      void setPays(const std::string&);
38      std::string getRue() const;
39      std::string getNo() const;
40      std::string getNpa() const;
41      std::string getVille() const;
42      std::string getPays() const;
43
44  private:
45      std::string rue = "";
46      std::string no = "";
47      std::string npa;
48      std::string ville;
49      std::string pays;
50  };
51
52  #endif
```

```cpp
/*
-----------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : Adresse.cpp
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Classe représentant une adresse et permettant d'y accéder et de la
manipuler

Remarque(s) : - Les atributs rue et numéro sont optionnels
- Les attributs npa, ville et pays sont obligatoires.

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------
*/

#include "Adresse.h"
#include "utilities.h"
#include <iostream>

using namespace std;

Adresse::Adresse() {
}

Adresse::Adresse(const std::string& npa, const std::string& ville,
    const std::string& pays, const std::string& rue = "", const std::string& no = "")
    :rue(rue), no(no), npa(npa),ville(ville),pays(pays) {
}


ostream& operator << (ostream& os, const Adresse& adresse) {

    if (!adresse.rue.empty()) {
        os << adresse.rue << " " << adresse.no << endl;
    }

    os << adresse.npa << " " << adresse.ville << endl
        << adresse.pays << endl;

    return os;
}

istream& operator >> (istream& is, Adresse& a) {
    cout << "Rue [optionnel] : ";
    getline(is, a.rue);

    if (!a.rue.empty()) {
        cout << "Numero [optionnel] : ";
        getline(is, a.no);
    }

    a.npa = lireString("NPA: ");
    a.ville = lireString("Ville: ");
    a.pays = lireString("Pays: ");

    return is;
}

void Adresse::setRue(const string& rue) {
    this->rue = rue;
}

void Adresse::setNo(const string& no) {
    this->no = no;
```

```
67  }
68
69  void Adresse::setNpa(const string& npa) {
70      this->npa = npa;
71  }
72
73  void Adresse::setVille(const string& ville) {
74      this->ville = ville;
75  }
76
77  void Adresse::setPays(const string& pays) {
78      this->pays = pays;
79  }
80
81  string Adresse::getRue() const {
82      return rue;
83  }
84
85  string Adresse::getNo() const {
86      return no;
87  }
88
89  string Adresse::getNpa() const {
90      return npa;
91  }
92
93  string Adresse::getVille() const {
94      return ville;
95  }
96
97  string Adresse::getPays() const {
98      return pays;
99  }
```

```
/*
-----------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : Date.h
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Fichier de définition pour Date.cpp

Remarque(s) : La variable annee n'est pas signée et est un long long, dans le cas
              où l'on devrait utiliser cette classe pour une date précédent J.C.

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------
*/

#ifndef DATE_H
#define DATE_H

#include <string>

enum class Mois {
    JANVIER, FEVRIER, MARS, AVRIL,
    MAI, JUIN, JUILLET, AOUT, SEPTEMBRE,
    OCTOBRE, NOVEMBRE, DECEMBRE
};

class Date;

std::ostream& operator << (std::ostream&, const Date&);
std::istream& operator >> (std::istream& os, Date&);
std::istream& operator >> (std::istream& os, Mois&);

class Date {
    friend std::ostream& operator << (std::ostream&, const Date&);
    friend std::istream& operator >> (std::istream&, Date&);
    friend std::istream& operator >> (std::istream&, Mois&);

public:
    Date();
    Date(unsigned short jour, Mois mois, unsigned annee);
    void setJour(const unsigned short& jour);
    void setMois(const Mois& mois);
    void setAnnee(const unsigned& annee);
    void setDate(const unsigned short& jour, const Mois& mois, const unsigned& annee);
    short getJour() const;
    Mois getMois() const;
    long long getAnnee()  const;
    bool getValid() const;

private:
    unsigned short jour;
    Mois mois;
    unsigned annee;
    bool dateValide = 1;
    const std::string MOIS[12] = { "janvier", "fevrier", "mars", "avril", "mais", "juin",
        "juillet", "aout", "septembre", "octobre", "novembre",
        "decembre" };
};

#endif
```

```cpp
/*
-----------------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : Date.cpp
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Classe représentant une adresse et permettant d'y accéder et de la
              manipuler.

Remarque(s) : La date utilise un type fortement énuméré pour la saisie du mois.

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------------
*/

#include "Date.h"
#include "utilities.h"
#include <iostream>

using namespace std;

Date::Date()
    :jour(1), mois(Mois::JANVIER), annee(1900) {
}

ostream& operator << (ostream& os, const Date& date) {
    os << date.jour << "-" << date.MOIS[(int)date.mois]<< "-" << date.annee;

    return os;
}
istream& operator >> (istream& is, Mois& mois) {
    const string ERREUR = "Saisie invalide, recommencez...";
    mois = Mois(lireInt(1, 12, "Mois [1-12]: ", ERREUR));

    return is;
}

istream& operator >> (istream& is, Date& date)  {
    const string ERREUR = "Saisie invalide, recommencez...";

    date.jour = lireInt(1, 31, "Jour: ", ERREUR);
    is >> date.mois;
    date.annee = lireInt(1900, 10000, "Annee: ", ERREUR);

    return is;
}

Date::Date(unsigned short jour, Mois mois, unsigned annee)
    :jour(jour), mois(mois), annee(annee) {
}

void Date::setJour(const unsigned short& jour) {
    this->jour = jour;
}

void Date::setMois(const Mois& mois) {
    this->mois = mois;
}

void Date::setAnnee(const unsigned& annee) {
    this->annee = annee;
}

void Date::setDate(const unsigned short& jour, const Mois& mois, const unsigned& annee) {
    setJour(jour);
```

```
67      setMois(mois);
68      setAnnee(annee);
69  }
70
71  short Date::getJour() const {
72      return jour;
73  }
74
75  Mois Date::getMois() const {
76      return mois;
77  }
78
79  long long Date::getAnnee() const {
80      return annee;
81  }
82
83  bool Date::getValid() const {
84      return dateValide;
85  }
```

```
/*
-----------------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : Personne.h
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Fichier de définition pour Personne.cpp

Remarque(s) :   ---

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------------
*/

#ifndef PERSONNE_H
#define PERSONNE_H

#include <string>
#include "Date.h"
#include "Adresse.h"

class Personne;

std::ostream& operator << (std::ostream&, const Personne&);
std::istream& operator >> (std::istream&, Personne&);

class Personne {
    friend std::ostream& operator << (std::ostream&, const Personne&);
    friend std::istream& operator >> (std::istream&, Personne&);

public:
    Personne();
    Personne(const std::string&, const std::string&, const Date&,
            const Adresse&, const bool&, const std::string&,
             const Adresse&, const unsigned int&);
    void setPrenom(const std::string&);
    void setNom(const std::string&);
    void setNaissance(const Date&);
    void setAdresse(const Adresse&);
    void setIsEmployee(const bool&);
    void setEmployeurNom(const std::string&);
    void setEmployeurAdresse(const Adresse&);
    void setSalaireAnnuel(const unsigned int&);
    std::string getPrenom() const;
    std::string getNom() const;
    Date getNaissance() const;
    Adresse getAdresse() const;
    bool getIsEmployee();
    std::string getEmployeurNom() const;
    Adresse getEmployeurAdresse() const;
    unsigned int getSalaireAnnuel() const;

private:
    std::string prenom;
    std::string nom;
    Date naissance;
    Adresse adresse;
    bool isEmployee;
    std::string employeurNom;
    Adresse employeurAdresse;
    unsigned int salaireAnnuel;
};

#endif
```

```cpp
/*
-----------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : Personne.cpp
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Classe représentant une adresse et permettant d'y accéder et de la
              manipuler.

Remarque(s) : - Les attributs nom, prenom, naissance sont obligatoires.
              - Les attirubts naissance, adresse et les informations sur l'employeur
              sont optionnels.

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------
*/

#include "Personne.h"
#include "utilities.h"
#include <iostream>

using namespace std;

Personne::Personne() {
}

Personne::Personne(const string& nom, const string& prenom, const Date& naissance,
    const Adresse& adresse,const bool& isEmployee, const string& employeurNom,
    const Adresse& employeurAdresse, const unsigned int& salaireAnnuel)
    :nom(nom), prenom(prenom),
    naissance(naissance),adresse(adresse),isEmployee(isEmployee),
    employeurNom(employeurNom), employeurAdresse(employeurAdresse),
    salaireAnnuel(salaireAnnuel) {
}

ostream& operator << (ostream& os, const Personne& p) {
    os << p.prenom << " " << p.nom << endl
        << "Ne le: " << p.naissance << endl
        << "Adresse:" << endl << p.adresse << endl;

    if (p.isEmployee) {
        os << "Employeur:" << endl
            << p.employeurNom << endl
            << p.employeurAdresse << endl
            << "Salaire annuel: " << p.salaireAnnuel;
    }

    return os;
}

istream& operator >> (istream& is, Personne& p) {
    const string ERREUR = "Saisie invalide, recommencez...";

    p.nom = lireString("Votre nom: ");
    p.prenom = lireString("Votre prenom: ");

    if (lireInt(0, 1,
        "Voulez-vous saisir une adresse ([0] Non, [1] Oui) ? ",
        ERREUR)) {
        is >> p.adresse;
    }

    if (lireInt(0, 1,
        "Voulez-vous saisir votre date de naissance ([0] Non, [1] Oui) ? ",
        ERREUR)) {
```

```
65          is >> p.naissance;
66      }
67
68      if (lireInt(0, 1,
69          "Avez-vous un employeur ([0] Non, [1] Oui) ? ",
70          ERREUR)) {
71          p.isEmployee = true;
72
73          p.employeurNom = lireString("Nom de l'employeur: ");
74
75          cout << "Adresse de l'employeur : " << endl;
76          is >> p.employeurAdresse;
77
78          cout << "Salaire annuel : ";
79          p.salaireAnnuel = lireInt(0, 1e+9,
80              "Quelle est votre salaire annuel ?",
81              ERREUR);
82      }
83      else {
84          p.isEmployee = false;
85      }
86
87      return is;
88  }
89
90  void Personne::setPrenom(const string& prenom) {
91      this->prenom = prenom;
92  }
93
94  void Personne::setNom(const string& nom) {
95      this->nom = nom;
96  }
97
98  void Personne::setNaissance(const Date& naissance) {
99      this->naissance.setJour(naissance.getJour());
100     this->naissance.setAnnee(naissance.getAnnee());
101     this->naissance.setMois(naissance.getMois());
102 }
103
104 void Personne::setAdresse(const Adresse& adresse) {
105     this->adresse = adresse;
106 }
107
108 void Personne::setIsEmployee(const bool& isEmployee) {
109     this->isEmployee = isEmployee;
110 }
111
112 void Personne::setEmployeurNom(const string&employeurNom) {
113     this->employeurNom = employeurNom;
114 }
115
116 void Personne::setEmployeurAdresse(const Adresse& employeurAdresse) {
117     this->employeurAdresse = employeurAdresse;
118 }
119
120 void Personne::setSalaireAnnuel(const unsigned int& salaireAnnuel) {
121     this->salaireAnnuel = salaireAnnuel;
122 }
123
124 string Personne::getPrenom() const {
125     return prenom;
126 }
127
128 string Personne::getNom() const {
129     return nom;
130 }
```

```
131
132 Date Personne::getNaissance() const {
133     return naissance;
134 }
135
136 Adresse Personne::getAdresse() const {
137     return adresse;
138 }
139
140 bool Personne::getIsEmployee() {
141     return isEmployee;
142 }
143
144 string Personne::getEmployeurNom() const {
145     return employeurNom;
146 }
147
148 Adresse Personne::getEmployeurAdresse() const {
149     return employeurAdresse;
150 }
151
152 unsigned int Personne::getSalaireAnnuel() const {
153     return salaireAnnuel;
154 }
```

```
1   /*
2   -----------------------------------------------------------------------------
3   Laboratoire : 02 - Personnes
4   Fichier     : utilities.h
5   Auteur(s)   : Frédéric Korradi et Adrien Barth
6   Date        : 28.02.2017
7
8   But         : Fichier de définitions pour utilities.cpp
9
10  Remarque(s) :   ---
11
12  Compilateur : MinGW-g++ 4.8.1
13  -----------------------------------------------------------------------------
14  */
15
16  #ifndef UTILITIES_H
17  #define UTILITIES_H
18
19  #include <string>
20
21  int lireInt(int, int, const std::string, const std::string);
22  std::string lireString(const std::string);
23
24  #endif
```

```cpp
/*
-----------------------------------------------------------------------------------
Laboratoire : 02 - Personnes
Fichier     : utilities.h
Auteur(s)   : Frédéric Korradi et Adrien Barth
Date        : 28.02.2017

But         : Fonctions permettant d'effectuer des saisies utilisateurs.

Remarque(s) :   ---

Compilateur : MinGW-g++ 4.8.1
-----------------------------------------------------------------------------------
*/

#ifndef UTILITIES_CPP
#define UTILITIES_CPP

#include <iostream>
#include <string>

using namespace std;

int lireInt(int min, int max, const string message, const string erreur) {

    if (min > max) {
        int tmp = min;
        min = max;
        max = tmp;
    }

    int nombre;

    while (true) {
        cout << message;
        cin >> nombre;

        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        cin.sync();

        if (nombre < min || max < nombre) {
            cout << erreur << endl;
        }
        else {
            return nombre;
        }
    }
}

string lireString(const string message) {
    string saisie;

    do {
        cout << message;
        getline(cin, saisie);
    } while (saisie.empty());

    return saisie;
}

#endif // !UTILITIES_CPP
```