TP 4 - Problem 6 - I

```
#include <exception>
#include <iostream>

#define SHOW(arg) std::cout << "Macro SHOW “" #arg "”: " << (arg) << '\n';

// Base class.
class Person {
public:
  // Ctor.
  Person(std::string fName, std::string lName, int birth_year)
      : fName_{fName}, lName_{lName}, birth_year_{birth_year} {
    if (fName_.empty())
      throw std::invalid_argument{"Person: the first name is empty."};
    if (lName_.empty())
      throw std::invalid_argument{"Person: the last name is empty."};
    if (birth_year_ <= 0)
      throw std::domain_error{"Person: the birth year is negative."};
  }
  // Dtor.
```

TP 4 - Problem 6 - II

```cpp
  ~Person() {}
  // Push on the console output stream a representation of a Person
  // instance.
  virtual void print() const {
    std::cout << "Person{" << get_name() << ' ' << birth_year_ << "}\n";
  }
  // Returns the birthday year.
  int get_year_birth() const { return birth_year_; }
  // Returns the full name.
  std::string get_name() const { return fName_ + ' ' + lName_; }

protected:
  std::string const fName_;
  std::string const lName_;
  int const birth_year_;
};

class Employee : public Person {
public:
  // Ctor: call the base class ctor and set the “sal_” and the
```

TP 4 - Problem 6 - III

```cpp
    // “employment_date_” data members.
    Employee(std::string fName, std::string lName, double sal,
             int birth_year, int employment_date)
        : Person{fName, lName, birth_year}, sal_{sal},
          employment_date_{employment_date} {}
    // Push on the console output stream a representation of a Employee
    // instance.
    virtual void print() const {
      std::cout << "Employee{" << get_name() << ' ' << birth_year_ << ' '
                << sal_ << ' ' << employment_date_ << "}\n";
    }
    // Reset the salary.
    void SetSalary(double sal) {
      if (sal <= 0)
        throw std::domain_error{
            "Employee::SetSalary: the salary is negative."};
      sal_ = sal;
    }
    // Returns a copy of the salary.
    double GetSalary() { return sal_; }
```

TP 4 - Problem 6 - IV

```cpp

protected:
  double sal_;
  int const employment_date_;
};

class Manager : public Employee {
public:
  // Ctor: call the “Employee” ctor and set the “com_” data member.
  Manager(std::string fName, std::string lName, double sal, int birth_year,
          int employment_date, double com)
      : Employee{fName, lName, sal, birth_year, employment_date},
        com_{com} {}
  // Push on the console output stream a representation of a Manager
  // instance.
  virtual void print() const {
    std::cout << "Manager{" << get_name() << ' ' << birth_year_ << ' '
              << sal_ << ' ' << employment_date_ << ' ' << com_ << "}\n";
  }
  // Reset the commission.
```

TP 4 - Problem 6 - V

```cpp
  void SetCom(double com) {
    if (com <= 0)
      throw std::domain_error{
          "Employee::SetSalary: the commission is negative."};
    com_ = com;
  }
  // Returns a copy of the commission.
  double GetCom() { return com_; }

protected:
  double com_;
};

class Clerk : public Employee {
public:
  // Ctor: call the “Employee” ctor and set the “m_” data member.
  Clerk(std::string fName, std::string lName, double sal, int birth_year,
        Manager const *m, int employment_date)
      : Employee{fName, lName, sal, birth_year, employment_date}, m_{m} {}
  // Push on the console output stream a representation of a Clerk
```

TP 4 - Problem 6 - VI

```
  // instance.
  virtual void print() const {
    std::cout << "Clerk{" << get_name() << ' ' << birth_year_ << ' '
              << sal_ << ' ' << employment_date_ << " Manager("
              << m_->get_name() << ")}\n";
  }
  // Reset the manager.
  void Set(Manager const *m) { m_ = m; }

protected:
  Manager const *m_;
};

void print_team(Employee const *team[], int n) {
  std::cout << "print_team output\n";
  for (int i{}; i < n; ++i)
    team[i]->print();
}

int main() {
```

TP 4 - Problem 6 - VII

```
    Person{"François", "Legendre", 1960}.print();
    Employee{"François", "Legendre", 120000, 1960, 1990}.print();
    Employee dupont{"Jean", "Dupont", 50000, 1980, 2005};
    Employee martin{"Catherine", "Martin", 50000, 1980, 2005};
    Manager legendre{"François", "Legendre", 120000, 1960, 1990, 2000};
    Clerk dupond{"Louise", "Dupond", 60000, 1990, &legendre, 2015};
    Employee const *team[]{&dupont, &martin, &legendre, &dupond};
    int constexpr team_size{sizeof team / sizeof team[0]};
    print_team(team, team_size);

    return 0;
}
```

TP 4 - Problem 6 - VIII

Output:

```
Person{François Legendre 1960}
Employee{François Legendre 1960 120000 1990}
print_team output
Employee{Jean Dupont 1980 50000 2005}
Employee{Catherine Martin 1980 50000 2005}
Manager{François Legendre 1960 120000 1990 2000}
Clerk{Louise Dupond 1990 60000 2015 Manager(François Legendre)}
```