Week 4

Exercise 33

../33–34/person/person.h

```
// Person class: interface header

#ifndef INCLUDED_PERSON_
#define INCLUDED_PERSON_

#include <string>
#include <iostream>
#include <sstream>

class Person
{
  std::string d_name;      // name of person
  std::string d_address;  // address field
  std::string d_phone;    // telephone number
  size_t      d_mass;      // the mass in kg.

  public:
    std::string const &name()     const;
    std::string const &address()  const;
    std::string const &phone()    const;
    size_t mass()                 const;
    // Getters

    void insert(std::ostream &outputStream); // Storing data
    void extract(std::istream &inputStream);  // Extracting data

  private:
    void setName(std::string const &name);
    void setAddress(std::string const &address);
    void setPhone(std::string const &phone);
    void setMass(size_t mass);
    // Setters

};

#endif
```

../33–34/person/person.ih

```
// Person class: implementation internal header

#include "person.h"

#define CERR std::cerr << __FILE__": "

using namespace std;
```

../33–34/person/basicFunctions.cc

```cpp
// Person member functions: basic setters and getters

#include "person.ih"

void Person::setName(string const &name)
{
  d_name = name;
}

string const &Person::name() const
{
  return d_name;
}

void Person::setAddress(string const &address)
{
  d_address = address;
}

string const &Person::address() const
{
  return d_address;
}

// Phone number setter defined seperately

string const &Person::phone() const
{
  return d_phone;
}

void Person::setMass(size_t mass)
{
  d_mass = mass;
}

size_t Person::mass() const
{
  return d_mass;
}
```

../33–34/person/extract.cc

```cpp
// Person member function: extract person data from istream

#include "person.ih"

void Person::extract(istream &inputStream)
{
  string inputString;
  getline(inputStream, inputString);     // Get full line
  istringstream ss(inputString);         // Transfer line to istringstream
  size_t index = 0;                      // Initialise counter for switch

  while (getline(ss, inputString, ',')) // While an element can still be
      extracted
  {
    switch (index)                       // Assign the object variables in order
    {
      case 0:
        setName(inputString);
        break;
      case 1:
        setAddress(inputString);
        break;
      case 2:
        setPhone(inputString);
        break;
      case 3:
        setMass(stoi(inputString));
        break;
    }
  ++index;
  }
}
```

../33–34/person/insert.cc

```cpp
// Person member function: insert data into ostream

#include "person.ih"

void Person::insert(ostream &outputStream)
{
  outputStream << "NAME:    " << name()     << '\n';
  outputStream << "ADDRESS: " << address()  << '\n';
  outputStream << "PHONE:   " << phone()    << '\n';
  outputStream << "MASS:    " << mass()     << '\n';
}
// Inserts all object characteristics into ostream. It was assumed that the
// variable identifiers were also desirable.
```

../33–34/person/setPhone.cc

```
// Person member function: set phone number after verification

#include "person.ih"

void Person::setPhone(string const &phone)
{
  if (phone.empty())
    d_phone = " - not available -";
  else if (phone.find_first_not_of("0123456789") == string::npos)
    d_phone = phone;
  // Switched the two options above around from the example, as an empty string
  // will also not contain any non-numerical characters.
  else
    cout << "A phone number may only contain digits\n";
}
```

Exercise 34

../33–34/main.ih

```
// Main file: internal header

#include <iostream>
#include <string>
#include "person/person.h"

using namespace std;

void populateArray(Person array[], size_t sizeArray);
void printArray(Person array[], size_t sizeArray);
```

../33–34/main.cc

```
// Main file

#include "main.ih"

int main()
{
  Person personArray[5];  // Define an array of five Person objects

  size_t arraySize = sizeof(personArray) / sizeof(personArray[0]);  // Determine length array

  populateArray(personArray, arraySize);  // Populate array using input
  printArray(personArray, arraySize);     // Print array using object info
}
```

../33–34/populateArray.cc

```
// Function: populate array using user input

#include "person/person.ih"
using namespace std;

void populateArray(Person array[], size_t sizeArray)
{
  for (size_t index = 0; index != sizeArray; ++index) // Loop through array
  {
    cout << "? ";                 // Ask for user input
    array[index].extract(cin);   // Input that data into the extract function
  }
}
```

../33–34/printArray.cc

```
// Function: print object array

#include "person/person.ih"
using namespace std;

void printArray(Person array[], size_t sizeArray)
{
  for (size_t index = 0; index != sizeArray; ++index) // For each array element
    array[index].insert(cout);   // Output the object's info using insert
}
```