Object-Oriented Software Engineering hw1

- Author: 黃柏瑄 (P78081528)
- Source code (`hw1.cpp`):

```cpp
// g++ --std=c++1z -O2 -Wall -o hw1 hw1.cpp
// ./hw1
#include <iostream>
#include <memory>
#include <string>
#include <vector>

struct Date {
  int year;
  int month;
  int day;
};
std::ostream& operator<<(std::ostream& out, const Date date) {
  out << date.year << "/" << date.month << "/" << date.day;
  return out;
}

template <typename T, typename... Args>
std::shared_ptr<T> make_aggregate_shared(Args&&... args) {
  return std::make_shared<T>(T{std::forward<Args>(args)...});
}
struct Ticket {
  std::string buyer_name;
  int num_of_people;
  std::string bus_name;
  Date bus_departure_date;
};

class TicketProducerComsumer {
 public:
  TicketProducerComsumer(const std::string name) : name{name} {}
  std::string get_name() const { return name; }

 protected:
  std::string name;
  std::vector<std::shared_ptr<Ticket>> tickets;
};

class Person : public TicketProducerComsumer {
 public:
  Person(const std::string name) : TicketProducerComsumer{name} {}
  void buy_ticket(const std::shared_ptr<Ticket> ticket) {
    tickets.push_back(std::move(ticket));
  }
  void print_booked_buses() const {
    if (tickets.empty()) {
      std::cout << name << " does not book any ticket for bus.\n";
      return;
    }
    std::cout << name << " has booked: ";
    for (auto& t : tickets) {
      std::cout << "(" << t->bus_name << ", " << t->bus_departure_date << ") ";
    }
    std::cout << std::endl;
  }
};

class Bus : public TicketProducerComsumer {
 public:
  Bus(const std::string name, const Date date)
      : TicketProducerComsumer{name}, departure_date{date} {}
  Date get_departure_date() { return departure_date; }
  void sell_ticket(const std::shared_ptr<Ticket> ticket) {
    tickets.push_back(std::move(ticket));
  }
  void print_passengers() const {
    if (tickets.empty()) {
      std::cout << name << " does not have any passenger.\n";
      return;
    }
    std::cout << "The passengers of " << name << ": ";
    for (auto& t : tickets) {
      std::cout << "(" << t->buyer_name << ", " << t->num_of_people << ") ";
    }
    std::cout << std::endl;
  }

 private:
  Date departure_date;
};

class TicketMachine {
 public:
  static TicketMachine& get_ticket_machine() {
    static TicketMachine instance;
    return instance;
  }
  TicketMachine(const TicketMachine&) = delete;
  void operator=(const TicketMachine&) = delete;
  void book(Person* buyer, Bus* bus, const int num_of_people) const {
    auto ticket = make_aggregate_shared<Ticket>(buyer->get_name(),
                                                num_of_people, bus->get_name(),
                                                bus->get_departure_date());

    bus->sell_ticket(ticket);
    buyer->buy_ticket(ticket);
  }

 private:
  TicketMachine() {}
};

int main() {
  /* People */
  auto alice = std::make_unique<Person>("Alice");
  auto bob = std::make_unique<Person>("Bob");
  auto carol = std::make_unique<Person>("Carol");
  auto dave = std::make_unique<Person>("Dave");
  auto eve = std::make_unique<Person>("Eve");

  /* Bus */
  auto bus100 = std::make_unique<Bus>("Bus100", Date{2021, 2, 25});
  auto bus101 = std::make_unique<Bus>("Bus101", Date{2021, 2, 26});
  auto bus102 = std::make_unique<Bus>("Bus102", Date{2021, 2, 27});
  auto bus103 = std::make_unique<Bus>("Bus103", Date{2022, 2, 28});

  /* Book tickets */
  auto& tmachine = TicketMachine::get_ticket_machine();
  tmachine.book(alice.get(), bus100.get(), 4);
  tmachine.book(alice.get(), bus102.get(), 2);
  tmachine.book(bob.get(), bus100.get(), 6);
  tmachine.book(carol.get(), bus101.get(), 3);
  tmachine.book(dave.get(), bus100.get(), 5);

  /* Validation */
  bus100->print_passengers();
  alice->print_booked_buses();
  bus101->print_passengers();
  bob->print_booked_buses();
  bus103->print_passengers();
  eve->print_booked_buses();
  return 0;
}
```

- Executive result:

```
$ g++ --std=c++1z -O2 -Wall -o hw1 hw1.cpp
$ ./hw1
The passengers of Bus100: (Alice, 4) (Bob, 6) (Dave, 5)
Alice has booked: (Bus100, 2021/2/25) (Bus102, 2021/2/27)
The passengers of Bus101: (Carol, 3)
Bob has booked: (Bus100, 2021/2/25)
Bus103 does not have any passenger.
Eve does not book any ticket for bus.
```