자료구조 과제 1

20202863 신지환

R-1.5 Consider the expression y + 2 * z ++ < 3 - w / 5. Add parentheses to show the precise order of evaluation given the C++ rules for operator precedence.

((y + (2 * (z++))) < (3 - (w / 5)))

R-1.12 Modify the CreditCard class from Code Fragment 1.3 to check that the price argument passed to function chargeIt and the payment argument passed to function makePayment are positive.

```
#include "CreditCard.h"
using namespace std;

CreditCard::CreditCard(const string& no, const string& nm, int lim, double bal) {
   number = no;
   name = nm;
   balance = bal;
   limit = lim;
}

bool CreditCard::chargeIt(double price) {
   if (price + balance > double(limit) || price <= 0)
      return false;
   balance += price;
   return true;
}

void CreditCard::makePayment(double payment) {
   if(payment > 0)
      balance -= payment;
}


ostream& operator<<(ostream& out, const CreditCard& c) {
   out << "Number = " << c.getNumber() << "\n"
   << "Name = " << c.getName() << "\n"
   << "Balance = " << c.getBalance() << "\n"
   << "Limit = " << c.getLimit() << "\n";
   return out;
}
```

R-1.13 Modify the CreditCard class from Code Fragment 1.2 to charge interest on each payment

```

#ifndef CREDITCARD_H_
#define CREDITCARD_H_

#include <string>                // provides string
#include <iostream>              // provides ostream
```

```cpp
class CreditCard {
public:
  CreditCard(const std::string& no, // constructor
    const std::string& nm, int lim, double bal=0);
                                    // accessor functions
  std::string   getNumber() const   { return number; }
  std::string   getName() const     { return name; }
  double     getBalance() const  { return balance; }
  int       getLimit() const    { return limit; }
  double     getInterest() const { return interestRate;}
  // add these new modifier functions in the public section : Exercise R-1.15

  bool chargeIt(double price);      // make a charge
  void makePayment(double payment); // make a payment
private:                              // private member data
  std::string   number;             // credit card number
  std::string   name;               // card owner's name
  int         limit;              // credit limit
  double       balance;            // credit card balance
  double       interestRate;
};
                                    // print card information
std::ostream& operator<<(std::ostream& out, const CreditCard& c);

#endif /* CREDITCARD_H_ */
```

R-1.15 Modify the CreditCard class from Code Fragment 1.2 to include modifier functions that allow a user to modify internal variables in a CreditCard class in a controlled manner

```cpp
#ifndef CREDITCARD_H_
#define CREDITCARD_H_

#include <string>                // provides string
#include <iostream>              // provides ostream

class CreditCard {
public:
  CreditCard(const std::string& no, // constructor
    const std::string& nm, int lim, double bal=0);
                                    // accessor functions
  std::string   getNumber() const   { return number; }
  std::string   getName() const     { return name; }
  double     getBalance() const  { return balance; }
  int       getLimit() const    { return limit; }
  double     getInterest() const { return interestRate;}
  // add these new modifier functions in the public section : Exercise R-1.15
  void setBalance(double newBalance);     // 잔액 변경
  void setLimit(int newLimit);           // 한도 변경
  void setInterestRate(double newRate);   // 이자율 변경

    // 기존 기능
  bool chargeIt(double price);  // 결제
  void makePayment(double payment); // 대금 납부

  bool chargeIt(double price);      // make a charge
  void makePayment(double payment); // make a payment
private:                          // private member data
  std::string   number;           // credit card number
  std::string   name;             // card owner's name
  int          limit;             // credit limit
  double       balance;           // credit card balance
  double       interestRate;
};
                              // print card information
std::ostream& operator<<(std::ostream& out, const CreditCard& c);

#endif /* CREDITCARD_H_ */
```

R-1.16 Modify the declaration of the first for loop in the Test class in Code Fragment 1.4
so that its charges will eventually cause exactly one of the three credit cards to go over
its credit limit. Which credit card is it?

```
for (int j = 1; j <= 30; j++) {
    wallet[0]->chargeIt(double(j));     // 카드 1: 그대로
    wallet[1]->chargeIt(2.5 * j);       // 카드 2: 더 많은 금액을 청구
    wallet[2]->chargeIt(double(3 * j));  // 카드 3: 그대로
}
```

C-1.9 Write a C++ class Vector2, that stores the (x,y) coordinates of a two-dimensional
vector, where x and y are of type double. Show how to override various C++ operators in
order to implement the addition of two vectors (producing a vector result), the
multiplication of a scalar times a vector (producing a vector result), and the dot product
of two vectors (producing a double result)

```
#include <iostream>

class Vector2 {
private:
    double x, y

public:
    // 생성자
    Vector2(double x = 0, double y = 0) : x(x), y(y) {}

    // 벡터 덧셈
```

```cpp
    Vector2 operator+(const Vector2& other) const {
        return Vector2(x + other.x, y + other.y);
    }

    // 스칼라 곱
    Vector2 operator*(double scalar) const {
        return Vector2(x * scalar, y * scalar);
    }

    // 벡터 내적
    double operator*(const Vector2& other) const {
        return (x * other.x) + (y * other.y);
    }

    // 출력 연산자 오버로딩
    friend std::ostream& operator<<(std::ostream& os, const Vector2& v) {
        os << "(" << v.x << ", " << v.y << ")";
        return os;
    }
};
```