== Laboratório 7 ==

1. **Escreva uma classe `Carrinho` para armazenar itens a serem comprados. Cada item é um objeto da classe `Produto` que possui os atributos privados tipo (p. ex., "brinquedo", "eletrodoméstico" etc.), marca e preço. Os produtos são armazenados em um vector de ponteiros para objetos da classe `Produto`. A classe `Carrinho` deve oferecer um método para inserir produtos e outro para remover produtos usando os operadores + e - sobrecarregados, respectivamente. Dessa forma, a inserção deve ser feita da seguinte maneira:**

   ```
   carrinho = carrinho + objetoBrinquedo;
   ```

   e a remoção:

   ```
   carrinho = carrinho - objetoBrinquedo;.
   ```

   Note que o método de inserção deve inserir o produto no vector, enquanto o de remoção deve tirar um produto específico do mesmo vector. A classe `Carrinho` ainda deve implementar um construtor que define o número máximo de itens no carrinho (argumento passado para o construtor do vector privado), um construtor de cópia e o operador << sobrecarregado para impressão na tela dos produtos atuais no carrinho.

   A classe `Produto` deve oferecer um método para inicialização dos seus atributos (pode ser o próprio construtor) e métodos do tipo "get" para os atributos.

   Faça uma função principal que contemple todos os métodos da classe `Carrinho`.

2. **Dada a função principal abaixo, implemente as classes `Jogo` e `Personagem` e as funções globais necessárias em arquivos *.cpp e *.h para que o programa possa ser compilado e executado no laboratório. A classe `Jogo` tem um vector de objetos da classe `Personagem` (`vector <Personagem> v`).**

```
/************************************************************/
// includes...
using namespace std;

int main () {
      /* Construtor da classe Jogo possui argumento que define o
      número máximo de personagens */
      Jogo jogo (4);

      /* Construtor da classe Personagem possui argumentos nome,
      nível de força e nível de inteligência */
      Personagem hulk ("Hulk", 90, 20);
      Personagem homemDeFerro ("Homem de Ferro", 60, 90);
      Personagem capitao ("Capitao America", 50, 70);
      Personagem thor ("Thor", 80, 60);

      /* Operador () sobrecarregado adiciona os personagens ao
      jogo de forma cascateada */
      jogo(hulk)(homemDeFerro)(capitao)(thor);
```

```
        /* Operador [] sobrecarregado retorna ponteiro para objeto
        da classe Personagem a partir do atributo nome do objeto e
        operador << sobrecarregado imprime todas as características
        do personagem na tela */
        if (jogo ["Hulk"]) {
              cout << jogo ["Hulk"];
        } else { cout << "Personagem nao encontrado!" << endl; }

        if (jogo ["Homem Formiga"]) {
              cout << jogo ["Homem Formiga"];
        } else { cout << "Personagem nao encontrado!" << endl; }

        /* Operador [] sobrecarregado retorna ponteiro para objeto
        da classe Personagem a partir de índice e operador <<
        sobrecarregado imprime todas as características do
        personagem na tela */
        for (unsigned i = 0; i < jogo.getNumeroPersonagens (); i++)
        {
              cout << jogo [i];
        }

        /* Função global calculaEstatistica retorna ponteiro para
        Personagem que possui maior força ou maior inteligência,
        conforme o ponteiro para função passada como segundo
        argumento */
        cout << "*** Mais Forte:\n"
             << calculaEstatistica (jogo, maisforte);
        cout << "*** Mais Inteligente:\n"
             << calculaEstatistica (jogo, maisinteligente);

        return 0;
}
/*************************************************************/
```

== Respostas ==

1.

```
/*****************************************************************************/
/*************************** Programa Principal ****************************/

#include <iostream>
#include <string>
#include <iomanip>
#include <vector>

#include "produto.h"
#include "carrinho.h"

using namespace std;



int main () {
    Carrinho car (5);

    Produto brinquedo ("brinquedo", "estrela", 90.00);
    Produto arroz ("arroz", "Tio Joao", 20.00);
    Produto pneu ("pneu", "Goodyear", 150.00);

    car = car + brinquedo;
    car = car + arroz;
    car = car + pneu;

    cout << "*** Completo\n" << car;

    car = car - arroz;

    cout << "*** Reduzido\n" << car;

    Carrinho car2 (car);

    cout << "*** Copia\n" << car2;

    return 0;
}

/*****************************************************************************/
/****************************** Arquivo produto.h ****************************/

#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

#ifndef PRODUTO_H
#define PRODUTO_H

class Produto {

    public:
            Produto (string, string, double);
            string getTipo ();
            string getMarca ();
            double getPreco ();
            void setPreco (double);

    private:
            string tipo, marca;
            double preco;
};

ostream &operator<< (ostream &, Produto *);

#endif

/*****************************************************************************/
```

```
/***************************** Arquivo produto.cpp *************************/

#include "produto.h"

ostream &operator<< (ostream &out, Produto *p) {
    out << setw(20) << "Tipo: " << setw(20) << p->getTipo () << endl;
    out << setw(20) << "Marca: " << setw(20) << p->getMarca () << endl;
    out << setw(20) << "Preco (R$): " << setw(20) << fixed << setprecision (2)
        << p->getPreco () << endl;

    return out;
}

Produto::Produto (string t, string m, double p): tipo (t), marca (m), preco (p) {}
string Produto::getTipo () { return tipo; }
string Produto::getMarca () { return marca; }
double Produto::getPreco () { return preco; }
void Produto::setPreco (double p) { preco = p; }

/*************************************************************************/
/***************************** Arquivo carrinho.h *************************/

#include <iostream>
#include <vector>
#include <iomanip>

#include "produto.h"

using namespace std;

#ifndef CARRINHO_H
#define CARRINHO_H

class Carrinho {
    friend ostream &operator<< (ostream &, Carrinho &);

    public:
    Carrinho (int);
    Carrinho (const Carrinho &);
    ~Carrinho ();

    Carrinho &operator+ (Produto &);
    Carrinho &operator- (Produto &);

    private:
           unsigned conta, maxNum;
           vector <Produto *> v;
};

#endif

/*************************************************************************/
/***************************** Arquivo carrinho.cpp *************************/

#include "carrinho.h"

ostream &operator<< (ostream &out, Carrinho &c) {
    for (unsigned i = 0; i < c.conta; i++)
                out << c.v.at (i);
    return out;
}

Carrinho::Carrinho (int n): conta (0), maxNum (n), v (n) {}
Carrinho::Carrinho (const Carrinho &c): conta (c.conta),
                        maxNum (c.maxNum), v (c.maxNum) {
    for (unsigned i = 0; i < conta; i++) {
           v.at (i) = new Produto (c.v.at (i)->getTipo (),
                                   c.v.at (i)->getMarca (), c.v.at (i)->getPreco ());
    }
}
Carrinho::~Carrinho () {
    cout << "Destruindo..." << endl;
    for (unsigned i = 0; i < conta; i++) delete v.at (i);
}

Carrinho &Carrinho::operator+ (Produto &p) {
    if (conta < maxNum - 1)
```

```
            v.at (conta++) = new Produto (p.getTipo (),
                                          p.getMarca (), p.getPreco ());
    else cout << "Carrinho cheio..." << endl;

    return *this;
}
Carrinho &Carrinho::operator- (Produto &p) {
    for (unsigned i = 0; i < conta; i++) {
            if (v.at (i)->getTipo () == p.getTipo ()) {
                    delete v.at (i);
                    v.erase (v.begin () + i);
                    conta--;
            }
    }
    return *this;
}
/*****************************************************************************/
```

2.

```
/*****************************************************************************/
/************************** Programa Principal ***************************/

#include <iostream>
#include <string>
#include <vector>
#include <iomanip>

#include "personagem.h"
#include "jogo.h"
#include "globais.h"

using namespace std;

int main () {
    /* Construtor da classe Jogo possui argumento que define o número de
    personagens */
    Jogo jogo (4);

    /* Construtor da classe Personagem possui argumentos nome, nível de força e
    nível de inteligência */
    Personagem hulk ("Hulk", 90, 20);
    Personagem homemDeFerro ("Homem de Ferro", 60, 90);
    Personagem capitao ("Capitao America", 50, 70);
    Personagem thor ("Thor", 80, 60);

    // Adiciona os personagens ao jogo
    jogo(hulk)(homemDeFerro)(capitao)(thor);

    /* Imprime todas as características do personagem na tela a partir do atributo
    nome */
    if (jogo ["Hulk"]) {
            cout << jogo ["Hulk"];
    } else {
            cout << "Personagem nao encontrado!" << endl;
    }

    if (jogo ["Homem Formiga"]) {
            cout << jogo ["Homem Formiga"];
    } else {
            cout << "Personagem nao encontrado!" << endl;
    }

    /* Imprime as características de todos os personagens na tela */
    for (unsigned i = 0; i < jogo.getNumeroPersonagens (); i++) {
            cout << jogo [i];
    }

    /*  Função global que retorna o objeto que possui maior força ou maior
    inteligência, conforme o ponteiro para função passada como segundo argumento */
    cout << "*** Mais Forte:\n" << calculaEstatistica (jogo, maisforte);
    cout << "*** Mais Inteligente:\n" << calculaEstatistica (jogo, maisinteligente);

    return 0;
}
```

```
/*****************************************************************************/
/****************************** Arquivo jogo.h *******************************/

#include <iostream>
#include <string>
#include <vector>

#include "personagem.h"

#ifndef JOGO_H
#define JOGO_H

class Jogo {
      public:
             Jogo (int);

             Jogo &operator()(Personagem &);

         unsigned getNumeroPersonagens ();

             Personagem *operator[](string s);

         Personagem *operator[](unsigned);

      private:
              vector <Personagem> v;
};

#endif

/*****************************************************************************/
/****************************** Arquivo jogo.cpp *****************************/

#include "jogo.h"

Jogo::Jogo (int n): v (n) {}

Jogo &Jogo::operator()(Personagem &p) {
     static int number = 0;
     v.at (number++) = p;
     return *this;
}
unsigned Jogo::getNumeroPersonagens () { return v.size (); }

Personagem *Jogo::operator[](string s) {
    for (unsigned i = 0; i < v.size (); i++) {
        if (!v.at (i).getNome ().compare (s))
             return &v.at (i);
        }
    return NULL;
}

Personagem *Jogo::operator[](unsigned i) {
    if ((i < 0) || (i >= v.size ()))
    return NULL;
    else return &v.at (i);
}

/*****************************************************************************/
/***************************** Arquivo personagem.h **************************/

#include <iostream>
#include <string>
#include <iomanip>

using namespace std;

#ifndef PERSONAGEM_H
#define PERSONAGEM_H

class Personagem {
      friend ostream &operator<<(ostream &, Personagem *);

    public:
         Personagem () {}
             Personagem (string, int, int);
```

```
            string getNome ();
            int getForca ();
            int getInteligencia ();

      private:
             string nome;
             int forca, inteligencia;
};

#endif

/*****************************************************************************/
/***************************** Arquivo personagem.cpp ************************/

#include "personagem.h"

ostream &operator<<(ostream &out, Personagem *p) {
   out << setw (20) << "Nome: " << setw (10) << p->nome << endl;
   out << setw (20) << "Forca: " << setw (10) << p->forca << endl;
   out << setw (20) << "Inteligencia: " << setw (10) << p->inteligencia << endl;

   return out;
}

Personagem::Personagem (string s, int f, int i):
            nome (s), forca (f), inteligencia (i) {}

string Personagem::getNome () { return nome; }
int Personagem::getForca () { return forca; }
int Personagem::getInteligencia () { return inteligencia; }

/*****************************************************************************/
/***************************** Arquivo globais.h *****************************/

#include <iostream>

#include "personagem.h"
#include "jogo.h"

using namespace std;

#ifndef GLOBAIS_H
#define GLOBAIS_H

Personagem *maisforte (Jogo &jogo) {
    Personagem *maisForte;
    int maiorForca = 0;
    for (unsigned i = 0; i < jogo.getNumeroPersonagens (); i++) {
           if (jogo [i]->getForca () > maiorForca) {
                  maisForte = jogo [i];
                  maiorForca = jogo [i]->getForca ();
           }
    }

    return maisForte;
}

Personagem *maisinteligente (Jogo &jogo) {
    Personagem *maisInteligente;
    int maiorInteligencia = 0;
    for (unsigned i = 0; i < jogo.getNumeroPersonagens (); i++) {
           if (jogo [i]->getInteligencia () > maiorInteligencia) {
                  maisInteligente = jogo [i];
                  maiorInteligencia = jogo [i]->getInteligencia ();
           }
    }

    return maisInteligente;
}

Personagem *calculaEstatistica (Jogo &jogo, Personagem * (*p) (Jogo &)) {
    return (*p) (jogo);
}

#endif

/*****************************************************************************/
```