```
#include <iostream>
#include <string>
#include <vector>
#include "Set.h"
#include "Data.h"
#include "Jogador.h"
namespace std {
class Jogo
{
private:
    Data data;
    int hora;
    string local;
    string adversario;
    string auxiliar;
    vector<Set> vSet;
    vector<Set>::iterator itSet;
    vector<Jogador>::iterator itJogador;
public:
    Jogo();
    virtual ~Jogo();
    string getAdversario() const;
    string getAuxiliar() const;
    Data getData() const;
    int getHora() const;
    string getLocal() const;
    void setAdversario(string adversario);
    void setAuxiliar(string auxiliar);
    void setData(Data data);
    void setHora(int hora);
    void setLocal(string local);
    void realizaSet(vector<Jogador>::iterator it, int tamanho);
};
}
-----------------------------------------------
#include <iostream>
#include <string>
#include <vector>
#include <cstdlib>
#include <list>
#include <time.h>
#include "Ponto.h"
namespace std {
class Set {
private:
        vector<Ponto> vPtsCasa;
        vector<Ponto> vPtsAdversario;
        vector<Ponto>::iterator itPtsCasa;
        vector<Ponto>::iterator itPtsAdversario;
        vector<Jogador>::iterator itJogador;
        int tamanho;
public:
        Set();
        virtual ~Set();
        void obterMaioresPontuadores();
        void listaPontoPorTipo();
    vector<Ponto>::iterator getItPtsAdversario() const;
    vector<Ponto>::iterator getItPtsCasa() const;
    vector<Ponto> getPtsAdversario() const;
    vector<Ponto> getPtsCasa() const;
    void setItPtsAdversario(vector<Ponto>::iterator itPtsAdversario);
    void setItPtsCasa(vector<Ponto>::iterator itPtsCasa);
    void setPtsAdversario(vector<Ponto> vPtsAdversario);
    void setPtsCasa(vector<Ponto> vPtsCasa);
};
}
-------------------------------------------------------------
void Set::obterMaioresPontuadores()
{
   list<string> nomes;
   list<string>::iterator itNomes;
   int tot;
   cout << "Da parte dos adversários" << endl;
   for (itPtsAdversario = vPtsAdversario.begin(); itPtsAdversario !=
vPtsAdversario.end(); ++itPtsAdversario)
   {
        cout << itPtsAdversario->getItJogador()->getNome() << endl;
```

```
        cout << itPtsAdversario->getTipoPonto() << endl;
    // Poderia ser criada alguma solução para verificar o número de vezes que o nome
aparece...
        nomes.push_back(itPtsCasa->getItJogador()->getNome());
     }
   nomes.sort();
   tot = 1;
// E verificar quantas vezes o nome se repete...
        //...
   nomes.clear();
   cout << "Da parte dos da casa" << endl;
   for (itPtsCasa = vPtsCasa.begin(); itPtsCasa != vPtsCasa.end(); ++itPtsCasa)
   {
       cout << itPtsCasa->getItJogador()->getNome() << endl;
       cout << itPtsCasa->getTipoPonto() << endl;
       nomes.push_back(itPtsCasa->getItJogador()->getNome());
   }
   nomes.sort();
   tot = 1;
// idem...
  }
-----------------------------------------------------------------
  void Set::listaPontoPorTipo()
  {
        int s_ataque = 0, s_bloqueio = 0, s_contraAtaque = 0, s_saque = 0,
s_erroAdversario = 0;
        int s_individual = 0, s_duplo = 0, s_triplo = 0;
        TipoPonto::TP pt;
        TipoBloqueio::TB bl;
        cout << "Tipo de pontos para adversários:" << endl;
        for (itPtsAdversario = vPtsAdversario.begin(); itPtsAdversario !=
vPtsAdversario.end(); ++itPtsAdversario)
        {
                pt = this->itPtsAdversario->getTipoPonto();
                switch(pt)
                {
                        case TipoPonto::ataque: ++s_ataque;break;
                        case TipoPonto::bloqueio: ++s_bloqueio;break;
                        case TipoPonto::contraAtaque: ++s_contraAtaque;break;
                        case TipoPonto::saque: ++s_saque;break;
                        case TipoPonto::erroAdversario: ++s_erroAdversario; break;
                }
                bl = this->itPtsAdversario->getTipoBloqueio();
                switch(bl)
                {
                        case TipoBloqueio::individual: ++s_individual;break;
                        case TipoBloqueio::duplo: ++s_duplo;break;
                       case TipoBloqueio::triplo: ++s_triplo;break;
                 }

                cout << "Ataques:" << s_ataque << endl;
                cout << "Bloqueios:" << s_bloqueio << endl;
                cout << "contra ataques:" << s_contraAtaque << endl;
                cout << "Saques:" << s_saque << endl;
                cout << "erros:" << s_erroAdversario << endl;
                cout << "individual:" << s_individual << endl;
                cout << "Duplo:" << s_duplo << endl;
                cout << "Triplo:" << s_triplo << endl;
        }

                cout << "*********************************" << endl;

                s_ataque = 0, s_bloqueio = 0, s_contraAtaque = 0, s_saque = 0,
s_erroAdversario = 0;
                s_individual = 0, s_duplo = 0, s_triplo = 0;

                cout << "Tipo de pontos para a casa:" << endl;
                for (itPtsCasa = vPtsCasa.begin(); itPtsCasa != vPtsCasa.end();
++itPtsCasa)
                {
                        pt = this->itPtsCasa->getTipoPonto();
                        switch(pt)
                        {
                            case TipoPonto::ataque: ++s_ataque;break;
                            case TipoPonto::bloqueio: ++s_bloqueio;break;
                            case TipoPonto::contraAtaque: ++s_contraAtaque;break;
                            case TipoPonto::saque: ++s_saque;break;
```

```
                    case TipoPonto::erroAdversario: ++s_erroAdversario; break;
                }
                bl = this->itPtsCasa->getTipoBloqueio();
                switch(bl)
                {
                   case TipoBloqueio::individual: ++s_individual;break;
                   case TipoBloqueio::duplo: ++s_duplo;break;
                   case TipoBloqueio::triplo: ++s_triplo;break;
                 }

                 cout << "Ataques:" << s_ataque << endl;
                 cout << "Bloqueios:" << s_bloqueio << endl;
                 cout << "contra ataques:" << s_contraAtaque << endl;
                 cout << "Saques:" << s_saque << endl;
                 cout << "erros:" << s_erroAdversario << endl;
                 cout << "individual:" << s_individual << endl;
                 cout << "Duplo:" << s_duplo << endl;
                 cout << "Triplo:" << s_triplo << endl;

                 cout << "********************************" << endl;
            }
}
```