



Universidade Federal do Pará
Instituto de Ciências Exatas e Naturais
Faculdade de Computação
Ciência da Computação

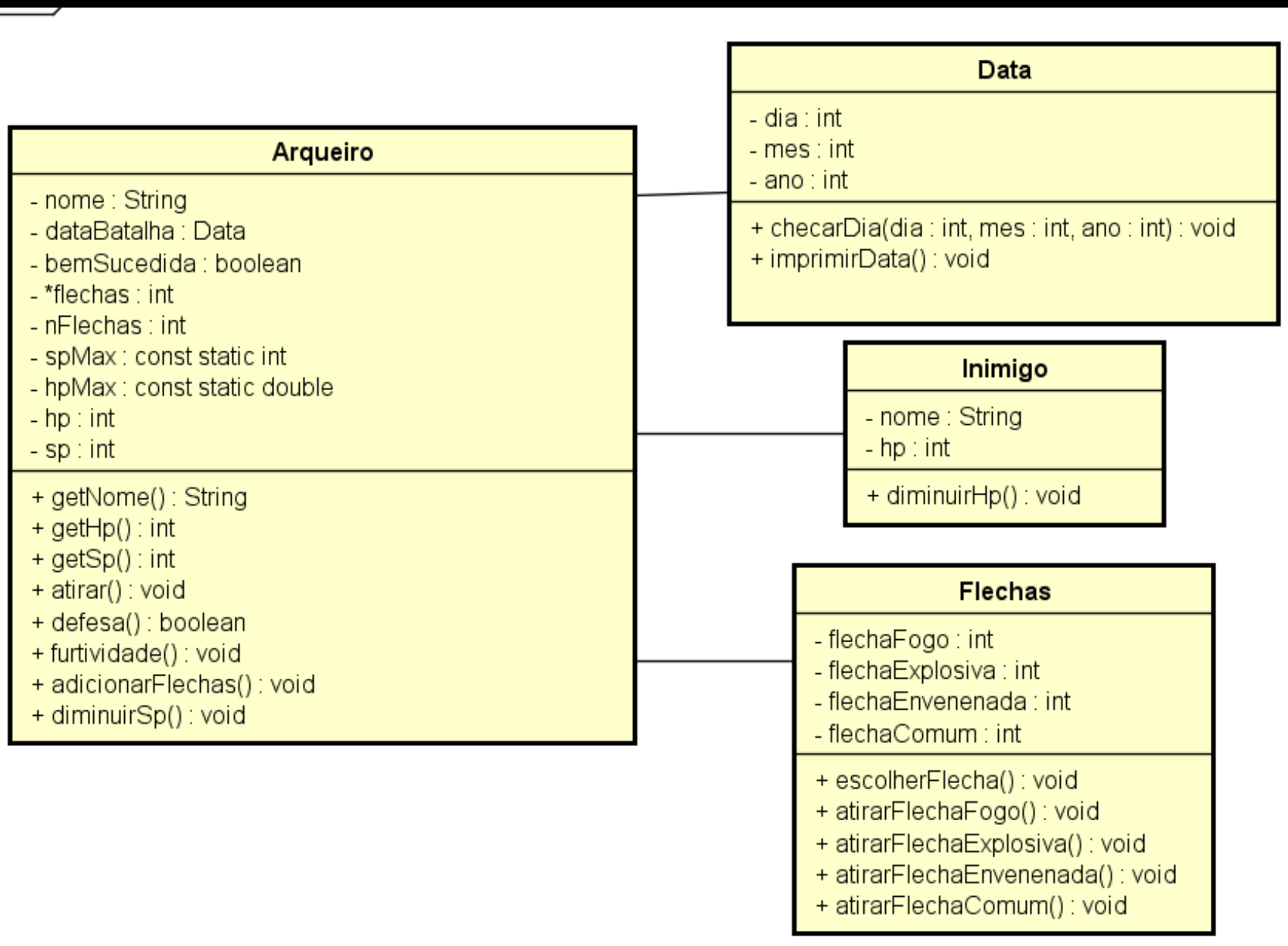
Classe em C++: Arqueiro

Adicionar à classe em questão um **atributo const static**, um **método static const**, compor a **classe Data.h** à classe em questão, **compor uma terceira classe**, à livre criatividade de implementação, também à classe em questão, além de **implementar alocação dinâmica**.



<https://github.com/anaisabelamr/Arqueiro>

UML



Sumário

- | | |
|--|----------|
| 1. Atributo Const Static | Slide 6 |
| 2. Método Const Static | Slide 8 |
| 3. Classe Data.h relacionada à classe Arqueiro | Slide 11 |
| 4. Segunda classe relacionada à classe Arqueiro | Slide 12 |
| 5. Terceira classe relacionada à classe Arqueiro | Slide 15 |
| 6. Alocação dinâmica | Slide 17 |
| 7. Sobrecarga de operadores | Slide 19 |

Atributo Const Static

Arqueiro.h

```
private:
→ string nome;
→ Data dataBatalha;
→ bool bemSucedida;
→ int *flechas;
→ int nFlechas;
→ const static int spMax; ←
→ const static double hpMax; ←
→ int hp;
→ int sp;
};
#endif // ARQUEIRO_H
```

Atributo Const Static

```
#include "Arqueiro.h"  
#include "Inimigo.h"  
#include <string>  
#include <iostream>  
#include <stdlib.h>  
#include <stdio.h>  
#include <windows.h>  
#include "Flechas.h"
```

```
using std::cout;  
using std::cin;
```

```
const static int spMax = 20; ←
```

```
const static double hpMax = 50; ←
```

Arqueiro.cpp

Método Const Static

Arqueiro.h

```
public:
    // ...

    const Arqueiro &operator=(const Arqueiro &);
    bool operator==(const Arqueiro &) const;

    Arqueiro();
    Arqueiro(int hp, int sp, const string nome, Data &);
    ~Arqueiro();
    void setNome(const string);
    string getNome();
    void setHp(int );
    int getHp();
    void setSp(int );
    int getSp();
    → static const void dadosArqueiro(); ←
    void atirar(Flechas &, Inimigo &);
    bool defesa(bool);
    void furtividade();
    void adicionarFlechas(const int &);
    → void diminuirSp();
```


Método Const Static

Arqueiro.cpp

```
const void Arqueiro::dadosArqueiro()
{
    → Arqueiro a;
    → cout << "Nome do Arqueiro: " << a.getNome() << endl;
    → cout << "\n\nHP: " << a.getHp() << "\nSP: " << a.getSp();
}
```

Método Const Static

main.cpp


```
→ i.setNome("Malcom Merlin\n");  
→ i.setHp(hpMax);  
→ arq.dadosArqueiro(); ←  
  
→ cout << "\n\nInimigo: " << i.getNome() << endl;  
→ cout << "\nHP: " << i.getHp() << endl;  
→
```

Classe Data.h relacionada à classe Arqueiro

main.cpp

```
Arqueiro arq;  
> Inimigo i;  
> Data d(15,6,2015); ←  
> Flechas f;  
  
> arq.setNome("Oliver Queen\n");  
> arq.setHp(hpMax);  
> arq.setSp(spMax);  
  
> i.setNome("Malcom Merlin\n");  
> i.setHp(hpMax);  
  
> d.imprimirData(); ←  
  
> arq.dadosArqueiro();
```

Arqueiro.h



```
Arqueiro();  
Arqueiro(int hp, int sp, const string nome, Data &);  
~Arqueiro();  
void setNome(const string);  
string getNome();  
void setHp(int );
```

Segunda classe relacionada à classe Arqueiro (Inimigo)

```
class Inimigo
{
public:
    → Inimigo();
    → Inimigo(int hp, const string nome);
    → ~Inimigo();
    → void setNome(const string);
    → string getNome();
    → void setHp(int );
    → int getHp();
    → void diminuirHp();
    → void inimigoMorto();
private:
    → string nome;
    → int hp;
};
```

Inimigo.h

Segunda classe relacionada à classe Arqueiro (Inimigo)

Inimigo.cpp

```
Inimigo::Inimigo()
{
    this->nome = "";
    hp = 0;
}

Inimigo::Inimigo(int hp, const string nome)
{
    this->hp = hp;
    this->nome = nome;
}

Inimigo::~Inimigo()
{
}

void Inimigo::setNome(string nome)
{
    this->nome = nome;
}

string Inimigo::getNome()
{
    return nome;
}
```

```
void Inimigo::setHp(int hp)
{
    this->hp = hp;
}

int Inimigo::getHp()
{
    return hp;
}

void Inimigo::diminuirHp()
{
    hp = 0.2 * hp;
}

void Inimigo::inimigoMorto()
{
    hp = 0;
    cout << "\n\nInimigo esta derrotado. Parabens, arqueiro!";
}
```

Segunda classe relacionada à classe Arqueiro (Inimigo)

```
Arqueiro();
Arqueiro(int hp, int sp, const string nome, Data &);
~Arqueiro();
void setNome(const string);
string getNome();
void setHp(int );
int getHp();
void setSp(int );
int getSp();
→ static const void dadosArqueiro();
void atirar(Flechas &, Inimigo &); ←
bool defesa(bool);
void furtividade();
void adicionarFlechas(const int &);
→ void diminuirSp();
```

Terceira classe relacionada à classe Arqueiro (Flechas)

```
class Flechas
{
public:
    Flechas();
    Flechas(int, int, int, int);
    ~Flechas();
    void escolherFlecha();
    void atirarFlechaFogo(Inimigo &);
    void atirarFlechaExplosiva(Inimigo &);
    void atirarFlechaEnvenenada(Inimigo &);
    void atirarFlechaComum(Inimigo &);

private:
    int flechaFogo = 10;
    int flechaExplosiva = 5;
    int flechaEnvenenada = 5;
    int flechaComum = 20;

};
```

Flechas.h

Terceira classe relacionada à classe Arqueiro (Flechas)

Arqueiro.cpp

```
Arqueiro();
Arqueiro(int hp, int sp, const string nome, Data &);
~Arqueiro();
void setNome(const string);
string getNome();
void setHp(int );
int getHp();
void setSp(int );
int getSp();
→ static const void dadosArqueiro();
void atirar(Flechas &, Inimigo &); ←
bool defesa(bool);
void furtividade();
void adicionarFlechas(const int &);
→ void diminuirSp();
.
```


Alocação dinâmica

```
void adicionarFlechas(int nFlechas, int *flechas, const int &novasFlechas)
{
    if (nFlechas != 0)
    {
        int * aux = new int[nFlechas];

        for (int i = 0; i < nFlechas; i++)
            aux[i] = flechas[i];

        delete [] flechas;
        → →
        → → nFlechas++;

        flechas = new int [ nFlechas ] ;
    }
}
```

Arqueiro.cpp

Alocação dinâmica

Arqueiro.cpp

```
        for (int i = 0; i < nFlechas-1; i++)
            flechas[i] = aux [i];
    }

    flechas[nFlechas-1] = novasFlechas;
    delete [] aux;
}
else
{
    flechas = new int [ ++nFlechas];
    flechas[0] = novasFlechas;
}
```

Sobrecarga de operadores

Arqueiro.h

```
class Arqueiro
{
    ....
    friend ostream &operator<<(ostream &, const Arqueiro &);
    ....
public:
    ....
    const Arqueiro &operator=(const Arqueiro &);
    bool operator==(const Arqueiro &) const;
```

Sobrecarga de operadores

Arqueiro.cpp

```
ostream &operator<<(ostream &output, const Arqueiro &imprime)
{
    output << "NOME DO ARQUEIRO: " << imprime.nome << "\nHP: " << imprime.hp << "\nSP: " << imprime.sp;
    return output;
}

bool Arqueiro::operator ==(const Arqueiro &comparaArqueiro) const
{
    if(comparaArqueiro.nome != nome) return false;
    if(comparaArqueiro.hp != hp) return false;
    if(comparaArqueiro.sp != sp) return false;
    return true;
}

const Arqueiro & Arqueiro::operator =(const Arqueiro &atributo)
{
    nome = atributo.nome;
    hp = atributo.hp;
    sp = atributo.sp;
}
```