**UNIVERSIDADE FEDERAL DO CEARÁ**
**Campus de Quixadá**
Prof. Thiago Werlley
QXD0007- Programação Orientada Objeto

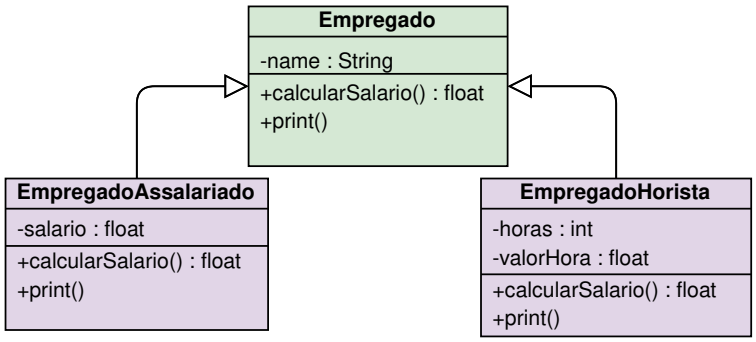
**AP1**
**2022.2**

## Instruções

- Todas as questões que envolvem programas devem ser resolvidos por programas em linguagem C++;
- Na solução dos exercícios, devem ser utilizados os conceitos aprendidos em aulas;
- Eventuais dúvidas podem ser sanadas com o professor.
- Lembrando de criar a interface e realizar o encapsulamento de todas as classes, criando os construtores e o drive de teste

1. Dada a hierarquia estabelecida e a necessidade de polimorfismo

   - O método **print** será um método virtual
   - O método **calcularSalario** será um método virtual puro
   - As classes derivadas definem seus próprios atributos e respectivos getters e setters.




**SAÍDA DO SISTEMA:**

- **Empregado**
  - **calcularSalario** = 0
  - **print**: nome
- **EmpregadoAssalariado**
  - **calcularSalario** = salario
  - **print**: nome e salario
- **EmpregadoHorista**
  - **calcularSalario**
    - if (horas <= 40) {horas*valorHora}
    - if (horas > 40) {(40/valorHora)+(horas-40)*valorHora*1.5}
  - **print**: nome, horas e valorHora

2. Ainda no exercício anterior deve-se:

**CRIE UMA CLASSE DivideByExceprion.h**

```
class DivideByZeroException : public runtime_error
{
public:
  // construtor especifica a mensagem de erro padrão
  DivideByZeroException(): runtime_error( "tentou dividir por zero" )
  {}
};
```

---

**DENTRO DA main**

```
double quotient( int numerator, int denominator )
{
  // lança DivideByZeroException se tentar dividir por zero
  if ( denominator == 0 )
    throw DivideByZeroException(); // termina a função

  // retorna resultado da divisão
  return static_cast< double >( numerator ) / denominator;
}
```

- A biblioteca utilizada é: **#include<stdexcept>**
- **try{...}**
  - quotient
  - Deve implementar a mensagem: *O valor do salario é: (coloque aqui o cálculo)*
- **catch(DivideByZeroException &divideByZeroException){...}**
  - Deve implementar a mensagem: *Uma exceção ocorreu: (coloque aqui a exceção)*

3. Explique os conceitos de *cast* e *downcasting* em C++.