***Importante***: Para cada item abaixo deve ser copiado trechos do código que cumprem o requisito e explicado, se não for aparente, o porquê o requisito é cumprido. Sejam bem explícitos. Deve ser indicado também o arquivo da classe em que está o trecho do código. Eu avaliarei o código do Github a partir desse documento para confirmá-lo e também para detectar possíveis erros.**Quem não seguir o que está indicado aqui, não terá o projeto avaliado e perderá a atividade.**

**Criar repositórios desde o início do desenvolvimento. Primeiro commit deve estar com todas as classes declaradas sem implementação.**

**Primeira versão do código completa dia 01 de dezembro.**

**Versão final do código dia 08 de dezembro.**

**No dia da apresentação deve ser mostrado o código rodando para vários casos de teste.**

**Será avaliado o quanto o projeto está elaborado. Trabalhos muito simples com métodos apenas mostrando mensagens, serão penalizados.**

**Requisitos de implementação para C++**

1. Todas as classes concretas devem vir de classes abstratas. Pelo menos três hierarquias de classes. Uma das hierarquias deve ter três níveis. Exemplo: Personagem (abstract) >>Ciborgue (Abstract) >>Robocop; Class Arma (Abstract) >> Beretta93R
2. Em todas as classes: **construtor de cópia**, **operatores<< e +=,** e **construtor default**. Fazer o máximo de reaproveitamento de código usando **static\_cast**
3. Todas as hierarquias devem ter classes Concretas, e em uma das hierarquias, três classes Concretas relacionadas: Exemplo Servico>>ServicoStream>> (Ne tflix, HBOStream, AmazonPrime, NowTv). Em uma pesquisa de 10 segundos: http://www.tomsguide.com/us/pictures-story/620-top-online-streaming-video.html
4. Atributos static e conststatic em todas as hierarquias de classe
5. Método static em todas as hierarquias de classe
6. Construtores em todas as classes, e três para todas as classes da hierarquia principal. Sempre validar os dados em todas as classes
7. Vector em todas em todas as hierarquias de classe
8. ENUM na hierarquia principal
9. Usar o **dynamic cast** e **typeid** no main junto com as classes concretas. Para uma da classe concreta identificada, chamar um método dessa classe e fazer uma ação;
10. Usar o rand. Nota: deve ser usado conforme o contexto do projeto. Se for usado em um método genérico sem relação com a classe e apenas para cumpri-lo, esse requisito será desconsiderado.http://en.cppreference.com/w/cpp/numeric/random/rand
11. No main o usuário deve fazer entrada via teclado e interagir com a aplicação