***­Importante***: Para 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.**

**Requisitos de implementação**

***Todos os atributos e funções membros devem estar relacionados a classe***

1. Pelo menos 4 atributos ok
   1. Livros:
      1. private float preco;
      2. private String nome;
      3. private Genero tiposdelivros ;
      4. private int numerodepaginas;
      5. private int numerodeexemplares;
   2. Filmes:
      1. private float preco;
      2. private String nome;
      3. private Genero tiposdeflme;
      4. private float duracao;
   3. Usuario:
      1. private String nome;
      2. private String endereco;
      3. private String Loguin,senha;
   4. SistemaDeLogin:
      1. private ArrayList<Usuario> usuarios;
      2. private ArrayList<Amazon> amazon;
      3. private static int numerodecadastros;
   5. LojaOnline
      1. protected Tipodecompra formadepagamento;
      2. protected static int quantidadedelivros = 0;
      3. protected int livrosccomprados;
      4. protected ArrayList<Livros> livros;
   6. InstantVideo:
      1. protected Tipodenegocio negocio;
      2. protected static ArrayList<Filmes> filmes;
      3. protected ArrayList<Filmes> filmesalugados;
      4. protected ArrayList<Filmes> filmescomprados;
   7. Amazon:
      1. protected final static String CNPJ = "15.436.940/0001-03";
      2. protected Nota satisfacaodocliente;
   8. Obs: Nas classes Amazon e SistemaDeLogin não encontrei necessidade de adicionar mais variáveis.
2. Pelo menos 4 funções membros sem incluir get e set
   1. Amazon:
      1. public void menu();
      2. public void darNota();
   2. InstantVideo:
      1. private boolean verificar(Filmes filme);
      2. private void mostrar();
      3. private void add();
      4. public void alugar();
      5. public void comprar();
      6. public void comprar(int i);
      7. public void menu();
   3. LojaOnline:
      1. public void addLivro();
      2. public void atualizarLivro();
      3. public void mostrarLivros();
      4. private int codigo();
      5. public void Comprar();
      6. private int totalDeExemplares();
      7. public void verDetalhes();
      8. public void menu();
   4. SistemaDeLogin:
      1. public void fazerLogin();
      2. public static void cadastrar(SistemaDeLogin sistema ,Usuario usuario);
      3. public static void deletar(SistemaDeLogin sistema);
      4. public static void menu(SistemaDeLogin sistema);
   5. Usuario
      1. public void Cadastrar();
      2. public boolean equals(Usuario obj);
   6. Filmes:
      1. public final void detalhes();
   7. Livros:
      1. public boolean comprarlivro();
      2. public void atualizar();
      3. public void detalhes();
      4. public void Menu();
   8. Obs: nas classes Usuario e Filmes não encontrei motivos de adicionar mais funções ao projeto.

**Requisitos de implementação**

1. Cinco classes: uma superclasse e duas subclasses, e duas classes relacionadas ao projeto
   1. Amazon : super classe.
   2. LojaOnline e InstantVideo : sub classes
   3. Usuario,SistemaDeLogin,Livros,Filmes : classes relacionadas ao projeto.
2. Sempre usar o super para o máximo de reaproveitamento de código
   1. InstantVideo:
      * 1. public void menu(){
        2. super.menu(); // na subclasse InstantVideo foi usado a sobrescrita do método menu entretanto ainda se necessita do método original para isso foi reutilizado o método menu da super classe.
        3. public InstantVideo() {
        4. super();// utilização do super na construção da super classe.
   2. LojaOnline
      * 1. public void detalhes(){
        2. super.detalhes();// como anteriormente o método foi sobrescrito por isso foi utilizado o super da classe superior.
        3. public void menu(){
        4. super.menu();// novamente na função menu, so que agora da classe LojaOnline.
        5. public LojaOnline(){
        6. super();// no construtor para inicializar a superclasse.
3. Diagrama de classes (obrigatório salvar também o png do diagrama no gitHub) ok
4. Todos os atributos devem ser inicializados. Fez validação de dados?
   1. Amazon:
      1. public void setSatisfacaodocliente() {
      2. Scanner in = new Scanner(System.in);
      3. int escolha;
      4. do{
      5. System.out.println("digite sua satisfação com nossos sistemas");
      6. System.out.println("\n1- Ruim\n2- Bom\n3- Exelente");
      7. escolha=in.nextInt();
      8. switch(escolha){
      9. case 1:
      10. this.satisfacaodocliente = Nota.Ruim;
      11. break;
      12. case 2:
      13. this.satisfacaodocliente = Nota.Bom;
      14. break;
      15. case 3:
      16. this.satisfacaodocliente = Nota.Exelente;
      17. break;
      18. default:
      19. System.out.println("digite um valor valido!!");
      20. }
      21. }while(escolha < 1 && escolha > 3);
      22. }
      23. A entrada é feita apenas pelo setSatisfacaodocliente, pois a outra variável (CNPJ) é static final;
   2. InstantVideo:
      1. public InstantVideo(ArrayList<Filmes> filmes, ArrayList<Filmes> filmesalugados) {
      2. super();
      3. setNegocio();
      4. InstantVideo.filmes = filmes;
      5. this.filmesalugados = filmesalugados;
      6. }
      7. Como Filmes e filmes alugados são apenas arrays a validação é feita pela classe array e pelos respectivos tipos.
      8. public final void setNegocio() {
      9. Scanner in = new Scanner(System.in);
      10. int escolha;
      11. do{
      12. System.out.println("digite o tipo de compra");
      13. System.out.println("\n1- Alugar\n2- Comprar");
      14. escolha=in.nextInt();
      15. switch(escolha){
      16. case 1:
      17. this.negocio = Tipodenegocio.ALUGAR;
      18. break;
      19. case 2:
      20. this.negocio = Tipodenegocio.COMPRAR;
      21. break;
      22. default:
      23. System.out.println("digite um valor valido!!");
      24. }
      25. }while(escolha < 1 && escolha > 2);
      26. }
      27. No setNegocio e dado na verdade um menu de escolha já que o dado é do tipo enum.
   3. LojaOnline:
      1. public LojaOnline(int tamanhoinicial) {
      2. super();
      3. setFormadepagamento();
      4. setLivros(tamanhoinicial);
      5. setLivrosccomprados(0);
      6. }
      7. Toda a validação foi feita no setLivros().
      8. public void setLivros(int tamanhoinicial) {
      9. if (tamanhoinicial > 0)
      10. this.livros = new ArrayList<>(tamanhoinicial);
      11. else
      12. this.livros = new ArrayList<>(tamanhoinicial);
      13. }
      14. Verificar se o tamanho o vetor é menor ou igual a 0;
   4. SistemaDeLogin:
      1. Como se trata apenas de um sistema para entrada de dados e possui apenas vetores não é necessário a validação de dados.
   5. Usuario:
      1. public Usuario(String nome, String Login,String senha,String endereco) {
      2. setNome(nome);
      3. setLoguin(Login);
      4. setSenha(senha);
      5. this.endereco = endereco;
      6. }
      7. Como a classe pode receber todo o tipo de entrada já que é String fica para o meu nível de conhecimento difícil de validar.
   6. Livros:
      1. public Livros(float preco, String nome, int numerodepaginas,int numerodeexemplares) {
      2. setPreco(preco);
      3. setNome(nome);
      4. setTiposdelivros();
      5. setNumerodepaginas(numerodepaginas);
      6. setNumerodeexemplares(numerodeexemplares);
      7. }
      8. Validação feita nos sets.
         1. public final void setPreco(float preco) {
         2. if (preco < 0){
         3. this.preco = 00;
         4. }else
         5. this.preco = preco;
         6. }
         7. public final void setTiposdelivros() {
         8. Scanner in =new Scanner(System.in);
         9. int escolha;
         10. System.out.println("ESCOLHA O GENERO DE FILME: ");
         11. do{
         12. System.out.println("1- ROMANCE\n2- AVENTURA\n3FICCAO\_CIENTIFICA");
         13. escolha = in.nextInt();
         14. switch(escolha){
         15. case 1:
         16. this.tiposdelivros=Genero.ROMANCE;
         17. break;
         18. case 2:
         19. this.tiposdelivros=Genero.AVENTURA;
         20. break;
         21. case 3:
         22. this.tiposdelivros=Genero.FICCAO\_CIENTIFICA;
         23. break;
         24. case 4:
         25. this.tiposdelivros=Genero.OUTROS;
         26. break;
         27. default :
         28. System.out.println("Digite um valor valido");
         29. }
         30. }while(escolha < 4 || escolha < 1);
         31. }
         32. public final void setNumerodepaginas(int numerodepaginas) {
         33. if (numerodepaginas < 0){
         34. this.numerodepaginas=0;
         35. }else
         36. this.numerodepaginas = numerodepaginas;
         37. }
         38. public final void setNumerodeexemplares(int numerodeexemplares) {
         39. if (numerodeexemplares < 0){
         40. this.numerodeexemplares=0;
         41. }else
         42. this.numerodeexemplares = numerodeexemplares;
         43. }
   7. Filmes:
      * 1. public Filmes(float preco, String nome, int duracao) {
        2. setPreco(preco);
        3. setNome(nome);
        4. setTiposdeflme();
        5. setDuracao(duracao);
        6. }
        7. public final void setPreco(float preco) {
        8. if (preco < 0){
        9. this.preco=0;
        10. }else
        11. this.preco = preco;
        12. }
        13. public final void setTiposdeflme() {
        14. Scanner in =new Scanner(System.in);
        15. int escolha;
        16. System.out.println("ESCOLHA O GENERO DE FILME: ");
        17. do{
        18. System.out.println("1- ROMANCE\n2- AVENTURA\n3FICCAO\_CIENTIFICA");
        19. escolha = in.nextInt();
        20. switch(escolha){
        21. case 1:
        22. this.tiposdeflme=Genero.ROMANCE;
        23. break;
        24. case 2:
        25. this.tiposdeflme=Genero.AVENTURA;
        26. break;
        27. case 3:
        28. this.tiposdeflme=Genero.FICCAO\_CIENTIFICA;
        29. break;
        30. case 4:
        31. this.tiposdeflme=Genero.OUTROS;
        32. break;
        33. default :
        34. System.out.println("Digite um valor valido");
        35. }
        36. }while(escolha < 4 || escolha < 1);
        37. }
        38. public final void setDuracao(float duracao) {
        39. if (duracao > 0)
        40. this.duracao = duracao;
        41. else
        42. this.duracao=1;
        43. }
5. Dois construtores
   1. Amazon:
      * 1. public Amazon(){
        2. this.satisfacaodocliente = Nota.Indefinida;
        3. }
        4. public Amazon(Nota satisfacaodocliente) {
        5. this.satisfacaodocliente=satisfacaodocliente;
        6. }
   2. InstantVideo:
      * 1. public InstantVideo(ArrayList<Filmes> filmes, ArrayList<Filmes> filmesalugados) {
        2. super();
        3. setNegocio();
        4. InstantVideo.filmes = filmes;
        5. this.filmesalugados = filmesalugados;
        6. }
        7. public InstantVideo() {
        8. super();
        9. this.negocio= Tipodenegocio.INDEFINIDOO;
        10. InstantVideo.filmes = new ArrayList<>();
        11. this.filmesalugados= new ArrayList<>(3);
        12. this.filmescomprados = new ArrayList<>();
        13. }
   3. LojaOnline:
      * 1. public LojaOnline(int tamanhoinicial) {
        2. super();
        3. setFormadepagamento();
        4. setLivros(tamanhoinicial);
        5. setLivrosccomprados(0);
        6. }
        7. public LojaOnline(){
        8. super();
        9. this.formadepagamento=Tipodecompra.CARTAO\_DE\_CREDITO;
        10. setLivros(1);
        11. setLivrosccomprados(0);
        12. }
   4. SistemaDeLogin:
      * 1. public SistemaDeLogin() {
        2. this.usuarios = new ArrayList<>();
        3. this.amazon = new ArrayList<>();
        4. numerodecadastros=0;
        5. }
        6. public SistemaDeLogin(ArrayList<Usuario> usuarios, ArrayList<Amazon> amazon) {
        7. this.usuarios = usuarios;
        8. this.amazon = amazon;
        9. numerodecadastros=0;
        10. }
   5. Usuario:
      * 1. public Usuario(String nome, String Login,String senha,String endereco) {
        2. setNome(nome);
        3. setLoguin(Login);
        4. setSenha(senha);
        5. this.endereco = endereco;
        6. }
        7. public Usuario() {
        8. setNome("desconhecido");
        9. setLoguin("sem nome");
        10. setSenha("");
        11. this.endereco = " ";
        12. }
   6. Livros:
      * 1. public Livros(float preco, String nome, int numerodepaginas,int numerodeexemplares) {
        2. setPreco(preco);
        3. setNome(nome);
        4. setTiposdelivros();
        5. setNumerodepaginas(numerodepaginas);
        6. setNumerodeexemplares(numerodeexemplares);
        7. }
        8. public Livros() {
        9. this.preco = 00;
        10. this.nome = "sem nome";
        11. this.tiposdelivros = Genero.OUTROS;
        12. this.numerodepaginas = 0;
        13. }
   7. Filmes:
      * 1. public Filmes() {
        2. this.preco = 00;
        3. this.nome = "Sem nome";
        4. this.tiposdeflme = Genero.OUTROS;
        5. this.duracao = 0;
        6. }
        7. public Filmes(float preco, String nome, int duracao) {
        8. setPreco(preco);
        9. setNome(nome);
        10. setTiposdeflme();
        11. setDuracao(duracao);
        12. }
6. Um atributo static.
   1. Amazon:
      1. protected final static String CNPJ = "15.436.940/0001-03";
   2. InstantVideo
      1. protected static ArrayList<Filmes> filmes;
   3. LojaOnline:
      1. protected static int quantidadedelivros = 0;
   4. SistemaDeLogin:
      1. private static int numerodecadastros=0;
7. Um atributo const static
   1. Amazon:
      1. protected final static String CNPJ = "15.436.940/0001-03";

*Correta modelagem dos statics?*

1. Um array
   1. InstantVideo:
      1. protected static ArrayList<Filmes> filmes;
      2. protected ArrayList<Filmes> filmesalugados;
      3. protected ArrayList<Filmes> filmescomprados;
   2. LojaOnline:
      1. protected ArrayList<Livros> livros;
   3. SistemaDeLogin:
      1. private ArrayList<Usuario> usuarios;
      2. private ArrayList<Amazon> amazon;
2. Método static – deve ser chamado no main
   1. Classe Principal:
      1. SistemaDeLogin.menu(sistema);
3. Sobrescrita de método: chamar dentro do método da classe. Usar o @override
   1. InstantVideo:
      1. @Override
      2. public void menu(){
      3. super.menu();
      4. Scanner in = new Scanner(System.in);
      5. int escolha;
      6. do{
      7. System.out.println("###########MENU%%%%%%%%%%%\n1- Alugar \n2- comprar \n3- Adicionar ");
      8. escolha = in.nextInt();
      10. switch(escolha){
      11. case 1:
      12. this.alugar();
      13. break;
      14. case 2:
      15. this.comprar();
      16. break;
      17. case 3:
      18. this.add();
      19. break;
      20. default:
      21. System.out.println("Digite um vvalor valido");
      22. }
      23. }while(escolha >3 && 1 > escolha);
      24. }
   2. LojaOnline:
      * 1. @Override
        2. public void detalhes(){
        3. super.detalhes();
        4. System.out.println("-------Loja Online-------");
        5. System.out.println("voce escolheu pagar com: "+this.formadepagamento.name()+".");
        6. System.out.println("voce ja comprou: "+this.livrosccomprados+" livros.");
        7. System.out.println("a loja possui: "+quantidadedelivros+" Livros diferemtes.");
        8. System.out.println("E um total de: "+totalDeExemplares());
        9. }
        10. @Override
        11. public void menu(){
        12. super.menu();
        13. Scanner in = new Scanner(System.in);
        14. int escolha;
        15. do{
        16. System.out.println("###########MENU%%%%%%%%%%%\n1- Comprar \n2- atualizar \n3- Adicionar \n4ver detahees ");
        17. escolha = in.nextInt();
        19. switch(escolha){
        20. case 1:
        21. this.Comprar();
        22. break;
        23. case 2:
        24. this.atualizarLivro();
        25. break;
        26. case 3:
        27. this.addLivro();
        28. break;
        29. case 4:
        30. this.detalhes();
        31. break;
        32. default:
        33. System.out.println("Digite um valor valido");
        34. }
        36. }while(escolha >4 && 1 > escolha);
        37. }
4. Usar Protected acessando diretamente os atributos na classe derivada
   1. Amazon: (super classe);
      1. protected final static String CNPJ = "15.436.940/0001-03";
      2. protected Nota satisfacaodocliente;
   2. InstantVideo:
      1. System.out.println(this.lema);//chamada de uma variarel da super classe na subcasse
      2. Isso é feito em todos os métodos de uso.
   3. LojaOnline:
      1. System.out.println(this.lema);//chamada de uma variarel da super classe na subcasse
      2. Da mesma forma é chamada a vasriavel lema diretamente pois esta é protected.
5. Usar suas classes no main
   1. É chamado no main apenas a classe SistemaDeLogin pois, nesta estão inseridas as outras classes.

Opcionais que garantem pontos extras:

Trabalhar com ENUM

1. Amazon:

protected enum Nota { Ruim, Bom, Exelente, Indefinida};

1. InstantVideo:

protected enum Tipodenegocio {ALUGAR , COMPRAR, INDEFINIDOO};

1. LojaOnline:

public enum Tipodecompra { BOLETO\_BAMCARIO, CARTAO\_DE\_CREDITO };

Trabalhar com pacotes

Pacote Amazon que possui as classes Amazon, IntantVideo e LojaOnline.

Pacote Usuario que possui as classes Usuarios e SistemaDeLogin.

Pacote Produtos que possui as classes Filmes e Livros.

Pacote Principal que possui a classe principal (única que possui o método main).