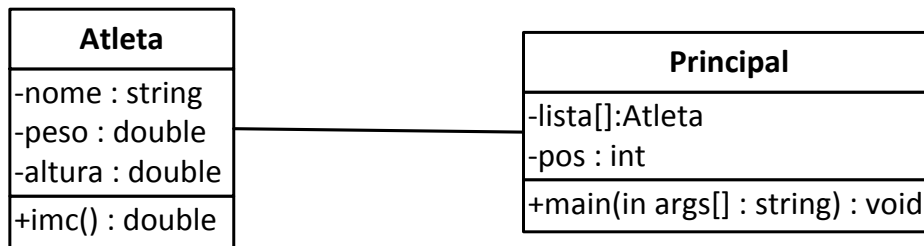


Classes – Matrizes de Objetos

Considere o seguinte problema em UML:



Crie um projeto chamado prjAtleta no netbeans:

Nome e Localização

Nome do Projeto:

Localização do Projeto: Procurar...

Pasta do Projeto:

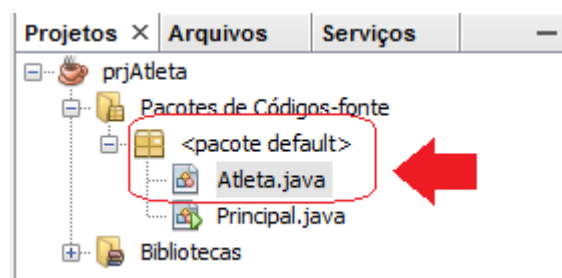
☐ Usar Pasta Dedicada para Armazenar Bibliotecas

Pasta Bibliotecas: Procurar...

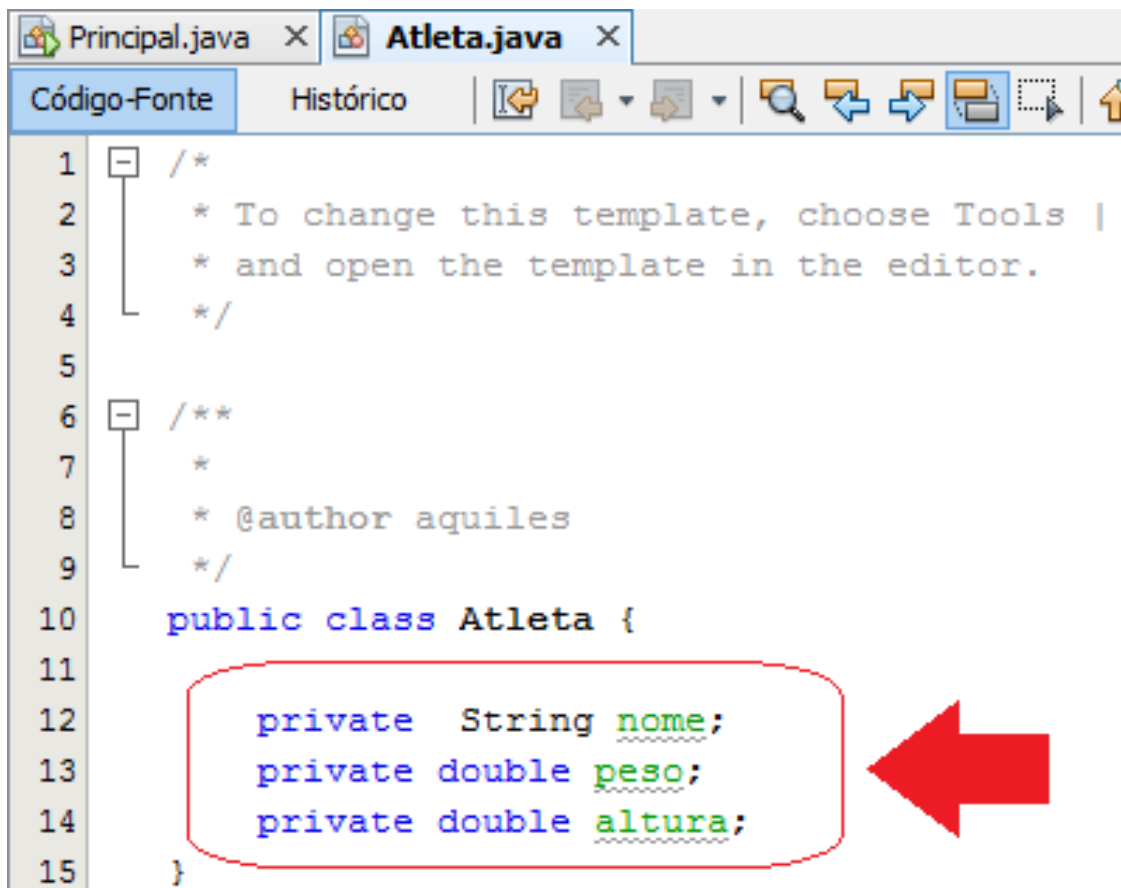
Usuários e projetos diferentes podem compartilhar as mesmas bibliotecas de compilação (consulte a Ajuda para obter detalhes).

☒ Criar Classe Principal

Adicione ao seu projeto a classe Atleta:

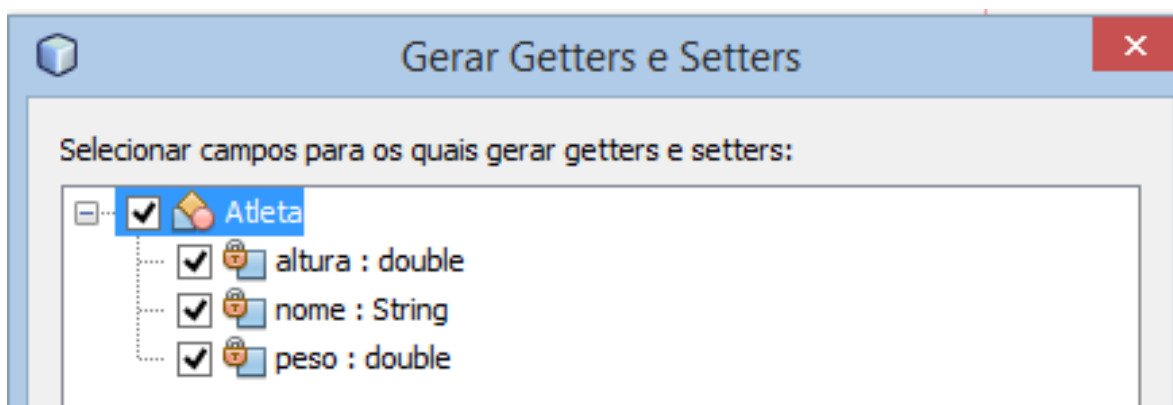


Codifique as propriedades:

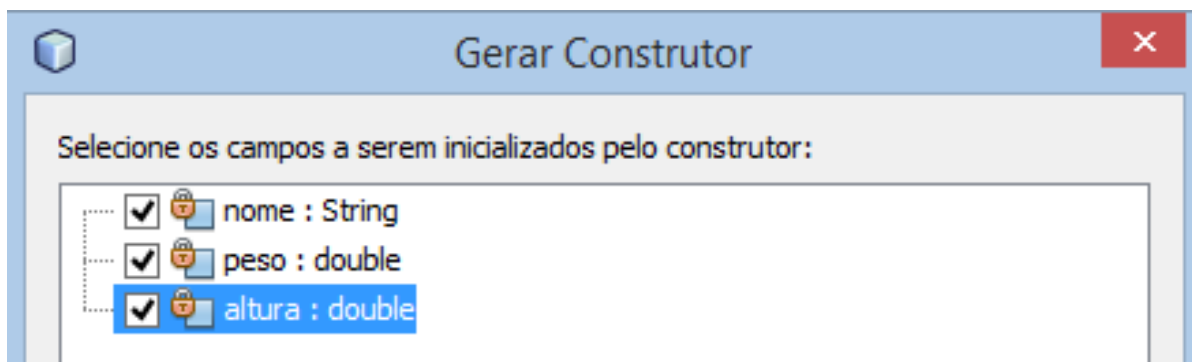


```
1  /*
2   * To change this template, choose Tools |
3   * and open the template in the editor.
4   */
5
6  /**
7   *
8   * @author aquiles
9   */
10 public class Atleta {
11
12     private String nome;
13     private double peso;
14     private double altura;
15 }
```

Acrescentes os getters e setters:



Crie em seguida o construtor padrão:



Logo após o construtor acrescentar o código do método do cálculo do IMC:


```

public class Principal {

    // vetor que irá conter a lista de atletas

    static Atleta[] lista;
    // posição na lista, com posição inicial zero
    static int pos = 0;

    public static void main(String[] args) {
        // a lista terá 10 atletas
        lista = new Atleta[10];
        int op;
        do{
            op = menu(); // opção selecionada
            switch(op){

                case 5: JOptionPane.showMessageDialog(null,
                    "Obrigado por usar o programa"); break;

            }

        } while(op!=5);
    }

    // método de menu:
    private static int menu() {
        return Integer.parseInt(
            JOptionPane.showInputDialog ("Menu:\n"
                + "\n1 - Cadastrar Atleta"
                + "\n2 - Pesquisar Atleta"
                + "\n3 - Excluir Atleta"
                + "\n4 - Relatório"
                + "\n5 -Sair")
        );
    }
}

```

Vamos codificar a rotina de cadastro conforme a tela seguinte:



```

public static void main(String[] args) {
    // a lista terá 10 atletas
    lista = new Atleta[10];
    int op;
    do{
        op = menu(); // opção selecionada
        switch(op){
            case 1: cadastrar(); // função cadastro
                    break;

            case 5: JOptionPane.showMessageDialog (null,
                "Obrigado por usar o programa"); break;
        }
    } while (op!=5);

    private static void cadastrar () {
    }
}

```

O código do cadastrar irá iniciar na posição zero da tabela e cadastra o atleta, avançando para a posição seguinte até atingir a décima posição, preenchendo todo o vetor. Note que cada posição contém inicialmente o valor null, portanto cabe a nós criar um objeto da classe atleta e adicionar o mesmo na lista.

```



private static void cadastrar () {

    // verificar se a lista está cheia
    if( pos == 10){
        JOptionPane.showMessageDialog (null,"lista esta cheia!");
        return;
    }
    // cria um atleta
    Atleta novo = new Atleta("",0,0);
    // preenche o nome, peso e altura do novo atleta:
    novo.setNome(JOptionPane.showInputDialog ("Digite o Nome:"));
    novo.setPeso(Double.parseDouble (JOptionPane.showInputDialog ("Digite Peso:")));
    novo.setAltura(Double.parseDouble (JOptionPane.showInputDialog ("Digite Altura:")));
    // cadastra o novo atleta na lista
    lista[pos] = novo;
    // avança para o próximo registro vazio
    pos++;
    JOptionPane.showMessageDialog (null,"Atleta cadastrado com sucesso...");
}

```

Para testar vamos codificar a opção relatório que conterà todos os vetores da lista preenchidos e mostrará os dados dos atletas cadastrados:

```
public static void main(String[] args) {  
    // a lista terá 10 atletas  
    lista = new Atleta[10];  
    int op;  
    do{  
        op = menu(); // opção selecionada  
        switch(op){  
            case 1: cadastrar(); // função cadastro  
                    break;  
            case 4: relatorio(); // função relatório  
                    break;  
            case 5: JOptionPane.showMessageDialog (null,  
                "Obrigado por usar o programa");  
        }  
    } while(op!=5);  
}  
  
private static void relatorio() {  
}
```



O método relatório contém a seguinte estrutura:

```
private static void relatorio() {  
    // percorre cada item da lista  
    for( Atleta a: lista){  
        // se o item da lista não é nulo exibe na tela  
        if(a!=null){  
            JOptionPane.showMessageDialog (null, "Atleta\n"  
                + "\nNome:" + a.getNome ()  
                + "\nPeso:" + a.getPeso ()  
                + "\nAltura:" + a.getAltura ()  
                + "\nIMC:" + a.imc () );  
        }  
    }  
}
```

Execute e teste o programa.