Fatec Mococa

Linguagens de Programação

Classes Java

Sandra Cristina Costa Prado

Declaração

Venda ven;

A variável **ven** não armazenará os dados propriamente, mas sim uma **referência** de onde os dados serão armazenados.

A variável declarada **ven** armazena null.

Instanciação

Venda ven = new Venda();

O operador **new** retorna uma referência (endereço de memória). A referência é a indicação de onde o objeto está armazenado

ven não é mais null. Agora armazena um endereço de memória

Construtor

```
Venda ven = new Venda();
```

Classe
Define os objetos.
É um arquivo de texto.

Construtor
Tem o mesmo nome da classe.
Executa (realiza, torna real)
o que está especificado na classe.

Referência para o objeto
O objeto é o ator principal, pois é ele que
1) Armazena os dados
2) Realiza as operações sobre os dados

Classes e objetos

Classe é um conceito estático: uma classe é um elemento reconhecido num texto de programa.

Objeto é um conceito puramente dinâmico: não pertence ao texto do programa, mas à memória do computador, um local que ele ocupa durante a execução do programa.

As características do objeto (dados que deve armazenar, operações que deve realizar, como deve interagir com outros objetos, etc.) são definidas na classe a que ele pertence.

Onde estão os objetos?

Quando uma variável é do tipo de uma classe, diz-se que ela referencia um objeto. Informalmente, ela é chamada de objeto, embora não armazene o objeto de fato.

Uma razão para este comportamento é que um objeto pode ser muito grande. É mais eficiente armazenar sua localização do que o próprio objeto.

A referência **ven** fica numa região chamada stack e o objeto referenciado por **ven** fica numa região chamada heap.

Exemplo

```
1  package lojaprestacoes;
2
3  public class Venda {
4
5      String cliente;
6      double valorBase;
7      int numPrest;
8  }
```

Campos

```
1  package lojaprestacoes;
2
3  public class Venda {
4
5      String cliente;
6      double valorBase;
7      int numPrest;
8  }
```

Campos são variáveis que

- são declaradas como membros da classe ou
- são declaradas fora de qualquer método ou construtor dentro da classe.

Campos

```java
package lojaprestacoes;

public class Venda {

    String cliente;
    double valorBase;
    int numPrest;
}
```

Campos são variáveis que
- são declaradas como membros da classe ou
- são declaradas fora de qualquer método ou construtor

dentro da classe.

```java
package lojaprestacoes;

public class Venda {

    String cliente;
    double valorBase;
    int numPrest;

    //Exemplos de construtores
    public Venda(){
        // Este é o construtor padrão.
        // É inserido automaticamente pelo compilador.
        // Mas, se inserirmos um outro construtor, o compilador
        // não insere este
    }

    public Venda(String cliente, double valorBase, int numPrest) {
        this.cliente = cliente;
        this.valorBase = valorBase;
        this.numPrest = numPrest;
    }
}
```

Campos e variáveis locais

```java
package lojaprestacoes;

public class Venda {

    String cliente;
    double valorBase;
    int numPrest;
}
```

campos

```java
package lojaprestacoes;

public class LojaPrestacoes {

    public static void main(String[] args) {

        Venda ven[] = new Venda[10];

        String cliente[] = {"Ana", "Luis", "Bia", "Ivo", "Ana", "Luis", "Bia", "Ana", "Bia", "Ian"};
        double valorBase[] = {400.00, 700.00, 200.00, 900.00, 500.00, 800.00, 100.00, 600.00, 300.00, 150.00};
        int numPrestacoes[] = {1, 2, 1, 3, 2, 3, 1, 3, 2, 1};

        for (int i = 0; i < ven.length; i++) {
            ven[i] = new Venda();
            ven[i].cliente = cliente[i];
            ven[i].valorBase = valorBase[i];
            ven[i].numPrest = numPrestacoes[i];
            System.out.printf("%10s %10.2f %10d\n", ven[i].cliente,
                    ven[i].valorBase,
                    ven[i].numPrest);
        }
    }
}
```

método

Variáveis locais