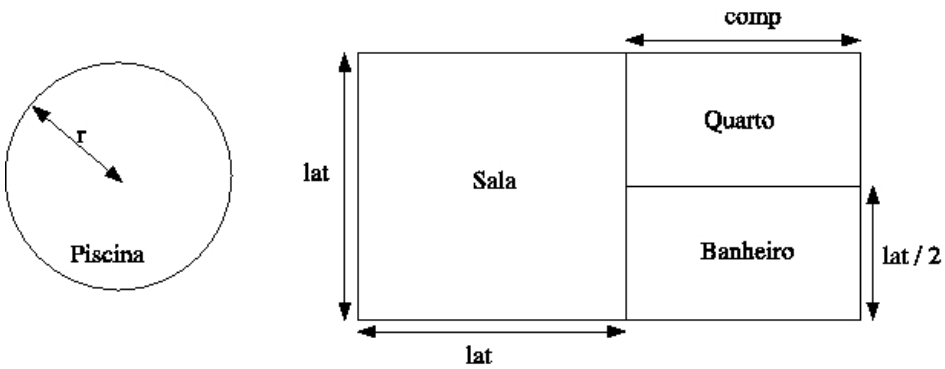
Aula 04 – Subrotinas

Norton Trevisan Roman

15 de março de 2013

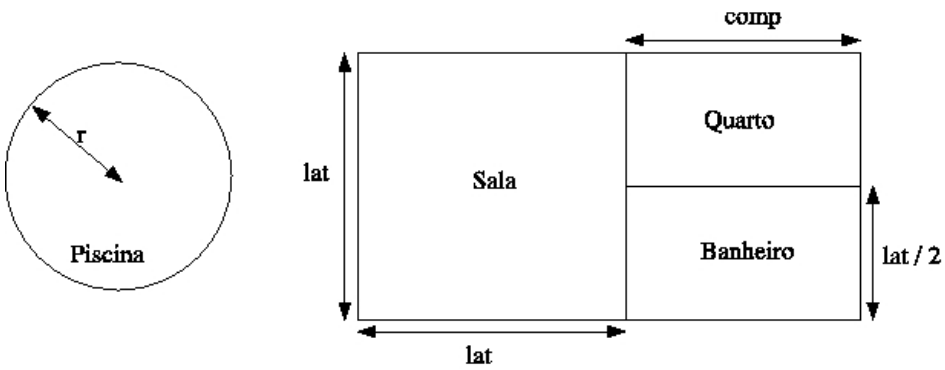
Constantes

- Suponha que queremos incrementar nossa cabana com uma piscina:



Constantes

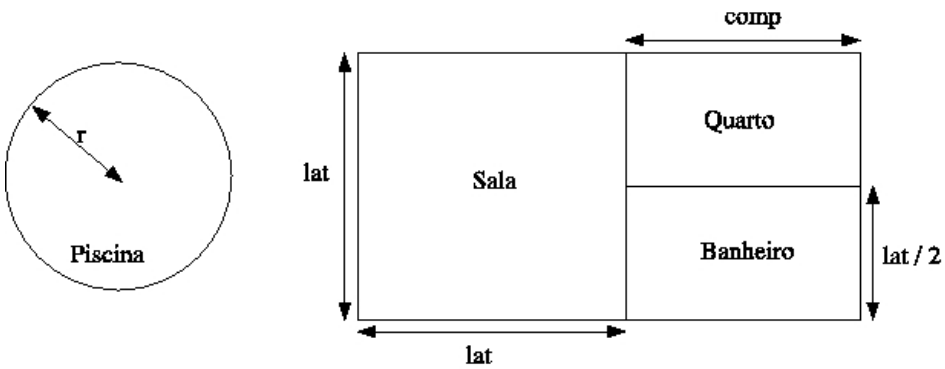
- Suponha que queremos incrementar nossa cabana com uma piscina:



- Queremos então fazer um programa que calcule a área da cabana e da piscina.

Constantes

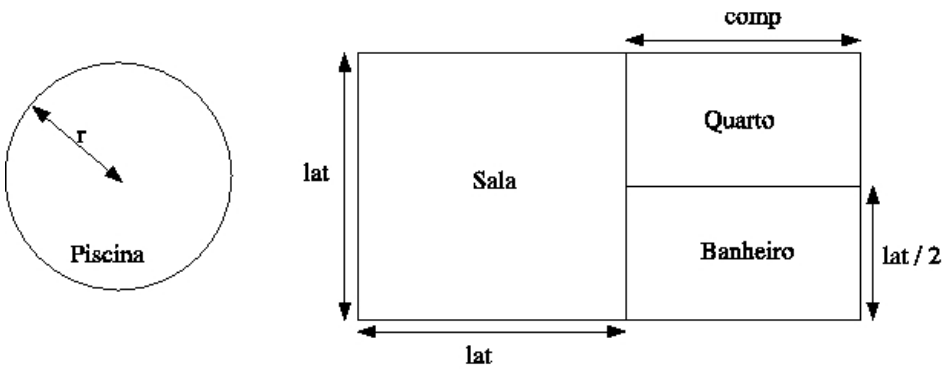
- Suponha que queremos incrementar nossa cabana com uma piscina:



- Queremos então fazer um programa que calcule a área da cabana e da piscina.
  - Como?

Constantes

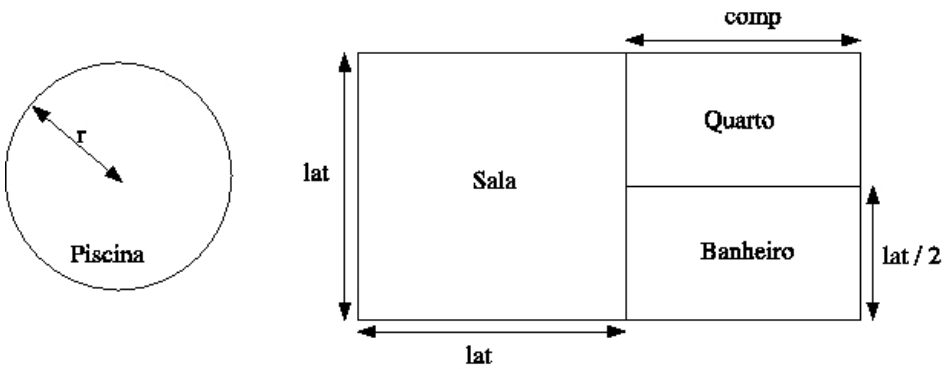
- Suponha que queremos incrementar nossa cabana com uma piscina:



- Queremos então fazer um programa que calcule a área da cabana e da piscina.
  - Como?
    - ★ Temos o raio da piscina

Constantes

- Suponha que queremos incrementar nossa cabana com uma piscina:



- Queremos então fazer um programa que calcule a área da cabana e da piscina.
  - Como?
    - ★ Temos o raio da piscina
    - ★ Basta vermos como adicionar o π

Constantes

- Podemos fazer:

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Podemos fazer:
- E a saída será
  - "Área: 12.56636"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Podemos fazer:
- E a saída será
  - "Área: 12.56636"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- Há algum problema com isso?

Constantes

- Podemos fazer:
- E a saída será
  - "Área: 12.56636"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- Há algum problema com isso?
  - E se fizermos:

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        pi = 12;
        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Podemos fazer:
- E a saída será
  - "Área: 12.56636"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- Há algum problema com isso?
  - E se fizermos:
  - Teremos "Área: 48.0"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        pi = 12;
        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Podemos fazer:
- E a saída será
  - "Área: 12.56636"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- Há algum problema com isso?
  - E se fizermos:
  - Teremos "Área: 48.0"
- O que aconteceu?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        pi = 12;
        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Podemos fazer:
- E a saída será
  - "Área: 12.56636"

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- Há algum problema com isso?
  - E se fizermos:
  - Teremos "Área: 48.0"
- O que aconteceu?
  - Inadvertidamente mudamos algo que deveria ser constante

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        double pi = 3.14159; // valor do pi

        pi = 12;
        areap = pi * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Devemos tornar π constante, fazendo:

```
final double PI = 3.14159;
```

Constantes

- Devemos tornar π constante, fazendo:

```
final double PI = 3.14159;
```

- E o que aconteceria com esse programa?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        final double PI = 3.14159; // valor do pi

        PI = 12;
        areap = PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Devemos tornar π constante, fazendo:

```
final double PI = 3.14159;
```

- E o que aconteceria com esse programa?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        final double PI = 3.14159; // valor do pi

        PI = 12;
        areap = PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

```
$ javac AreaPiscina.java
AreaPiscina.java:11: cannot assign a value to
final variable PI
        PI = 12;
        ^
1 error
```

Constantes

- Devemos tornar π constante, fazendo:

```
final double PI = 3.14159;
```

- E o que aconteceria com esse programa?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        final double PI = 3.14159; // valor do pi

        PI = 12;
        areap = PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

```
$ javac AreaPiscina.java
AreaPiscina.java:11: cannot assign a value to
final variable PI
        PI = 12;
        ^
1 error
```

- "final" especifica que o valor não mais poderá ser mudado no programa

Constantes

- Devemos tornar π constante, fazendo:

```
final double PI = 3.14159;
```

- E o que aconteceria com esse programa?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina
        final double PI = 3.14159; // valor do pi

        PI = 12;
        areap = PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

```
$ javac AreaPiscina.java
AreaPiscina.java:11: cannot assign a value to
final variable PI
        PI = 12;
        ^
1 error
```

- "final" especifica que o valor não mais poderá ser mudado no programa
  - Define uma constante (representada em maiúscula, por convenção)

Constantes

- Alternativamente, poderíamos usar uma constante já definida no java:

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Alternativamente, poderíamos usar uma constante já definida no java:
  - `Math.PI`, valendo 3.141592653589793
  - `Math.PI` é *double*, por isso areap também o é

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Alternativamente, poderíamos usar uma constante já definida no java:
  - `Math.PI`, valendo 3.141592653589793
  - `Math.PI` é *double*, por isso areap também o é

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- E como podemos melhorar o `raio * raio`?

Constantes

- Alternativamente, poderíamos usar uma constante já definida no java:
  - `Math.PI`, valendo 3.141592653589793
  - `Math.PI` é *double*, por isso areap também o é

- E como podemos melhorar o `raio * raio`?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Constantes

- Alternativamente, poderíamos usar uma constante já definida no java:
  - `Math.PI`, valendo 3.141592653589793
  - `Math.PI` é *double*, por isso areap também o é

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * raio * raio;
        System.out.println("Área: "+areap);
    }
}
```

- E como podemos melhorar o `raio * raio`?
  - `Math.pow(a,b)` dá o resultado de a^b
  - O resultado também é *double*

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Métodos

- Math é como se fosse uma biblioteca, que nos fornece o método pow (além da constante PI)

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Métodos

- Math é como se fosse uma biblioteca, que nos fornece o método pow (além da constante PI)
  - Não é uma biblioteca, é mais que isso, mas veremos mais adiante

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Métodos

- Math é como se fosse uma biblioteca, que nos fornece o método pow (além da constante PI)
  - Não é uma biblioteca, é mais que isso, mas veremos mais adiante
- Método?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Métodos

- Math é como se fosse uma biblioteca, que nos fornece o método pow (além da constante PI)
  - Não é uma biblioteca, é mais que isso, mas veremos mais adiante
- Método?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Método

Um método é uma implementação de uma subrotina

Métodos

- Math é como se fosse uma biblioteca, que nos fornece o método pow (além da constante PI)
  - Não é uma biblioteca, é mais que isso, mas veremos mais adiante
- Método?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Método

Um método é uma implementação de uma subrotina

- Nesse caso, `pow(a,b)` recebe dois valores, a e b, devolvendo o resultado de a^b

Métodos

- Math é como se fosse uma biblioteca, que nos fornece o método pow (além da constante PI)
  - Não é uma biblioteca, é mais que isso, mas veremos mais adiante
- Método?

```
/*
Programa para calcular a área de uma piscina
redonda.
*/
class AreaPiscina {
    public static void main(String[] args) {
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        areap = Math.PI * Math.pow(raio,2);
        System.out.println("Área: "+areap);
    }
}
```

Método

Um método é uma implementação de uma subrotina

- Nesse caso, `pow(a,b)` recebe dois valores, a e b, devolvendo o resultado de a^b
  - Os valores a e b fornecidos ao método são chamados argumentos de seus parâmetros

Métodos

- Vamos então juntar os dois programas que vimos até agora em um só:

```
class AreaCasa {
    public static void main(String[] args) {
        float lateral = 11; // lateral da cabana
        float cquarto = 7; // comprimento do quarto
        float areaq; // área do quarto
        float areas; // área da sala
        float areat; // área total
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        System.out.println("Cálculo da área da casa");
        // cálculo da área da sala
        areas = lateral*lateral;
        System.out.println("A área da sala é "+areas);
        // cálculo da área do banheiro
        areaq = cquarto*(lateral/2);
        System.out.println("A área do banheiro é "+areaq);
        // cálculo da área do quarto
        System.out.println("A área do quarto é "+areaq);
        // cálculo da área total
        areat = areas + 2*areaq;
        System.out.println("A área total é " + areat);
        // cálculo da área da piscina
        areap = Math.PI * Math.pow(raio,2);
        System.out.println("A área da piscina é "+areap);
    }
}
```

Métodos

- Vamos então juntar os dois programas que vimos até agora em um só:
- E qual a saida?

```
class AreaCasa {
    public static void main(String[] args) {
        float lateral = 11; // lateral da cabana
        float cquarto = 7; // comprimento do quarto
        float areaq; // área do quarto
        float areas; // área da sala
        float areat; // área total
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        System.out.println("Cálculo da área da casa");
        // cálculo da área da sala
        areas = lateral*lateral;
        System.out.println("A área da sala é "+areas);
        // cálculo da área do banheiro
        areaq = cquarto*(lateral/2);
        System.out.println("A área do banheiro é "+areaq);
        // cálculo da área do quarto
        System.out.println("A área do quarto é "+areaq);
        // cálculo da área total
        areat = areas + 2*areaq;
        System.out.println("A área total é " + areat);
        // cálculo da área da piscina
        areap = Math.PI * Math.pow(raio,2);
        System.out.println("A área da piscina é "+areap);
    }
}
```

Métodos

- Vamos então juntar os dois programas que vimos até agora em um só:
- E qual a saida?

```
Cálculo da área da casa
A área da sala é 121.0
A área do banheiro é 38.5
A área do quarto é 38.5
A área total é 198.0
A área da piscina é 12.566370614359172
```

```
class AreaCasa {
    public static void main(String[] args) {
        float lateral = 11; // lateral da cabana
        float cquarto = 7; // comprimento do quarto
        float areaq; // área do quarto
        float areas; // área da sala
        float areat; // área total
        double raio = 2; // raio da piscina
        double areap; // área da piscina

        System.out.println("Cálculo da área da casa");
        // cálculo da área da sala
        areas = lateral*lateral;
        System.out.println("A área da sala é "+areas);
        // cálculo da área do banheiro
        areaq = cquarto*(lateral/2);
        System.out.println("A área do banheiro é "+areaq);
        // cálculo da área do quarto
        System.out.println("A área do quarto é "+areaq);
        // cálculo da área total
        areat = areas + 2*areaq;
        System.out.println("A área total é " + areat);
        // cálculo da área da piscina
        areap = Math.PI * Math.pow(raio,2);
        System.out.println("A área da piscina é "+areap);
    }
}
```

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?
  - Podemos dividi-lo em 2 partes:

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?
  - Podemos dividi-lo em 2 partes:
    - ★ Uma para o cálculo da casa

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?
  - Podemos dividi-lo em 2 partes:
    - ★ Uma para o cálculo da casa
    - ★ Outra para o cálculo da piscina

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?
  - Podemos dividi-lo em 2 partes:
    - ★ Uma para o cálculo da casa
    - ★ Outra para o cálculo da piscina
- Como?

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?
  - Podemos dividi-lo em 2 partes:
    - ★ Uma para o cálculo da casa
    - ★ Outra para o cálculo da piscina
- Como?
  - Criando nossos próprios métodos:

Métodos

- Esse programa está ficando confuso.
  - Mistura a casa com a piscina
- Que fazer?
  - Podemos dividi-lo em 2 partes:
    - ★ Uma para o cálculo da casa
    - ★ Outra para o cálculo da piscina
- Como?
  - Criando nossos próprios métodos:

```
/* Calcula a área da casa */
static void areaCasa() {
    float lateral = 11; // comprimento da lateral da cabana
    float cquarto = 7; // lateral maior do quarto
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}

/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

Métodos

- O que significa o void?

```
/* Calcula a área da casa */
static void areaCasa() {
    float lateral = 11; // comprimento da lateral da cabana
    float cquarto = 7; // lateral maior do quarto
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Métodos

- O que significa o void?
  - Que o método não irá retornar nenhum valor

```
/* Calcula a área da casa */
static void areaCasa() {
    float lateral = 11; // comprimento da lateral da cabana
    float cquarto = 7; // lateral maior do quarto
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Métodos

- O que significa o void?
  - Que o método não irá retornar nenhum valor
  - Ele apenas executa a tarefa e termina

```
/* Calcula a área da casa */
static void areaCasa() {
    float lateral = 11; // comprimento da lateral da cabana
    float cquarto = 7; // lateral maior do quarto
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Métodos

- O que significa o void?
  - Que o método não irá retornar nenhum valor
  - Ele apenas executa a tarefa e termina
- E o static?

```
/* Calcula a área da casa */
static void areaCasa() {
    float lateral = 11; // comprimento da lateral da cabana
    float cquarto = 7; // lateral maior do quarto
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Métodos

- O que significa o void?
  - Que o método não irá retornar nenhum valor
  - Ele apenas executa a tarefa e termina
- E o static?
  - Por hora, apenas aceitemos...

```
/* Calcula a área da casa */
static void areaCasa() {
    float lateral = 11; // comprimento da lateral da cabana
    float cquarto = 7; // lateral maior do quarto
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Métodos

- O que significa o double?

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

Métodos

- O que significa o double?
  - Que o método irá retornar um valor do tipo double

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

Métodos

- O que significa o double?
  - Que o método irá retornar um valor do tipo double
  - Semelhante ao `pow(a,b)`

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

Métodos

- O que significa o double?
  - Que o método irá retornar um valor do tipo double
  - Semelhante ao `pow(a,b)`
- E o `return`?

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

Métodos

- O que significa o double?
  - Que o método irá retornar um valor do tipo double
  - Semelhante ao `pow(a,b)`
- E o `return`?
  - É quando o valor é efetivamente retornado

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

Métodos

- O que significa o double?
  - Que o método irá retornar um valor do tipo double
  - Semelhante ao `pow(a,b)`

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

- E o `return`?
  - É quando o valor é efetivamente retornado
  - A subrotina pára aí

Métodos

- O que significa o double?
  - Que o método irá retornar um valor do tipo double
  - Semelhante ao `pow(a,b)`

```
/* Calcula a área da piscina */
static double areaPiscina() {
    double raio = 2; // raio da piscina

    return(Math.PI * Math.pow(raio,2));
}
```

- E o `return`?
  - É quando o valor é efetivamente retornado
  - A subrotina pára aí
  - Alternativas:
    - `return(Math.PI * Math.pow(raio,2));`
    - `return Math.PI * Math.pow(raio,2);`

Métodos

- E como usamos isso no corpo do programa?

Métodos

- E como usamos isso no corpo do programa?

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina();
    System.out.println("A área da piscina é "
                       +areap);
}
```

Métodos

- E como usamos isso no corpo do programa?
  - Note que `areaPiscina()` retorna valor, então guardamos esse valor em `areap`

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina();
    System.out.println("A área da piscina é "
                       +areap);
}
```

Métodos

- E como usamos isso no corpo do programa?
  - Note que `areaPiscina()` retorna valor, então guardamos esse valor em `areap`
  - Já `areaCasa()` não retorna nada, então apenas a executamos

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina();
    System.out.println("A área da piscina é "
                       +areap);
}
```

Visão geral do código

```
/*
Programa para calcular a área de uma casa
(e seus cômodos) de 3 cômodos: uma sala de
10X10m, um banheiro e um quarto de 5X7m cada.
Calcula também a área de sua piscina.
*/
class AreaCasa {
    /*
        Calcula a área da casa
    */
    static void areaCasa() {
        float lateral = 11; // lateral da cabana
        float cquarto = 7; // lateral maior do quarto
        float areaq; // área do quarto
        float areas; // área da sala
        float areat; // área total

        System.out.println("Cálculo da área da casa");
        // cálculo da área da sala
        areas = lateral*lateral;
        System.out.println("A área da sala é "+areas);
        // cálculo da área do banheiro
        areaq = cquarto*(lateral/2);
        System.out.println("A área do banheiro é "
                           +areaq);
        // cálculo da área do quarto
        System.out.println("A área do quarto é "+areaq);
        // cálculo da área total
        areat = areas + 2*areaq;
        System.out.println("A área total é " + areat);
}
```

```
    /*
        Calcula a área da piscina
    */
    static double areaPiscina() {
        double raio = 2; // raio da piscina

        return(Math.PI * Math.pow(raio,2));
    }

    public static void main(String[] args) {
        double areap; // área da piscina

        areaCasa();

        areap = areaPiscina();
        System.out.println("A área da piscina é "
                           +areap);
    }
}
```

Métodos

- Qual a utilidade de criarmos nossos próprios métodos?

Métodos

- Qual a utilidade de criarmos nossos próprios métodos?
- Clareza:

Métodos

- Qual a utilidade de criarmos nossos próprios métodos?
- Clareza:
  - Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina();
    System.out.println("A área da piscina é "+areap);
}
```

Métodos

- Qual a utilidade de criarmos nossos próprios métodos?
- Clareza:
  - Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes
- Portabilidade:

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina();
    System.out.println("A área da piscina é "+areap);
}
```

Métodos

- Qual a utilidade de criarmos nossos próprios métodos?
- Clareza:
  - Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina();
    System.out.println("A área da piscina é "+areap);
}
```

- Portabilidade:
  - Se precisarmos, em outro programa, usar a mesma subrotina, ela já está separada

Parâmetros

- Nossos métodos, contudo, não são gerais

Parâmetros

- Nossos métodos, contudo, não são gerais
  - areaCasa() funciona apenas para casas da dimensão de nosso projeto

Parâmetros

- Nossos métodos, contudo, não são gerais
  - areaCasa() funciona apenas para casas da dimensão de nosso projeto
  - areaPiscina() funciona apenas para piscinas redondas de raio 2

Parâmetros

- Nossos métodos, contudo, não são gerais
  - areaCasa() funciona apenas para casas da dimensão de nosso projeto
  - areaPiscina() funciona apenas para piscinas redondas de raio 2
- Como poderíamos fazer para tornar esses procedimentos mais gerais?

Parâmetros

- Nossos métodos, contudo, não são gerais
  - areaCasa() funciona apenas para casas da dimensão de nosso projeto
  - areaPiscina() funciona apenas para piscinas redondas de raio 2
- Como poderíamos fazer para tornar esses procedimentos mais gerais?
  - Mantendo o formato da casa e da piscina

Parâmetros

- Nossos métodos, contudo, não são gerais
  - areaCasa() funciona apenas para casas da dimensão de nosso projeto
  - areaPiscina() funciona apenas para piscinas redondas de raio 2
- Como poderíamos fazer para tornar esses procedimentos mais gerais?
  - Mantendo o formato da casa e da piscina
  - Permitindo que seu tamanho varie

Parâmetros

- Nossos métodos, contudo, não são gerais
  - areaCasa() funciona apenas para casas da dimensão de nosso projeto
  - areaPiscina() funciona apenas para piscinas redondas de raio 2
- Como poderíamos fazer para tornar esses procedimentos mais gerais?
  - Mantendo o formato da casa e da piscina
  - Permitindo que seu tamanho varie

Com parâmetros:

```
/*
    Calcula a área da piscina
*/
static double areaPiscina(double raio) {
    return(Math.PI * Math.pow(raio,2));
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}
```

- Como chamamos esse método de outras partes do programa?

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`
- Como chamamos esse método de outras partes do programa?

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}


public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`
- Como chamamos esse método de outras partes do programa?
- E o que acontece ao chamarmos areaPiscina(2) de dentro do main?

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}


public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`
- Como chamamos esse método de outras partes do programa?
- E o que acontece ao chamarmos areaPiscina(2) de dentro do main?
  - O S.O. irá alocar memória para todas as variáveis e parâmetros declarados dentro do método...

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}


public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`
- Como chamamos esse método de outras partes do programa?
- E o que acontece ao chamarmos areaPiscina(2) de dentro do main?
  - O S.O. irá alocar memória para todas as variáveis e parâmetros declarados dentro do método...
  - Colocando o valor passado como parâmetro lá:

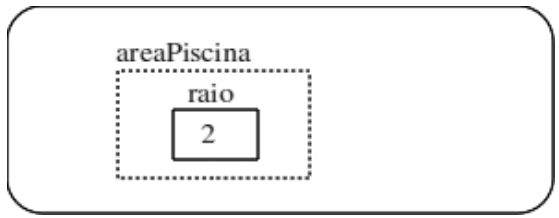
```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}


public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Parâmetros

- O método, agora, deve receber um valor (argumento) em seu parâmetro
  - Como o `Math.pow`

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}


public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa();

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

- Como chamamos esse método de outras partes do programa?
- E o que acontece ao chamarmos areaPiscina(2) de dentro do main?
  - O S.O. irá alocar memória para todas as variáveis e parâmetros declarados dentro do método...
  - Colocando o valor passado como parâmetro lá:



Parâmetros – passagem por valor

- Ao ato de passar um valor externo para dentro de um procedimento, via parâmetro, chamamos de passagem por valor

Parâmetros – passagem por valor

- Ao ato de passar um valor externo para dentro de um procedimento, via parâmetro, chamamos de passagem por valor
  - Nesse caso, o valor externo é copiado para a região de memória correspondente ao parâmetro.

Parâmetros – passagem por valor

- Ao ato de passar um valor externo para dentro de um procedimento, via parâmetro, chamamos de passagem por valor
  - Nesse caso, o valor externo é copiado para a região de memória correspondente ao parâmetro.

- O que acontece se tivermos algo assim?

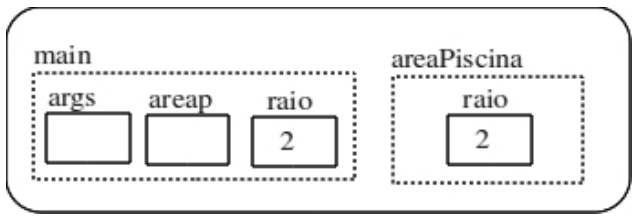
```
public static void main(String[] args) {
    double areap; // área da piscina
    double raio = 2; // raio da piscina

    areaCasa();

    areap = areaPiscina(raio);
    System.out.println("A área da piscina é "+areap);
}
```

Parâmetros – passagem por valor

- Ao ato de passar um valor externo para dentro de um procedimento, via parâmetro, chamamos de passagem por valor
  - Nesse caso, o valor externo é copiado para a região de memória correspondente ao parâmetro.

- O que acontece se tivermos algo assim?
  - O valor da variável raio, em main, é copiado para dentro da variável raio em areaPiscina

```
public static void main(String[] args) {
    double areap; // área da piscina
    double raio = 2; // raio da piscina

    areaCasa();

    areap = areaPiscina(raio);
    System.out.println("A área da piscina é "+areap);
}
```

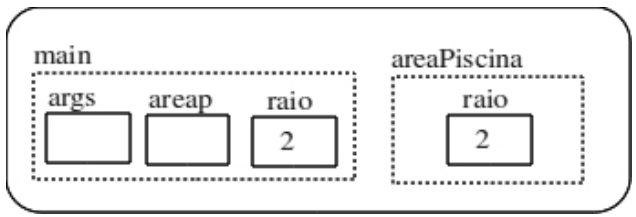


Parâmetros – passagem por valor

- Ao ato de passar um valor externo para dentro de um procedimento, via parâmetro, chamamos de passagem por valor
  - Nesse caso, o valor externo é copiado para a região de memória correspondente ao parâmetro.

- O que acontece se tivermos algo assim?
  - O valor da variável raio, em main, é copiado para dentro da variável raio em areaPiscina
    - ★ São duas variáveis diferentes, correspondendo a duas regiões de memória diferentes

```
public static void main(String[] args) {
    double areap; // área da piscina
    double raio = 2; // raio da piscina

    areaCasa();

    areap = areaPiscina(raio);
    System.out.println("A área da piscina é "+areap);
}
```

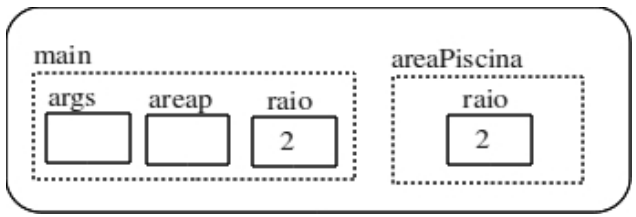


Parâmetros – passagem por valor

- Ao ato de passar um valor externo para dentro de um procedimento, via parâmetro, chamamos de passagem por valor
  - Nesse caso, o valor externo é copiado para a região de memória correspondente ao parâmetro.

- O que acontece se tivermos algo assim?
  - O valor da variável raio, em main, é copiado para dentro da variável raio em areaPiscina
    - ★ São duas variáveis diferentes, correspondendo a duas regiões de memória diferentes
    - ★ Sim... main é um método também

```
public static void main(String[] args) {
    double areap; // área da piscina
    double raio = 2; // raio da piscina

    areaCasa();

    areap = areaPiscina(raio);
    System.out.println("A área da piscina é "+areap);
}
```



Visão geral do código

- Incluindo parâmetros em areaCasa()

```
class AreaCasa {
    /*
        Calcula a área da casa

        Parâmetros:
            lateral - comprimento da lateral da cabana
            cquarto - lateral maior do quarto
    */
    static void areaCasa(float lateral, float cquarto) {
        float areaq; // área do quarto
        float areas; // área da sala
        float areat; // área total

        System.out.println("Cálculo da área da casa");
        // cálculo da área da sala
        areas = lateral*lateral;
        System.out.println("A área da sala é "+areas);
        // cálculo da área do banheiro
        areaq = cquarto*(lateral/2);
        System.out.println("A área do banheiro é "
                           +areaq);
        // cálculo da área do quarto
        System.out.println("A área do quarto é "+areaq);
        // cálculo da área total
        areat = areas + 2*areaq;
        System.out.println("A área total é " + areat);
}
```

```
/*
    Calcula a área da piscina

    Parâmetros:
        raio - O raio da piscina
*/
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
}
```

Parâmetros

- E como fica o método areaCasa na memória?

Parâmetros

- E como fica o método areaCasa na memória?

```
/*
    Calcula a área da casa

    Parâmetros:
        lateral - comprimento da lateral da cabana
        cquarto - lateral maior do quarto
*/
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Parâmetros

- E como fica o método areaCasa na memória?
- Ao ser chamado (ou invocado) em main, será separada uma região na memória para esse método

```
/*
    Calcula a área da casa

    Parâmetros:
        lateral - comprimento da lateral da cabana
        cquarto - lateral maior do quarto
*/
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Parâmetros

- E como fica o método areaCasa na memória?
- Ao ser chamado (ou invocado) em main, será separada uma região na memória para esse método
  - Contendo todas suas variáveis internas (chamadas de locais)

```
/*
    Calcula a área da casa

    Parâmetros:
        lateral - comprimento da lateral da cabana
        cquarto - lateral maior do quarto
*/
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Parâmetros

- E como fica o método areaCasa na memória?
- Ao ser chamado (ou invocado) em main, será separada uma região na memória para esse método
  - Contendo todas suas variáveis internas (chamadas de locais)
  - Contendo todos seus parâmetros

```
/*
    Calcula a área da casa

    Parâmetros:
        lateral - comprimento da lateral da cabana
        cquarto - lateral maior do quarto
*/
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

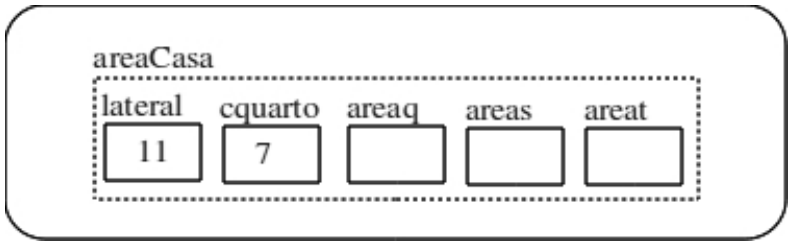
Parâmetros

- E como fica o método areaCasa na memória?
- Ao ser chamado (ou invocado) em main, será separada uma região na memória para esse método
  - Contendo todas suas variáveis internas (chamadas de locais)
  - Contendo todos seus parâmetros
  - E copiando-se os valores de entrada para dentro dos parâmetros

```
/*
    Calcula a área da casa

    Parâmetros:
        lateral - comprimento da lateral da cabana
        cquarto - lateral maior do quarto
*/
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Parâmetros

- E como fica o método areaCasa na memória?
- Ao ser chamado (ou invocado) em main, será separada uma região na memória para esse método
  - Contendo todas suas variáveis internas (chamadas de locais)
  - Contendo todos seus parâmetros
  - E copiando-se os valores de entrada para dentro dos parâmetros




```
/*
    Calcula a área da casa

    Parâmetros:
        lateral - comprimento da lateral da cabana
        cquarto - lateral maior do quarto
*/
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}
```

Métodos e Memória

- Considerando o programa como um todo, como agirá a memória?

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

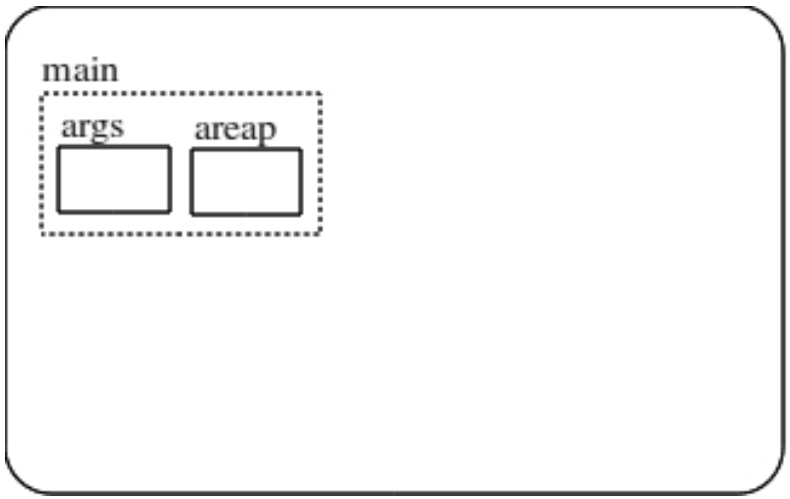
- Considerando o programa como um todo, como agirá a memória?
- Ao iniciar main, será alocado espaço para suas variáveis e parâmetros:

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

- Considerando o programa como um todo, como agirá a memória?
- Ao iniciar main, será alocado espaço para suas variáveis e parâmetros:




```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

- Então areaCasa(11,7) é executada:

```
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

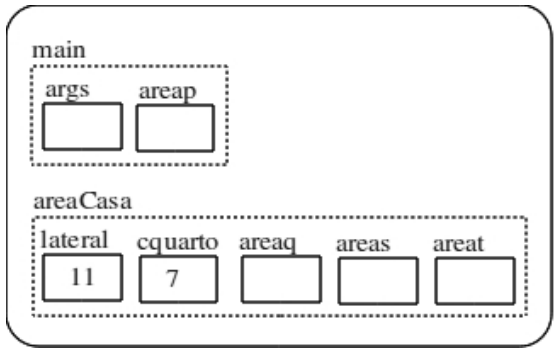
Métodos e Memória

- Então areaCasa(11,7) é executada:
  - E o mesmo processo ocorre

```
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

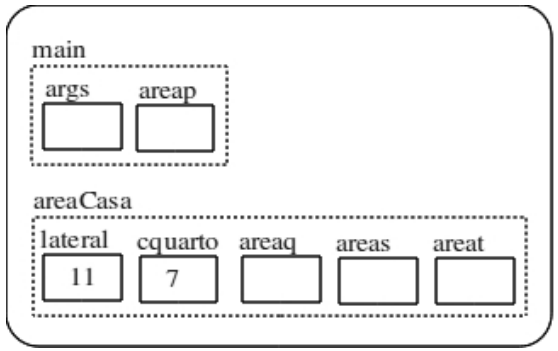
- Então areaCasa(11,7) é executada:
  - E o mesmo processo ocorre




```
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

- Então areaCasa(11,7) é executada:
  - E o mesmo processo ocorre

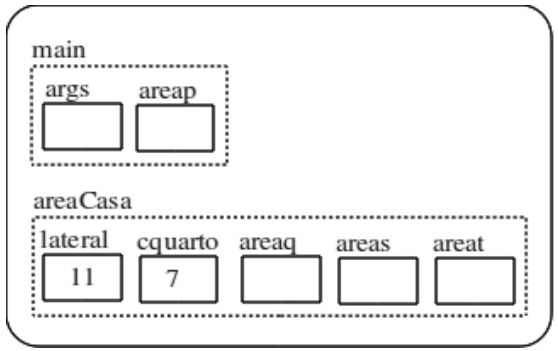


- Logo antes do final da execução, a configuração é a seguinte:

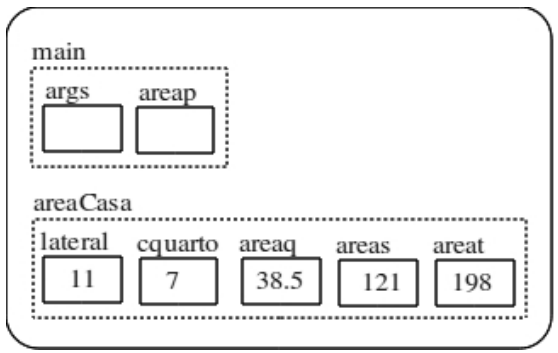
```
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

- Então areaCasa(11,7) é executada:
  - E o mesmo processo ocorre



- Logo antes do final da execução, a configuração é a seguinte:




```
static void areaCasa(float lateral, float cquarto) {
    float areaq; // área do quarto
    float areas; // área da sala
    float areat; // área total

    System.out.println("Cálculo da área da casa");
    // cálculo da área da sala
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    // cálculo da área do banheiro
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    // cálculo da área do quarto
    System.out.println("A área do quarto é "+areaq);
    // cálculo da área total
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

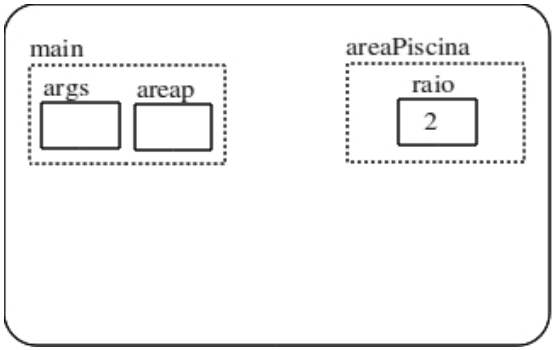
Métodos e Memória

- Ao terminar o método, a memória é limpa, e areaPiscina é rodada:

```
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

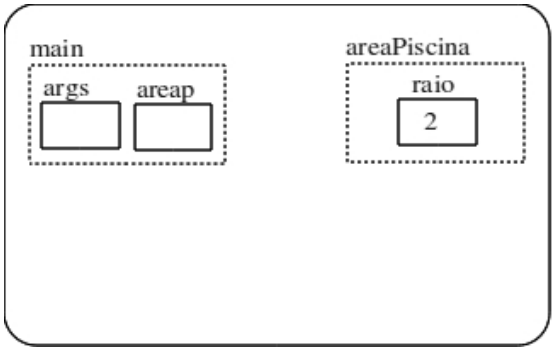
- Ao terminar o método, a memória é limpa, e areaPiscina é rodada:




```
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

Métodos e Memória

- Ao terminar o método, a memória é limpa, e areaPiscina é rodada:

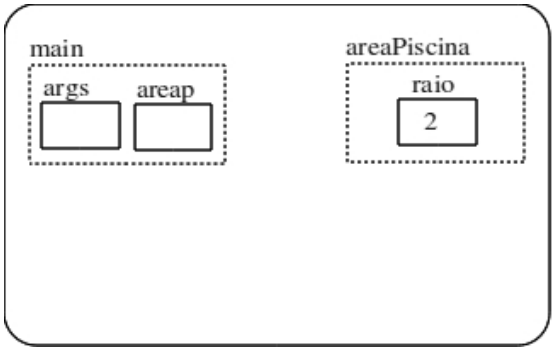



```
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

- Ao terminar o método, a memória é limpa, e o resultado é armazenado em *areap*:

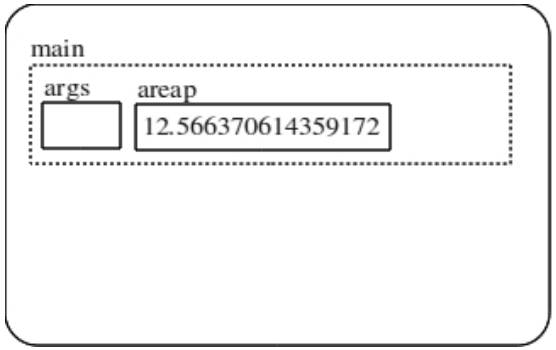
Métodos e Memória

- Ao terminar o método, a memória é limpa, e areaPiscina é rodada:



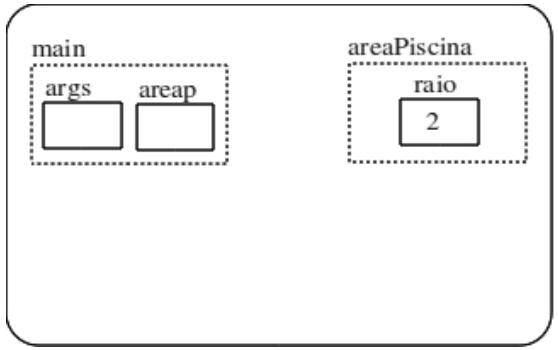

```
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

- Ao terminar o método, a memória é limpa, e o resultado é armazenado em *areap*:



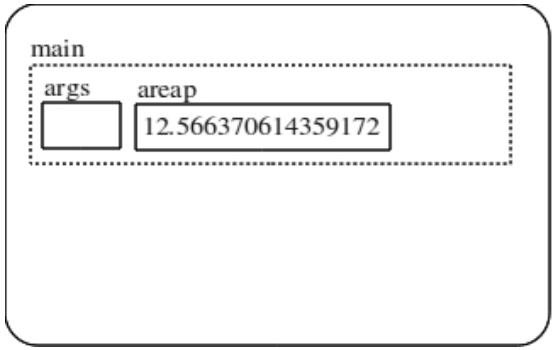
Métodos e Memória

- Ao terminar o método, a memória é limpa, e areaPiscina é rodada:




```
static double areaPiscina(double raio) {
    return Math.PI * Math.pow(raio,2);
}

public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "+areap);
}
```

- Ao terminar o método, a memória é limpa, e o resultado é armazenado em *areap*:



- Finalmente, quando *main* terminar, também ela será removida da memória

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?
  - Clareza:

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?
  - Clareza:
    - ★ Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?
  - Clareza:
    - ★ Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes
  - Portabilidade:

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?
  - Clareza:
    - ★ Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes
  - Portabilidade:
    - ★ Se precisarmos, em outro programa, usar a mesma subrotina, ela já está separada

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?
  - Clareza:
    - ★ Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes
  - Portabilidade:
    - ★ Se precisarmos, em outro programa, usar a mesma subrotina, ela já está separada
  - Melhor uso da memória

Métodos e Memória

- Repare que toda vez que um método termina ele libera a memória que ocupava
- Então... qual a utilidade de criarmos nossos próprios métodos?
  - Clareza:
    - ★ Ao olharmos o corpo do programa, vemos claramente o que é feito, sem nos preocuparmos com detalhes
  - Portabilidade:
    - ★ Se precisarmos, em outro programa, usar a mesma subrotina, ela já está separada
  - Melhor uso da memória
    - ★ As variáveis relevantes ao sub-problema (subrotina) ocupam a memória apenas quando da solução desse sub-problema

Métodos e Memória

- Revendo os métodos *main*:

```
public static void main(String[] args) {
    float lateral = 11;
    float cquarto = 7;
    float areaq;
    float areas;
    float areat;
    double raio = 2;
    double areap;

    System.out.println("Cálculo da área da casa");
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    System.out.println("A área do quarto é "+areaq);
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
    areap = Math.PI * Math.pow(raio,2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
```

Métodos e Memória

- Revendo os métodos *main*:

```
public static void main(String[] args) {
    float lateral = 11;
    float cquarto = 7;
    float areaq;
    float areas;
    float areat;
    double raio = 2;
    double areap;

    System.out.println("Cálculo da área da casa");
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    System.out.println("A área do quarto é "+areaq);
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
    areap = Math.PI * Math.pow(raio,2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
```

- Uso de memória no início do *main*:

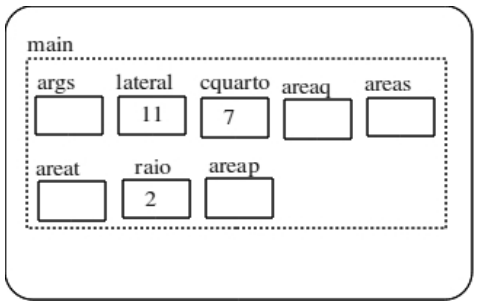
Métodos e Memória

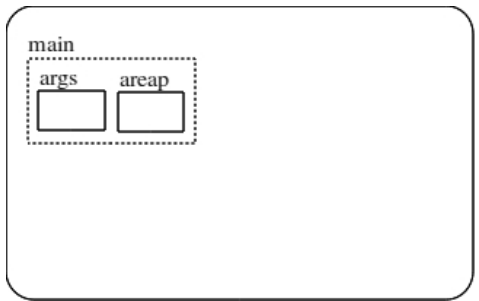
- Revendo os métodos *main*:

```
public static void main(String[] args) {
    float lateral = 11;
    float cquarto = 7;
    float areaq;
    float areas;
    float areat;
    double raio = 2;
    double areap;

    System.out.println("Cálculo da área da casa");
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    System.out.println("A área do quarto é "+areaq);
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
    areap = Math.PI * Math.pow(raio,2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
```

- Uso de memória no início do *main*:





Métodos e Memória

- Revendo os métodos *main*:

```
public static void main(String[] args) {
    float lateral = 11;
    float cquarto = 7;
    float areaq;
    float areas;
    float areat;
    double raio = 2;
    double areap;

    System.out.println("Cálculo da área da casa");
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    System.out.println("A área do quarto é "+areaq);
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
    areap = Math.PI * Math.pow(raio,2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
```

Métodos e Memória

- Revendo os métodos *main*:

```
public static void main(String[] args) {
    float lateral = 11;
    float cquarto = 7;
    float areaq;
    float areas;
    float areat;
    double raio = 2;
    double areap;

    System.out.println("Cálculo da área da casa");
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    System.out.println("A área do quarto é "+areaq);
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
    areap = Math.PI * Math.pow(raio,2);
    System.out.println("A área da piscina é "+areap);
}
```

- Uso de memória no fim do *main*:

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
```

Métodos e Memória

- Revendo os métodos *main*:

```
public static void main(String[] args) {
    float lateral = 11;
    float cquarto = 7;
    float areaq;
    float areas;
    float areat;
    double raio = 2;
    double areap;

    System.out.println("Cálculo da área da casa");
    areas = lateral*lateral;
    System.out.println("A área da sala é "+areas);
    areaq = cquarto*(lateral/2);
    System.out.println("A área do banheiro é "+areaq);
    System.out.println("A área do quarto é "+areaq);
    areat = areas + 2*areaq;
    System.out.println("A área total é " + areat);
    areap = Math.PI * Math.pow(raio,2);
    System.out.println("A área da piscina é "+areap);
}
```

```
public static void main(String[] args) {
    double areap; // área da piscina

    areaCasa(11,7);

    areap = areaPiscina(2);
    System.out.println("A área da piscina é "
                       +areap);
}
```

- Uso de memória no fim do *main*: