Métodos

```
TipDeRetorno nomeDoMehtodo(
TipoDoArg1 arg1, TipoDoArg2
arg2, ...){
 TipoDeRetorno valorDeRetorno;
 ...
 return valorDeRetorno;
}
```

Se o método não utiliza nenhum argumento, parêntese vazia, devem ser incluídas na declaração.

Valor de retorno e argumentos

Método sem retorno:

```
void nomeDoMehtodo( TipoDoArg1 arg1,
TipoDoArg2 arg2, ... ){
    ...
}
```

Valor e retorno de argumentos

Exemplo (métodos da classe Math do pacote lang)

```
public static int min( int a, int b ){
     ... // retorna o menor dos 2 inteiros
  a e b
}
```

Métodos estáticos

```
package algumPacote;
public class NomeDaClasse{
  public int mehtodo1( ){
       ... // acessível por qualquer classe
  }
  protected int mehtodo2( ){
       ... // acessível por esta classe e suas
  subclasses
  }
  int mehtodo3( ){
       ... // acessível pelas classes deste pacote
  }
  private int mehtodo4( ){
       ... // acessível por esta classe
  }
}
```

Restrições de Acesso

```
nomeDoObjeto.mehtodo1( );

nomeDoObjeto.mehtodo2( var1, var2 );

Resultado resultado = objeto.mehtodo3( );

int x = Math.min( a, b );
```

Chamando método de outro objeto

Como criar um objeto

```
NomeDaClasse nomeDoObjeto = new NomeDaClasse( );


class NomeDaClasse{
  NomeDaClasse( ){
      ... // comandos executados na criação do
  objeto
  }
}
```

Pode-se usar “public” na criação (pra classes públicas) e também pode-se passar argumentos na criação do objeto.

- Crie uma classe com um método que verifica se o número é primo ou não;

- Utilize este método para que seu aplicativo construa um Crivo de Eratóstenes*, onde o limite será passado pelo usuário.

* Algoritmo para encontrar números primos até um certo valor limite.

Prática