**GUIA BÁSICO DA LINGUAGEM DE PROGRAMAÇÃO**



Sumário

[**1.** **Literais básicos** 3](#_Toc39608501)

[1.1 Characters 3](#_Toc39608502)

[1.2 Strings 3](#_Toc39608503)

[2. A terminologia básica 4](#_Toc39608504)

[2.1 Componentes do programa 5](#_Toc39608505)

[2.2Palavras-chave 5](#_Toc39608506)

[3. Impressão de dados 11](#_Toc39608507)

[3.1 Exibindo texto usando println () e print () 11](#_Toc39608508)

[REFERENCIAS 12](#_Toc39608509)

~~Ponto importante é a independência de uma plataforma. o Java possui um~~**~~coletor de lixo~~**~~que~~**~~limpa automaticamente a memória~~**~~de objetos não utilizados durante o tempo de execução.~~

  Java é primariamente uma linguagem imperativa baseada no conceito orientado a objetos: quase toda parte de um programa é um objeto. Portanto, um programa em si pode ser considerado um conjunto de objetos em interação.

1. **Literais básicos**

Aqui estão alguns exemplos de literais número inteiro válido separados por vírgulas: 0, 1, 2, 10, 11, 100.

Se um valor inteiro contém uma grande quantidade de dígitos, podemos adicionar sublinhados para dividir o dígito em blocos para aumentar a legibilidade: 1\_000\_000. É mais legível que o mesmo valor escrito como 1000000.

## Characters

Um único caractere pode representar um dígito, uma letra ou outro símbolo. Para escrever um caractere deve-se usar aspas simples da seguinte forma: 'A', 'B', 'C', 'x', 'y', 'z', '0', '1', '2', '9'. Literais de caracteres podem representar símbolos de um alfabeto, dígitos de '0'para '9', espaços em branco ( ' ') ou outros caracteres ou símbolos ( '$').

Não confunda caracteres que representam números (por exemplo '9') com os próprios números (por exemplo 9).

Um caractere **não** pode incluir dois ou mais dígitos ou letras porque representa apenas um **único** símbolo. Os dois exemplos que se seguem são **incorretos** : 'abc', '543'. Esses literais contêm muitos caracteres.

## 1.2 Strings

Uma sequência é uma sequência de qualquer caractere individual. Strings representam informações de texto, como texto de publicidade, endereço de uma página da web ou login em um site.

Para escrever uma string, usamos aspas duplas em vez de simples. Aqui estão alguns exemplos válidos: "text", "I want to know Java", "123456", "e-mail@gmail.com". Uma sequência que consiste em um único caractere como  "A" também é uma sequência válida, mas não a confunda com o 'A' caractere.

Portanto strings podem incluir letras, dígitos, espaços em branco e outros caracteres.

**Resumindo :**

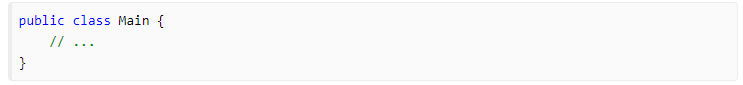
* 123 é um número inteiro, "123"é uma string;
* 'A' é um personagem, "A" é uma string;
* '1'é um caractere, 1 é um número inteiro.

## A terminologia básica

* **Programa** - uma sequência de instruções (chamadas instruções), executadas uma após a outra de maneira previsível. O fluxo sequencial é a sequência mais comum e direta de instruções, na qual as instruções são executadas na ordem em que são gravadas - de cima para baixo de maneira sequencial;
* **Instrução** - uma única ação (como imprimir um texto) terminada por ponto e vírgula ( ;);
* **Bloco** - um grupo de zero, uma ou mais instruções incluídas por um par de chaves {...};
* **Método** - uma sequência de instruções que representa uma operação de alto nível (também conhecida como subprograma ou procedimento).
* **Sintaxe** - um conjunto de regras que definem como um programa precisa ser escrito para ser válido; Java tem sua própria sintaxe específica que aprenderemos;
* **Palavra-chave** - uma palavra que tem um significado especial na linguagem de programação ( public, classe muitos outros). Essas palavras não podem ser usadas como nomes de variáveis ​​para o seu próprio programa;
* **Identificador ou nome** - uma palavra que se refere a algo em um programa (como uma variável ou um nome de função);
* **Comentário** - uma explicação textual do que o código faz. Os comentários em Java começam com //.
* Espaço em **branco** - todos os caracteres que não são visíveis (espaço, tabulação, nova linha etc.).

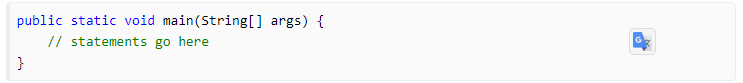
## Componentes do programa

**A classe pública.**É a unidade básica de um programa. Todo programa Java deve ter pelo menos uma classe. A definição de uma classe consiste na “class” palavra - chave seguida pelo nome da classe. Uma classe pode ter qualquer nome, como App, Main, ou Program, mas não deve começar com um dígito. Um conjunto de chaves {...}inclui o corpo de uma classe.



O texto acima //é apenas um comentário, não parte do programa.

* **O método principal.**Para tornar o programa executável, colocamos um método nomeado main dentro de uma classe. É o ponto de entrada para um programa Java. Novamente, os colchetes {...}incluem o corpo do método, que contém instruções de programação.



O elemento String[] args representa uma sequência de argumentos transmitidos ao programa pelo mundo exterior.

## 2.2Palavras-chave

Um programa Java simples consiste em muitos elementos, incluindo **palavras-chave** que fazem parte da linguagem. No total, o Java fornece mais de 50 palavras-chave segue elas abaixo:

[**abstract**](https://en.wikipedia.org/wiki/Abstract_type) **-** Um método sem definição deve ser declarado como abstrato e a classe que o contém deve ser declarada como abstrata. Classes abstratas não podem ser instanciadas. Métodos abstratos devem ser implementados nas subclasses. A palavra-chave abstract não pode ser usada com variáveis ou construtores. Observe que uma classe abstrata não é obrigada a ter um método abstrato.

[**assert**](https://en.wikipedia.org/wiki/Assertion_(software_development)) **-** descreve um predicado (uma declaração verdadeiro-falso) colocado em um programa Java para indicar que o desenvolvedor pensa que o predicado é sempre verdadeiro naquele local. Se uma asserção for avaliada como falsa no tempo de execução, ocorrerá uma falha de asserção, que normalmente causa a interrupção da execução. Opcionalmente, habilite pelo método ClassLoader.

[**boolean**](https://en.wikipedia.org/wiki/Boolean_data_type) **-** Define uma variável booleana apenas para os valores "true" ou "false". Por padrão, o valor do tipo primitivo booleano é falso. Essa palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo [boolean](https://en.wikipedia.org/wiki/Boolean_data_type" \o "Boolean data type).

[**break**](https://en.wikipedia.org/wiki/Break_statement) **-** Usado para finalizar a execução no corpo do loop atual.

[**byte**](https://en.wikipedia.org/wiki/Byte) **–** A palavra - chave byte é usada para declarar um campo que pode conter um inteiro de complemento de dois assinados de 8 bits. Esta palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo byte.

[**case**](https://en.wikipedia.org/wiki/Switch_statement) **-** Uma declaração no [switch](https://en.wikipedia.org/wiki/List_of_Java_keywords#switch) bloco pode ser rotulada com um ou mais [case](https://en.wikipedia.org/wiki/List_of_Java_keywords#case)ou [default](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "default)rótulos. A [switch](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "switch)instrução avalia sua expressão e, em seguida, executa todas as instruções que seguem o [case](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "case)rótulo correspondente ; veja [switch](https://en.wikipedia.org/wiki/List_of_Java_keywords#switch).

[**catch**](https://en.wikipedia.org/wiki/Exception_handling_syntax#Java) **-** Usado em conjunto com um try bloco e um finally bloco opcional . As instruções no catchbloco especificam o que fazer se um tipo específico de exceção for lançado pelo trybloco.

[**char**](https://en.wikipedia.org/wiki/Character_(computing))**-** Define uma variável de caractere capaz de conter qualquer caractere do conjunto de caracteres do arquivo de origem java.

[**class**](https://en.wikipedia.org/wiki/Class_(computer_science)#Java) **-** Um tipo que define a implementação de um tipo específico de objeto. Uma definição de classe define [campos de](https://en.wikipedia.org/wiki/Field_(computer_science))[instância](https://en.wikipedia.org/wiki/Object_(computer_science)) e classe , [métodos](https://en.wikipedia.org/wiki/Method_(computer_science)) e [classes internas](https://en.wikipedia.org/wiki/Inner_class) , além de especificar as [interfaces que](https://en.wikipedia.org/wiki/Interface_(computer_science)) a classe implementa e a [superclasse](https://en.wikipedia.org/wiki/Superclass_(computer_science)) imediata da classe. Se a superclasse não for especificada explicitamente, a superclasse é implicitamente . A palavra-chave class também pode ser usada no formato Class **.class** para obter um objeto Class sem precisar de uma instância dessa classe. Por exemplo, **String.class** pode ser usado em vez de fazer **nova String (). GetClass ()** .[Object](https://docs.oracle.com/javase/10/docs/api/java/lang/Object.html)

[**continue**](https://en.wikipedia.org/wiki/Continue_(Java)) **-**Usado para retomar a execução do programa no final do corpo do loop atual. Se seguido por um rótulo, continueretoma a execução no final do corpo do loop rotulado.

**Default -** A palavra-chave default pode opcionalmente ser usado em uma [instrução switch](https://en.wikipedia.org/wiki/Switch_statement) para rotular um bloco de instruções a ser executado se não case corresponde ao valor especificado; veja [*switch*](https://en.wikipedia.org/wiki/List_of_Java_keywords#switch). Como alternativa, a palavra - chave default também pode ser usada para declarar valores padrão em uma [anotação Java](https://en.wikipedia.org/wiki/Java_annotation) . A partir do Java 8, a defaultpalavra - chave pode ser usada para permitir que uma interface forneça uma implementação de um método.

[**do**](https://en.wikipedia.org/wiki/Do_while_loop) **-** A palavra - chave do é usada em conjunto com [while](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "while) para criar um [loop do-while](https://en.wikipedia.org/wiki/Do-while_loop) , que executa um bloco de instruções associado ao loop e testa uma expressão booleana associada ao while. Se a expressão for avaliada como true, o bloco será executado novamente; isso continua até que a expressão seja avaliada como false.

[**double**](https://en.wikipedia.org/wiki/Double_precision) **-** A palavra - chave double é usada para declarar uma variável que pode conter um [número de ponto flutuante](https://en.wikipedia.org/wiki/Floating-point_number)[IEEE 754 de](https://en.wikipedia.org/wiki/IEEE_754)[precisão dupla de](https://en.wikipedia.org/wiki/Double_precision) 64 bits . Esta palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo .

[**else**](https://en.wikipedia.org/wiki/Conditional_(programming)) **–** A palavra - chave else é usada em conjunto com [if](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "if)para criar uma [instrução if-else](https://en.wikipedia.org/wiki/Conditional_(programming)) , que testa uma [expressão booleana](https://en.wikipedia.org/wiki/Boolean_expression) ; se a expressão é avaliada como true, o bloco de instruções associado a ifé avaliado; se for avaliado false, o bloco de instruções associado ao elseserá avaliado.

[**enum**](https://en.wikipedia.org/wiki/Enumerated_type) **-**Uma palavra-chave Java usada para declarar um [tipo enumerado](https://en.wikipedia.org/wiki/Enumerated_type) . Enumerações estendem a classe base [Enum](https://docs.oracle.com/javase/10/docs/api/java/lang/Enum.html).

[**extends**](https://en.wikipedia.org/wiki/Inheritance_(object-oriented_programming)) **-** Usado em uma declaração de classe para especificar a superclasse; usado em uma declaração de interface para especificar uma ou mais superinterfaces. A classe X estende a classe Y para adicionar funcionalidade, adicionando campos ou métodos à classe Y ou substituindo métodos da classe Y. Uma interface Z estende uma ou mais interfaces adicionando métodos. A classe X é considerada uma subclasse da classe Y; A interface Z é considerada uma sub-interface das interfaces que ela se estende. Também usado para especificar um limite superior em um parâmetro de tipo em Genéricos.

[**final**](https://en.wikipedia.org/wiki/Final_(Java)) **-**Define uma entidade uma vez que não possa ser alterada nem derivada posteriormente. Mais especificamente: uma classe final não pode ser subclassificada, um método final não pode ser substituído e uma variável final pode ocorrer no máximo uma vez como uma expressão à esquerda em um comando executado. Todos os métodos em uma classe final são implicitamente final.

[**finally**](https://en.wikipedia.org/wiki/Exception_handling_syntax#Java) **-**Usado para definir um bloco de instruções para um bloco definido anteriormente pela palavra - chave  try. O finallybloco é executado após a execução sair do trybloco e de quaisquer catchcláusulas associadas, independentemente de uma exceção ter sido lançada ou capturada ou o método de execução deixado no meio dos blocos tryou catchusando a returnpalavra - chave.

[**float**](https://en.wikipedia.org/wiki/Single_precision) **–** A palavra - chave  float é usada para declarar uma variável que pode conter um número de ponto flutuante IEEE 754 de [precisão única de](https://en.wikipedia.org/wiki/Single_precision) 32 bits . Esta palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo float.

[**for**](https://en.wikipedia.org/wiki/For_loop) **-**A palavra - chave  for é usada para criar um [loop for](https://en.wikipedia.org/wiki/For_loop) , que especifica uma inicialização variável, uma [expressão booleana](https://en.wikipedia.org/wiki/Boolean_expression) e uma incrementação. A inicialização da variável é realizada primeiro e, em seguida, a expressão booleana é avaliada. Se a expressão for avaliada como true, o bloco de instruções associado ao loop será executado e, em seguida, o incremento será executado. A expressão booleana é então avaliada novamente; isso continua até que a expressão seja avaliada como false. A partir do [J2SE 5.0](https://en.wikipedia.org/wiki/J2SE_5.0) , a forpalavra - chave também pode ser usada para criar o chamado " [aprimorado para loop](https://en.wikipedia.org/wiki/Foreach) ", [[15]](https://en.wikipedia.org/wiki/List_of_Java_keywords#cite_note-FOOTNOTEFlanagan200550-54-15) que especifica uma [matriz](https://en.wikipedia.org/wiki/Array_data_type) ou [Iterable](https://docs.oracle.com/javase/10/docs/api/java/lang/Iterable.html)objeto; cada iteração do loop executa o bloco de instruções associado usando um elemento diferente na matriz ou Iterable.

[**if**](https://en.wikipedia.org/wiki/If_statement) **-**A palavra-chave if é usada para criar uma [instrução if](https://en.wikipedia.org/wiki/If_statement) , que testa uma [expressão booleana](https://en.wikipedia.org/wiki/Boolean_expression) ; se a expressão for avaliada como true, o bloco de instruções associado à instrução if será executado. Essa palavra-chave também pode ser usada para criar uma [instrução if-else](https://en.wikipedia.org/wiki/Conditional_(programming)).

**Implements -** Incluído em uma declaração de classe para especificar uma ou mais [interfaces](https://en.wikipedia.org/wiki/Interface_(Java)) implementadas pela classe atual. Uma classe herda os tipos e métodos abstratos declarados pelas interfaces.

**Import -**Usado no início de um [arquivo de origem](https://en.wikipedia.org/wiki/Source_file) para especificar classes ou [pacotes Java](https://en.wikipedia.org/wiki/Java_package) inteiros a serem consultados posteriormente, sem incluir seus nomes de pacotes na referência. Desde o J2SE 5.0, as importinstruções podem importar staticmembros de uma classe.

[**int**](https://en.wikipedia.org/wiki/Integer_(computer_science))**-** A palavra - chave  int é usada para declarar uma variável que pode conter um número inteiro de complemento assinado por dois de 32 bits. Esta palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo int.

[**interface**](https://en.wikipedia.org/wiki/Interface_(Java)) **-** Usado para declarar um tipo especial de classe que contém apenas métodos abstratos ou padrão, static finalcampos constant ( ) e staticinterfaces. Posteriormente, ele pode ser implementado por classes que declaram a interface com a palavra - chave  implements. Como a herança múltipla não é permitida em Java, as interfaces são usadas para contorná-la. Uma interface pode ser definida dentro de outra interface.

[**long**](https://en.wikipedia.org/wiki/Long_integer)**-** A palavra - chave  long é usada para declarar uma variável que pode conter um número inteiro de complemento assinado por dois de 64 bits. Esta palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo long.

[**native**](https://en.wikipedia.org/wiki/Java_Native_Interface) **-** Usado nas declarações do método para especificar que o método não é implementado no mesmo arquivo de origem Java, mas em outro idioma.

[**new**](https://en.wikipedia.org/wiki/Object_lifetime#Java)**-** Usado para criar uma instância de um objeto de classe ou matriz. O uso de palavras-chave para esse fim não é completamente necessário (como exemplificado por [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)) ), embora atenda a dois propósitos: permite a existência de um espaço de nome diferente para métodos e nomes de classes, define estaticamente e localmente que um objeto novo é realmente criado e que tipo de tempo de execução é (sem dúvida introduzindo dependência no código).

[**package**](https://en.wikipedia.org/wiki/Java_package) **-** Pacote Java é um grupo de classes e interfaces semelhantes. Pacotes são declarados com a palavra - chave  package.

[**private**](https://en.wikibooks.org/wiki/Java_Programming/Classes,_Objects_and_Types) **–** A palavra - chave  private é usada na declaração de um método, campo ou classe interna; membros privados só podem ser acessados ​​por outros membros de sua própria classe.

[**protected**](https://en.wikibooks.org/wiki/Java_Programming/Classes,_Objects_and_Types)**-** A palavra - chave  protected é usada na declaração de um método, campo ou classe interna; membros protegidos só podem ser acessados ​​por membros de sua própria classe, [subclasses](https://en.wikipedia.org/wiki/Inheritance_(object-oriented_programming)) ou classes dessa classe do mesmo [pacote](https://en.wikipedia.org/wiki/Java_package) .

[**public**](https://en.wikibooks.org/wiki/Java_Programming/Classes,_Objects_and_Types) **–** A palavra - chave  public é usada na declaração de uma classe, método ou campo; classes públicas, métodos e campos podem ser acessados ​​pelos membros de qualquer classe.

[**return**](https://en.wikipedia.org/wiki/Method_(computer_science)) **-** Usado para concluir a execução de um método. Pode ser seguido por um valor exigido pela definição do método que é retornada ao chamador.

[**short**](https://en.wikipedia.org/wiki/Short_integer) **-**A palavra - chave  short é usada para declarar um campo que pode conter um inteiro de complemento de dois assinados de 16 bits. Esta palavra-chave também é usada para declarar que um método retorna um valor do tipo primitivo short.

[**static**](https://en.wikipedia.org/wiki/Static_variable) **-** Usado para declarar um campo, método ou classe interna como um campo de classe. As classes mantêm uma cópia dos campos da classe, independentemente de quantas instâncias existem nessa classe. Static também é usado para definir um método como um método de classe. Os métodos de classe são [vinculados](https://en.wikipedia.org/wiki/Name_binding) à classe em vez de a uma instância específica e só podem operar em campos de classe. (Classes e interfaces declaradas como staticmembros de outra classe ou interface são, na verdade, classes de nível superior e *não* são classes internas.

[**strictfp**](https://en.wikipedia.org/wiki/Strictfp)**-**Uma palavra-chave Java usada para restringir a precisão e o arredondamento dos cálculos de ponto flutuante para garantir a portabilidade.

[**super**](https://en.wikipedia.org/wiki/Inheritance_(object-oriented_programming))**-** Herança usada basicamente para obter ligação dinâmica ou polimorfismo em tempo de execução em java. Usado para acessar membros de uma classe herdada pela classe em que ela aparece. Permite que uma subclasse acesse métodos [substituídos](https://en.wikipedia.org/wiki/Method_overriding_(programming)) e membros ocultos de sua superclasse. A palavra - chave  super também é usada para encaminhar uma chamada de um construtor para um construtor na superclasse. Também usado para especificar um limite inferior em um parâmetro de tipo em Genéricos.

[**switch**](https://en.wikipedia.org/wiki/Switch_statement) **-**A  palavra - chaveswitch é usada em conjunto com [case](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "case)e [default](https://en.wikipedia.org/wiki/List_of_Java_keywords" \l "default)para criar uma [instrução switch](https://en.wikipedia.org/wiki/Switch_statement) , que avalia uma variável, corresponde seu valor a uma específica casee executa o bloco de instruções associado a ela case. Se nenhum case corresponder ao valor, o bloco opcional rotulado por defaulté executado, se incluído.

[**synchronized**](https://en.wikipedia.org/wiki/Mutual_exclusion) **-**Usado na declaração de um método ou bloco de código para adquirir o bloqueio [mutex](https://en.wikipedia.org/wiki/Mutex" \o "Mutex) para um objeto enquanto o [thread](https://en.wikipedia.org/wiki/Thread_(computer_science)) atual executa o código. Para métodos estáticos, o objeto bloqueado é da classe Class. Garante que, no máximo, um encadeamento de cada vez, operando no mesmo objeto, execute esse código. O bloqueio mutex é liberado automaticamente quando a execução sai do código sincronizado. Campos, classes e interfaces não podem ser declarados como *sincronizados*.

[**this**](https://en.wikipedia.org/wiki/This_(Java)) **-**Usado para representar uma instância da classe em que aparece. thispode ser usado para acessar os membros da classe e como uma referência à instância atual. A palavra - chave  this também é usada para encaminhar uma chamada de um construtor em uma classe para outro construtor na mesma classe.

[**throw**](https://en.wikipedia.org/wiki/Exception_handling_syntax#Java) **-** Faz com que a instância de exceção declarada seja lançada. Isso faz com que a execução continue com o primeiro manipulador de exceção em anexo declarado pela catchpalavra-chave para manipular um tipo de exceção compatível com a atribuição. Se nenhum manipulador de exceção for encontrado no método atual, o método retornará e o processo será repetido no método de chamada. Se nenhum manipulador de exceções for encontrado em qualquer chamada de método na pilha, a exceção será passada para o manipulador de exceções não capturado do encadeamento.

[**throws**](https://en.wikipedia.org/wiki/Exception_handling_syntax#Java) **-**Usado nas declarações do método para especificar quais exceções não são tratadas no método, mas passadas para o próximo nível mais alto do programa. Todas as exceções não detectadas em um método que não são instâncias RuntimeExceptiondevem ser declaradas usando a palavra - chave  throws.

[**transient**](https://en.wikipedia.org/wiki/Transient_(computer_programming)) **-** Declara que um campo de instância não faz parte do formulário [serializado](https://en.wikipedia.org/wiki/Serialization) padrão de um objeto. Quando um objeto é serializado, apenas os valores de seus campos de instâncias não transitórios são incluídos na representação serial padrão. Quando um objeto é desserializado, os campos transitórios são inicializados apenas com o valor padrão. Se o formulário padrão não for usado, por exemplo, quando uma tabela *serialPersistentFields* for declarada na hierarquia de classes, todas as palavra - chave  transient serão ignoradas.

[**try**](https://en.wikipedia.org/wiki/Exception_handling_syntax#Java) **-** Define um bloco de instruções que possuem manipulação de exceção. Se uma exceção for lançada dentro do bloco try, um bloco catch opcional poderá manipular os tipos de exceção declarados. Além disso, finallypode ser declarado um bloco opcional que será executado quando a execução sair do bloco try e das cláusulas catch, independentemente de uma exceção ser lançada ou não. Um bloco try deve ter pelo menos uma cláusula catch ou um bloco finally.

[**void**](https://en.wikipedia.org/wiki/Void_return_type) **-** A palavra - chave  void é usada para declarar que um método não retorna nenhum valor.

[**volatile**](https://en.wikipedia.org/wiki/Volatile_variable) **-** Utilizado em declarações de campo para garantir a visibilidade das alterações nas variáveis ​​nos threads. Toda leitura de uma variável volátil será lida na memória principal, e não no cache da CPU, e todas as gravações em uma variável volátil serão gravadas na memória principal, e não apenas no cache da CPU. Portanto, métodos, classes e interfaces não podem ser declarados *voláteis*, nem variáveis ​​ou parâmetros locais.

[**while**](https://en.wikipedia.org/wiki/Do_while_loop) **–** A palavra - chave   while é usada para criar um [loop while](https://en.wikipedia.org/wiki/While_loop) , que testa uma [expressão booleana](https://en.wikipedia.org/wiki/Boolean_expression) e executa o bloco de instruções associado ao loop, se a expressão for avaliada true; isso continua até que a expressão seja avaliada como false. Essa palavra-chave também pode ser usada para criar um [loop do-while](https://en.wikipedia.org/wiki/Do-while_loop).

## Impressão de dados

## 3.1 Exibindo texto usando println () e print ()

**A saída padrão** é um receptor para o qual um programa pode enviar informações (texto). É suportado por todos os sistemas operacionais comuns. Java fornece um objeto especial System.out para trabalhar com a saída padrão. Nós geralmente o usamos para imprimir alguma coisa.

O  método println exibe a sequência passada, seguida por uma nova linha na tela ( **linha de impressão** ). Como exemplo, o seguinte fragmento de código imprime quatro linhas.



## REFERENCIAS

<https://en.wikipedia.org/wiki/List_of_Java_keywords>

<https://hyperskill.org/learn/step/8977>

(Falta deixar bonitinho)