**Java**

**Basic Info**

* Java editions: **Java SE** + **Java EE** + **Java ME** + **JavaFX** (internet related)
* **JVM:** Produces and converts **bytecodes** to native machine code & vice-versa.
* **JRE:** Provides **utilities** and components required for running Java programs.
* **JDK:** **Framework** for Java used for writing, compiling (***javac***) & debugging.
* **JNI:** Java native interface, used to communicate with **other languages**.
* **JCL:** Java class library, contains all Java standard libraries with ***.jar*** extension.

**Some OOPs Concepts**

* **Coupling:** One class depending on another.
* **Cohesion:** Weakly cohesive library **splits** tasks into various **subtasks**.
* **Association:** Relation among objects like **one-to-one**, **one-to-many** etc.
* **Aggregation:** Weakly related objects.
* **Composition:** Strongly related objects.

**Data Types**

* 8 primitive types: ***byte***, ***short***, ***long***, ***int***, ***float***, ***double***, ***boolean***, ***char***.
* Primitive data types are also known as **reference/object** data types.
* **Literal:** Name of an identifier.
* **Identifier:** Any class, variable or method etc.
* **Narrowing:** Converting a high sized data type to lower one.
* **Widening:** ***\*Opposite of narrowing.\****
* **Objects** are stored in **heap** segments.

**Similarities With C/C++**

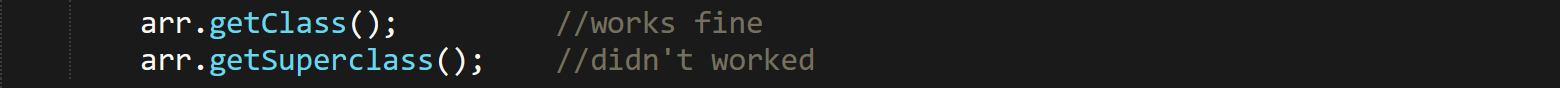
* Escape sequences.
* Default value of variables.
* Bitwise operators.
* Ternary operator.
* Switch cases.
* Multidimensional arrays.
* Behaviour of curly braces.
* Methods can be passed into parameters.

**Arrays**

* Arrays are **objects** in Java.
* Size can be specified by ***int*** or ***short***, not ***long***.
* Array size **can’t** be redefined.







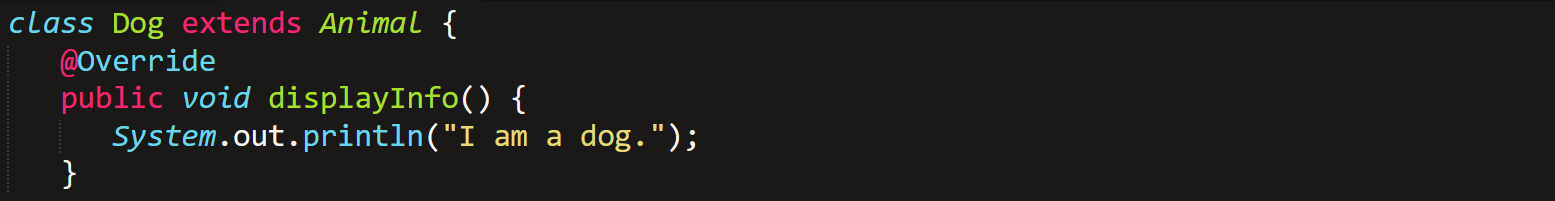
**Special Methods**

* Basic statements like for printing are part of ***java.io*** class, **doesn’t** require import.



**Object Oriented Programming**

* ***super*** keyword is used for referring to any member of the **parent class**.
* There are **3** types of inheritance in Java.
* Constructors **aren’t** inherited in Java.
* ***protected*** is used to let a member be accessible in inherited classes.
* Static and final methods **can’t** be overridden.



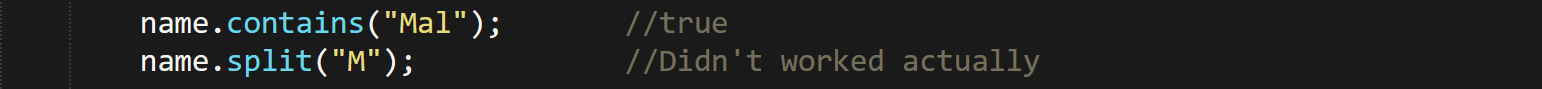
* Overriding is also known as **runtime polymorphism**.
* ***interface*** classes provide the **skeleton blueprint** to use (**doesn’t** uses keyword ***class***).

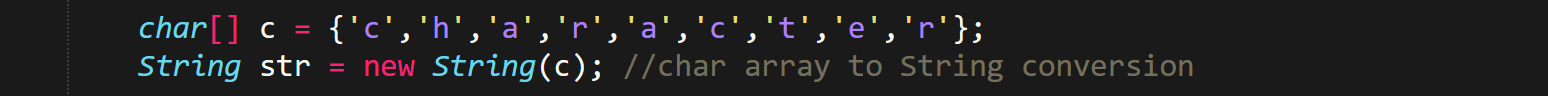
**Abstract Classes**

* **Can’t** produce instances.
* Methods are to be individually declared ***abstract*** (if needed).
* Otherwise, there can be regular methods in ***abstract*** class too.
* Only ***abstract*** classes can have ***abstract*** method.
* There **can’t** be abstract variables.

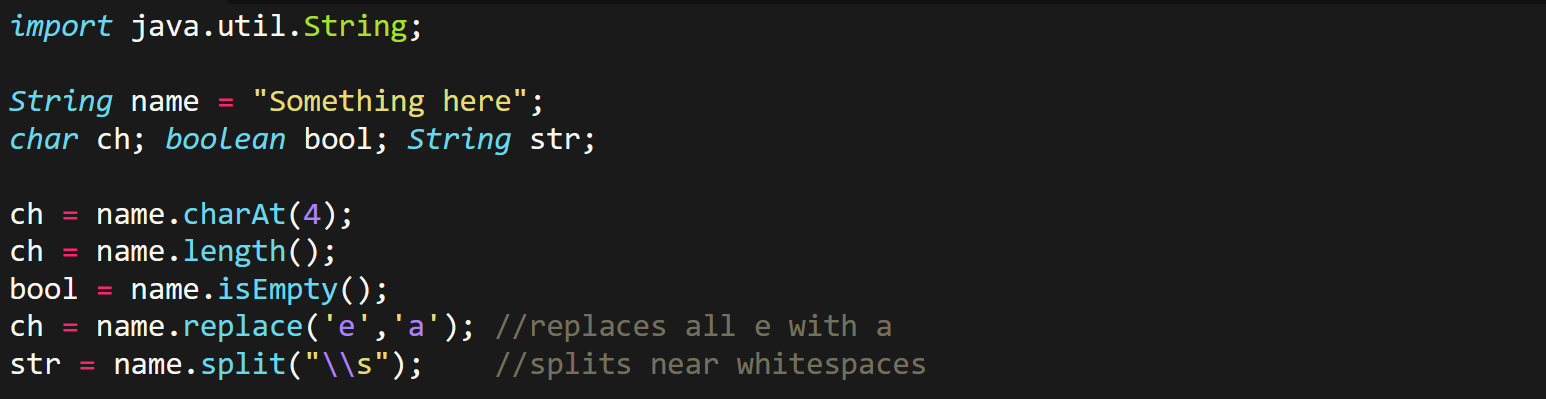
**Strings**

* ***String*** is an object in Java.





* Those using ***java.util.String***;



* **Concatenation** is done using **+** operator.

**Packages**

* **Package** is the folder containing the ***.java*** files you want to import.
* Every ***.java*** files in package must have the line:



* One ***.java*** file in package folder can contain only one ***class***, and that ***class*** must be from **name of the file**.
* ***class*** inside it must be declared public manually.

To import a ***package***:-



* We can have **nested classes** in order to **not** create many ***.java*** files for individual class file.

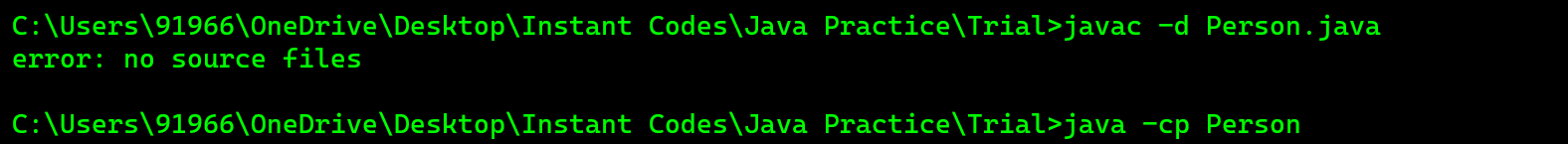
To create object of a **nested *class***:-



**Nested classes** have ***class*** files like this ***\*Animal inside Person\****:-



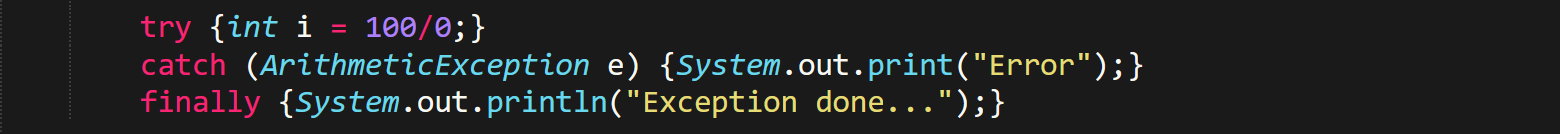
To **compile & run** a package file:-



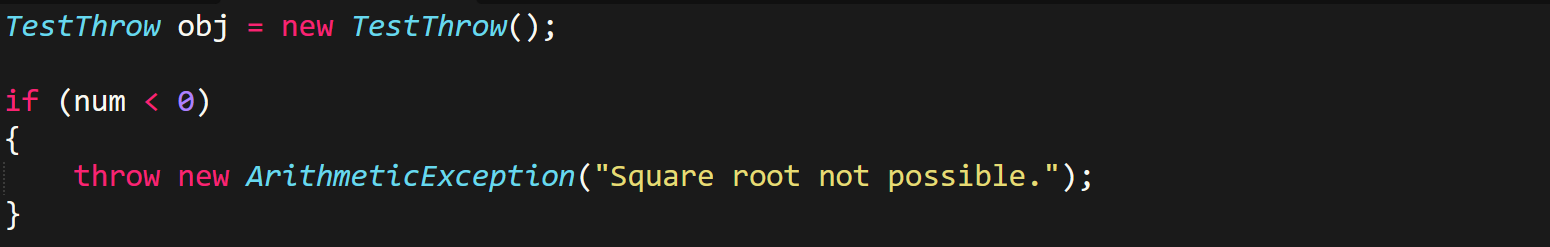
**Exception Handling**



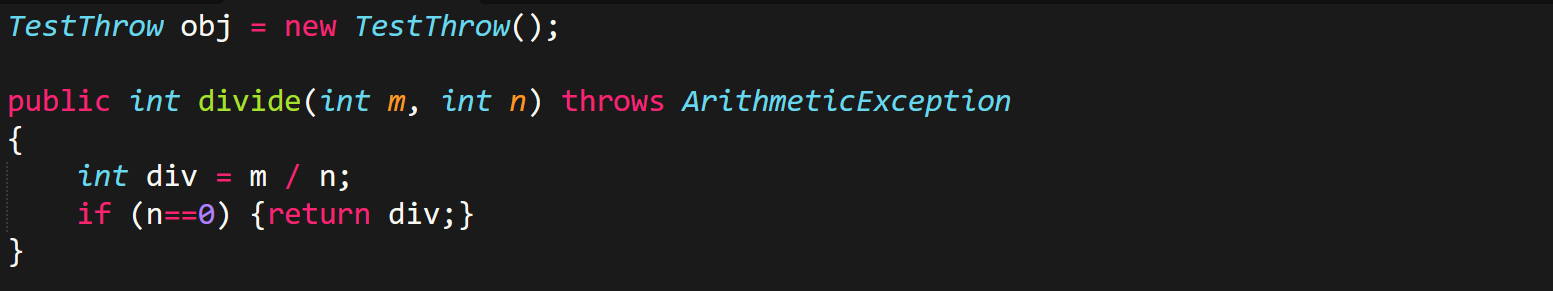
* **Exception:** Anything can **disrupts** the normal flow of a program.
* **Checked exception:** Checked during **compile time**, **present** in class Throwable.
* **Unchecked exception:** Checked during **runtime**, **not present** in class Throwable.
* **Error:** **Nothing** can be done about it through the program.



***throws***:-



***throwable***:-



**Multithreading**

* **User thread:** High priority thread.
* **Daemon thread:** Low priority thread, used for garbage collection & more.
* ***run()*** is subset of method ***start()***, thus ***run()*** is by default executed when ***start()*** is called.
* ***start()*** is called once only, unlike ***run()***.

Java’s multithreading classes includes:-

* ***class* Thread:**



* ***class* Runnable:**



Common methods:-



**Thread** is created by **inheriting** a ***class*** from ***class*** **Thread**:-

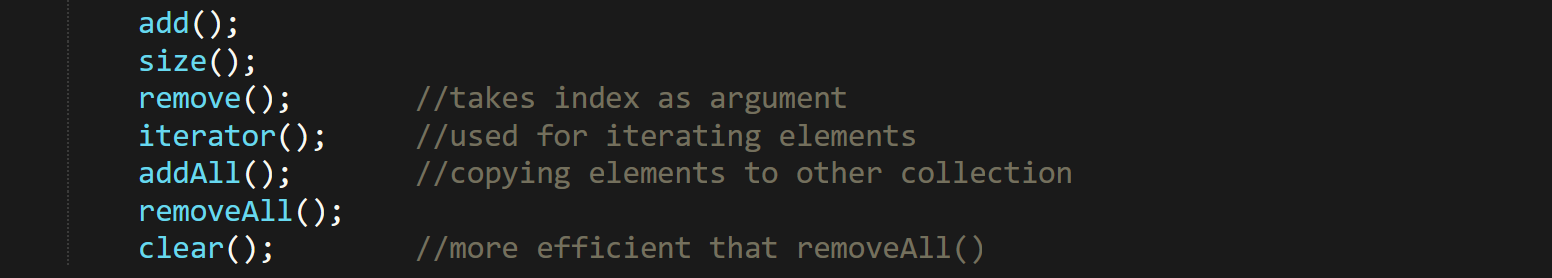


* ***join()*** method **waits** for a thread to complete.
* Thread operations are **better** when done with **exception handling**.
* ***InterruptedException*** is specially used for in multithreading.

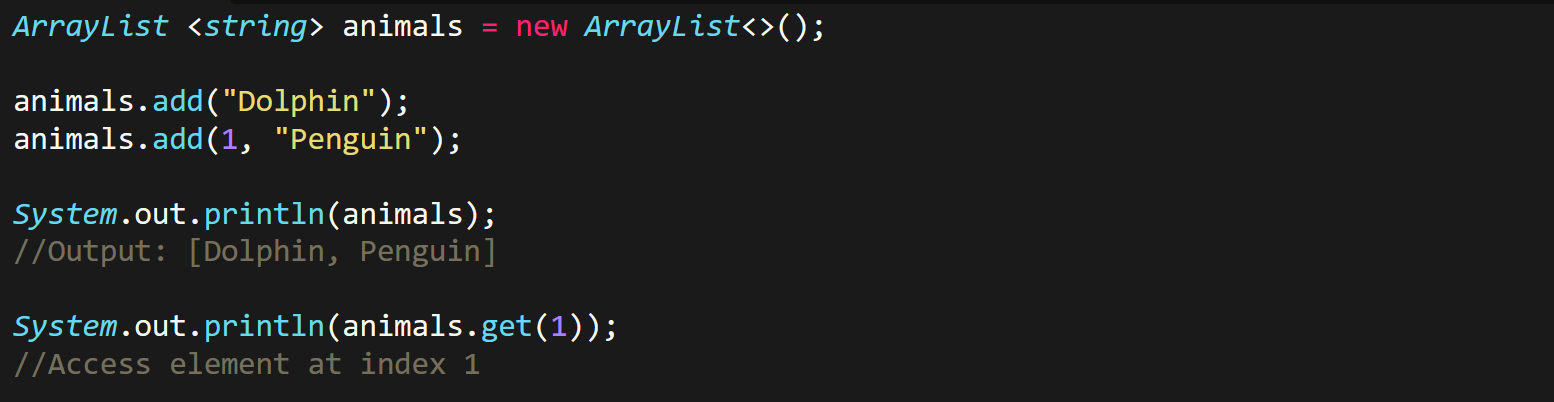
**Collection Framework**

* Found in ***java.util.\*;***

Common methods:-

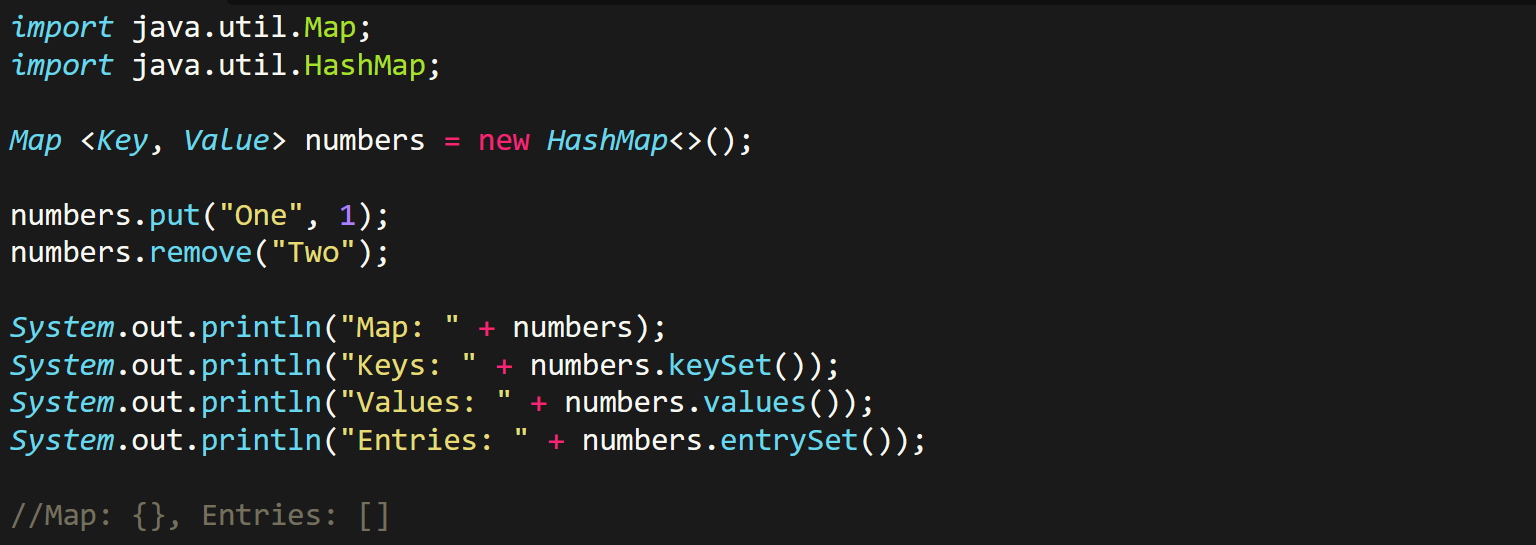


ArrayList:-



Maps:-





Iterator:-



