

Glossary

!!! note

Terms in this glossary are defined informally and are intended to provide the reader with enough understanding to follow the tutorial. For more rigorous and complete definitions, consult [our external resources page](#).

!!! note

Some of these definitions also appear in the tutorial itself, while other appear only here.

Logical Conditionals

Logical conditionals are expressions of the form "If P then Q", and they are false if and only if "P" is true and "Q" is false. In JML they are written as "P => Q".

Property

For us, a property will simply be some fact about some thing, as in "this ball has the property of being round" or "this method has the property of always returning positive ints".

Specification

A specification is a section of JML code that asserts that a portion of java code has a certain property. Specifications can be composed of other specifications. We *verify* that a piece of code conforms to a specification using a software verification tool like **OpenJML**.

Precondition

Preconditions are the assumptions we make about the environment before we verify things. We use them to verify that some property holds of some object, *under some assumption*. For example, a precondition in a method specification could be that the value of an argument variable was positive. This ability (to make specifications that depend on assumptions) is extremely useful, because it allows us to break up "the burden of proof" (i.e, what openJML has to prove) into different components, which can make proofs easier to discover and easier to understand.

Postcondition

Postconditions are conditions about the environment that our specification requires are true after our method has executed. When combined with preconditions, we can write specifications of the form "If this precondition obtains, then this postcondition must obtain".

Pure methods

'pure' method is one without any side effects (i.e, one which doesn't mutate any instance variables). Pure methods are very important in JML because they are the only methods in terms of which we can write our contracts.

Quantifier

A quantifier is an expression that refers to a collection of values, as in the phrases "Every apple is a fruit", or "Some insects are bees". In JML we can express statements asserting a property holds over *every* thing in some category using the `\forall` keyword, and we can express statements asserting a property holds over *some* thing in some category using the keyword `\exists`.

2s Complement

A particular binary representation of integers. See [here](#) for details.