|  |
| --- |
| Circle Language Spec: Execution Control |

## Normal Execution Order

Normal execution order of a procedure means one command being followed up by the next.

But in the new computer language, a command is basically just a collection of calls to other commands with no particular order.

If you don’t indicate the order in which to execute the commands, then they will execute in an arbitrary order. Sometimes this is acceptable from a functional point of view.

The order of a procedure can also be *automatically* determined by *input / output dependency*. A command, called inside a procedure, can take input, that is the output of another command. Then the other command needs to execute first, in order to pass its output on to the next command. This is called the *automatic execution order* principle, part of the flat & structured interchange principles.

If commands should not be executed in an arbitrary order, and the input / output dependency does not determine the order either, then you can indicate the order of execution yourself. This is what we are usually talking about when we refer to *normal execution order*.

Normal execution order places one command call after another to form the order in which to execute the commands. One command actually *calls* the next one. However, it is a special call, because the call never returns to the caller, which it normally does when calling a command. The call to the next command is also called the Command End.

The first command to run is a command, that is no other commands’ next command, not even in the input, output dependency.

Normal execution order is also indicated for calls to execution control statements, like If calls and For calls.

Normal execution order is a form of execution control. Following the *normal execution order* is considered following a consecution of jumps. *Jumps* are execution control statements explained in the article *Jumps*.

The hope that comes with automatic execution order determined by input / output dependency is to not have to define any normal execution order, because if parts of a procedure have no input / ouput dependency, then those parts can always execute in any arbitrary order.

Automatic execution order is not worked out in this article. It is will be worked out in the article *Automatic Execution Order*. Normal execution order is about indicating the order of execution yourself.