Panic

Code inCairo1.0 maypanic - which means it may fail with an unrecoverable error - meaning it is impossible to catch and handle. When the program panics, using thelinear type system all living variables on the stack would beDropped or otherwise destructed, which makes sure the run remains provable and valid.

panic function

The basic function that all panic stems from is thepanic function. It is defined as: extern fn panic(data: Array) -> never; Thepanic function takes a single argument, which is afelt252 array. This array is the data that is passed as the reason the run panicked. Thepanic function never returns, and is marked as such with thenever type .

nopanic notation

Functions may be marked with thenopanic notation. This means that the function will never panic. This can be useful for writing code that may never fail. Onlynopanic functions may be called from anopanic function.

nopanic and traits

If a trait function is marked withnopanic , all implementations of that trait must also be marked withnopanic , as the trait function may be called from anopanic function. An example for such a function isDestruct traitdestruct function, which isnopanic as it is called during panic handling, seelinear type system for more info. If a trait function is not marked withnopanic , all implementations of that trait may be marked withnopanic or not. An example for such a function isAdd traitadd function, which isnopanic forfelt252 addition, but isn’t so for integer addition.

panic_with macro

A function returning anOption orResult may be marked with thepanic_with macro. This macro takes two arguments, which is the data that is passed as the panic reason as well as the name for a wrapping function. If the function returnsNone orErr ,panic function will be called with the given data.

[panic_with('got none value', unwrap)]

fn identity(value: Option) -> Option { a } Somefn unwrap(value: Option) → u128 that internally may panic may be created.

9.4 Inference Memory model ðŸ