Intermediate representation operations set

As the RCL is an abstract virtual machine, there is no question of low-level instructions at any time, but well about the concatenative stack-based style.

Note: all these operations can be re-implemented in the IR itself, but having them in a native way allows better optimizations and greater ease of word management.

Notations

An operator opr action on a stack Δ or a quotation (any upper case letter) is written such as an arrow type taking the arguments $a\ b\ldots\ x$ of any type from the stack Δ as the top items of it and resulting a new stack where the top items are written to as $a\ b\ldots\ x$ of the new stack Δ. For example, $opr : \Delta\ a\ b \to \Delta\ b$ is an operator that take the two top items from the stack Δ and resulting a new stack Δ where the taken item a was popped:

```
3 5 2 opr  ==  3 2
```

A quotation is a pack of operation on the stack that has not yet been evaluated. This can be seen as a stack on the stack or as an anonymous function. We write it as $Q\ a\ b\ldots x$. A quotation one the stack is written $\Delta\ a\ Q[x\ y\ldots z]\ldots c$ if we need to precise the type of data.
The goal of a quotation is to be unquoted at an appropriate time, this will have the effect of executing the content of stack Q in stack Δ. For example:

```
9 [1 +] unquote  ==  9 1 +  ==  10
```

An unquote is written $Q[a] \to a$ where a is the content of the quotation that will be apply on the stack, and the special notation Δ is used to indicate that we work on the main stack.

The composition of operations/functions is noted like a LIFO (Last Input First Output) stack, this composition is the elementary relationship linking the data to each other. The theory stops there, there is really nothing else to say at the elementary level of the concept of intermediate representation.

Stack shufflers

Stack shuffler	Action on the stack (given by its type)
`swap`	$swap : \Delta\ b\ a \to \Delta\ a\ b$
`flip`	$flip : \Delta\ c\ b\ a \to \Delta\ a\ b\ c$
`dup`	$dup : \Delta\ a\ \to \Delta\ a\ a$
`pop`	$pop : \Delta\ a \to \Delta$
`id`	$id : \Delta\ a \to \Delta\ a$
`dip`	$dip : \Delta\ Q\ R[r] \to \Delta\ r\ Q$
`quote`	$quote : \Delta\ a \to \Delta\ Q[a]$

Stack shuffler	Action on the stack (given by its type)
`unquote`	$unquote : \Delta\ Q[a] \to \Delta\ a$
`cons`	$cons : \Delta\ Q[a]\ b \to Q[a\ b]$
`uncons`	$uncons : \Delta\ Q[b\ a] \to \Delta\ Q[b]\ a$