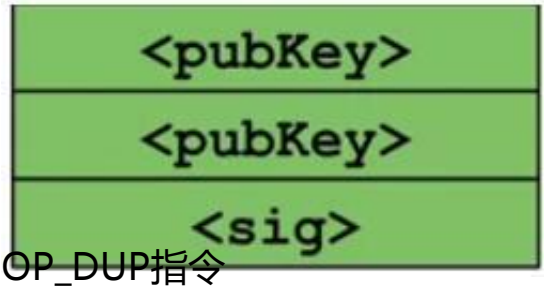
Reverse Polish Language Image.

栈是一种非常常见的数据结构，用Andreas Antonopolous的话说，它允许“栈顶”上的信息“推送”或“弹出”。前者解释了向堆栈中添加信息的过程，而后者描述了从堆栈中删除信息的过程。此外，信息弹出或推送的顺序遵循“后进先出”原则。



OP_DUP指令

这个顶值，即

Hash functions	SHA1	SHA-2		
		256	384	512
Size of hash value (n)	160	256	384	512
Complexity of the best attack	2^{80}	2^{128}	2^{192}	2^{256}
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Message block size (m)	512	512	1024	1024
Word size	32	32	64	64
Numbers of words	5	8	8	8
Digest rounds number	80	64	80	80
Constants Kt number	4	64	80	80

回到发送者和他的同事之间的交易，用户仍然需要向堆栈中添加另一段数据。接下来的信息是作者在交易开始时指定的公钥。需要生成签名来赎回所请求的比特币。

此时，堆栈顶部有两个关键的哈希值数据:作者指定的公钥的哈希值和Mitchell使用的公钥的哈希值。从这里开始使用“OP_EQUALVERIFY”指令，确保发送者确实使用了正确的公钥。在早期经历了几次比特币交易失败后，作者再三检查了Mitchell的公钥。 当公钥匹配时，OP_EQUALVERIFY指令将消耗这些数据点。用户现在只剩下一个签名和一个公钥。最后一步是验证该事务的签名是否正确。

签名和公钥堆栈

比特币脚本语言在这里很有优势，因为它不需要从大量的库中提取来确认签名的有效性。所有这些都内置在语言中。

最后的“OP_CHECKSIG”指令，然后将剩余的两项从堆栈中弹出，如果