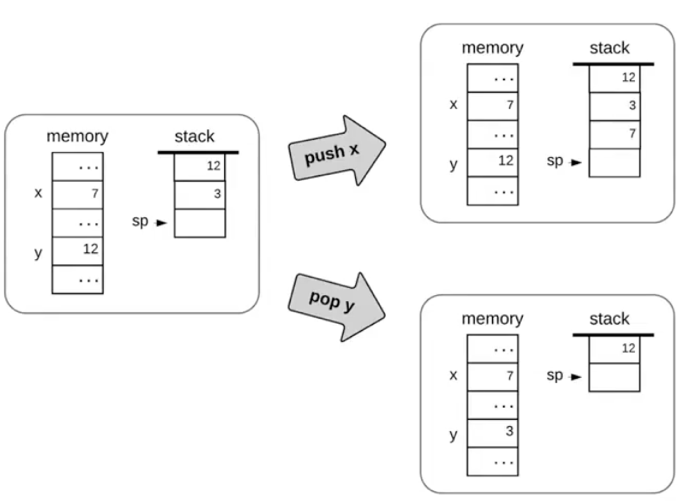
**Virtual Machine**

Unit 1.2: VM Abstraction: the stack

1. Stack operations:

* push: add a plate at the stack’s top
* pop: remove the top plate

push and pop相当于完成了一次赋值



1. Stack machine is manipulated by:
2. Arithmetic / Logical commands
3. Memory segment commands
4. Branching commands
5. Function commands
6. Stack arithmetic

* add: the top two values
* neg: negate the top value

1. Applying a function f on the stack

* pops the arguments from the stack
* Computes f on the arguments
* Pushes the results onto the stack

箭头

中度可信度描述已自动生成

Unit 1.3: VM Abstraction: Memory Segments

1. Variable kinds

* Argument variables
* Local variables
* Static variables

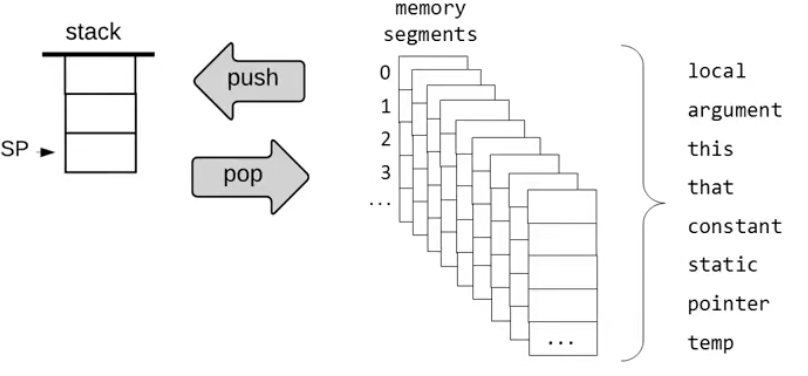
1. Virtual memory segments

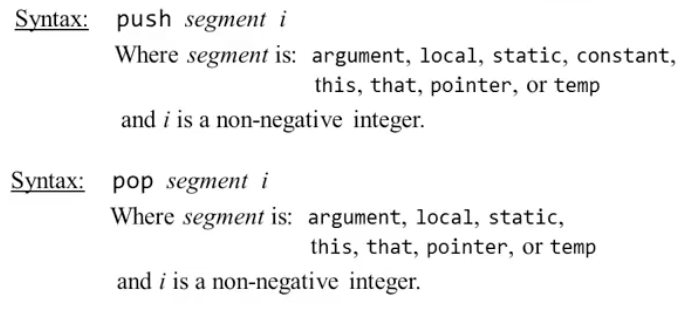
图表, 直方图

描述已自动生成

1. The difference between push and pop:

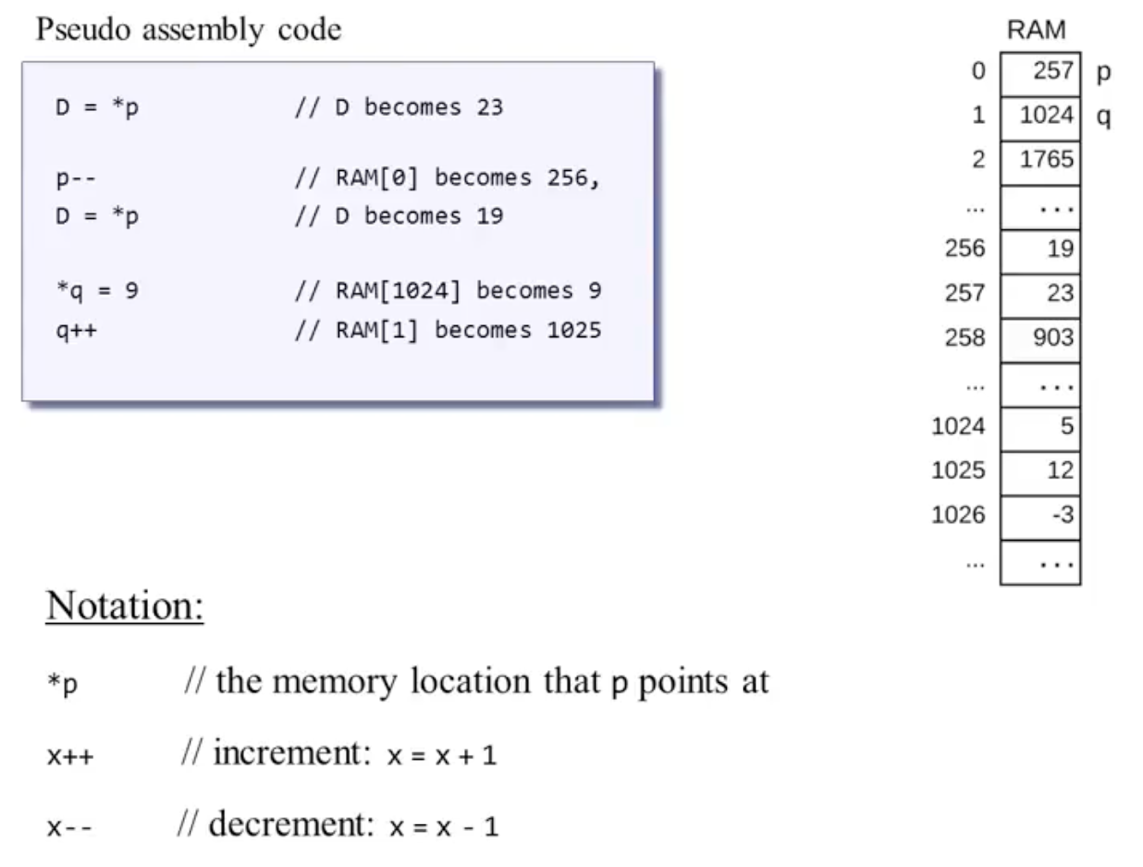
**You cannot pop a constant**



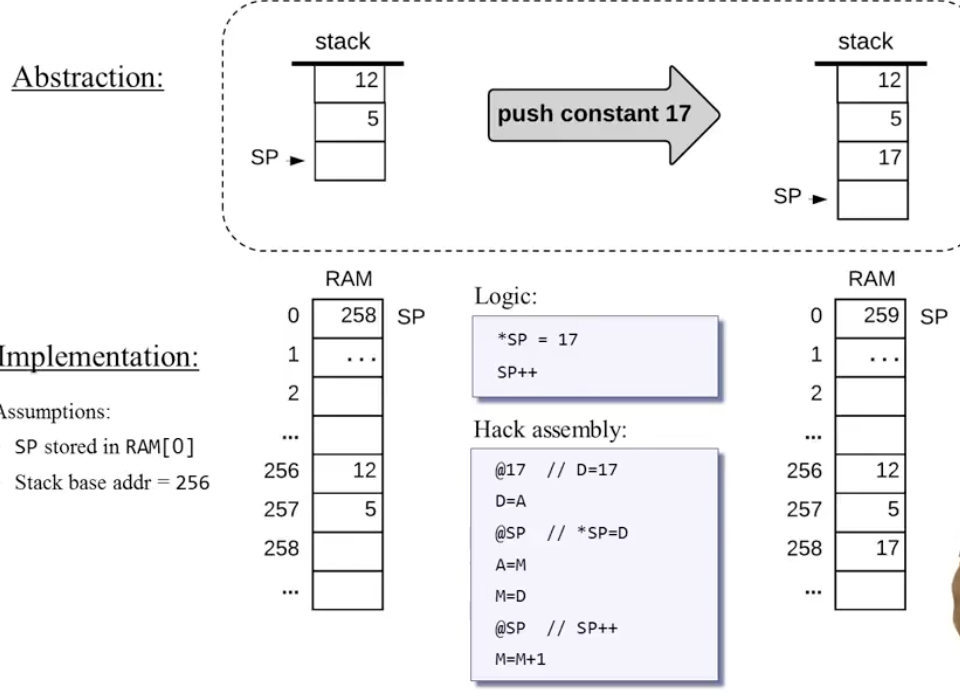


Unit 1.4: VM Implementation: the Stack

1. Pointer manipulation



1. Stack machine



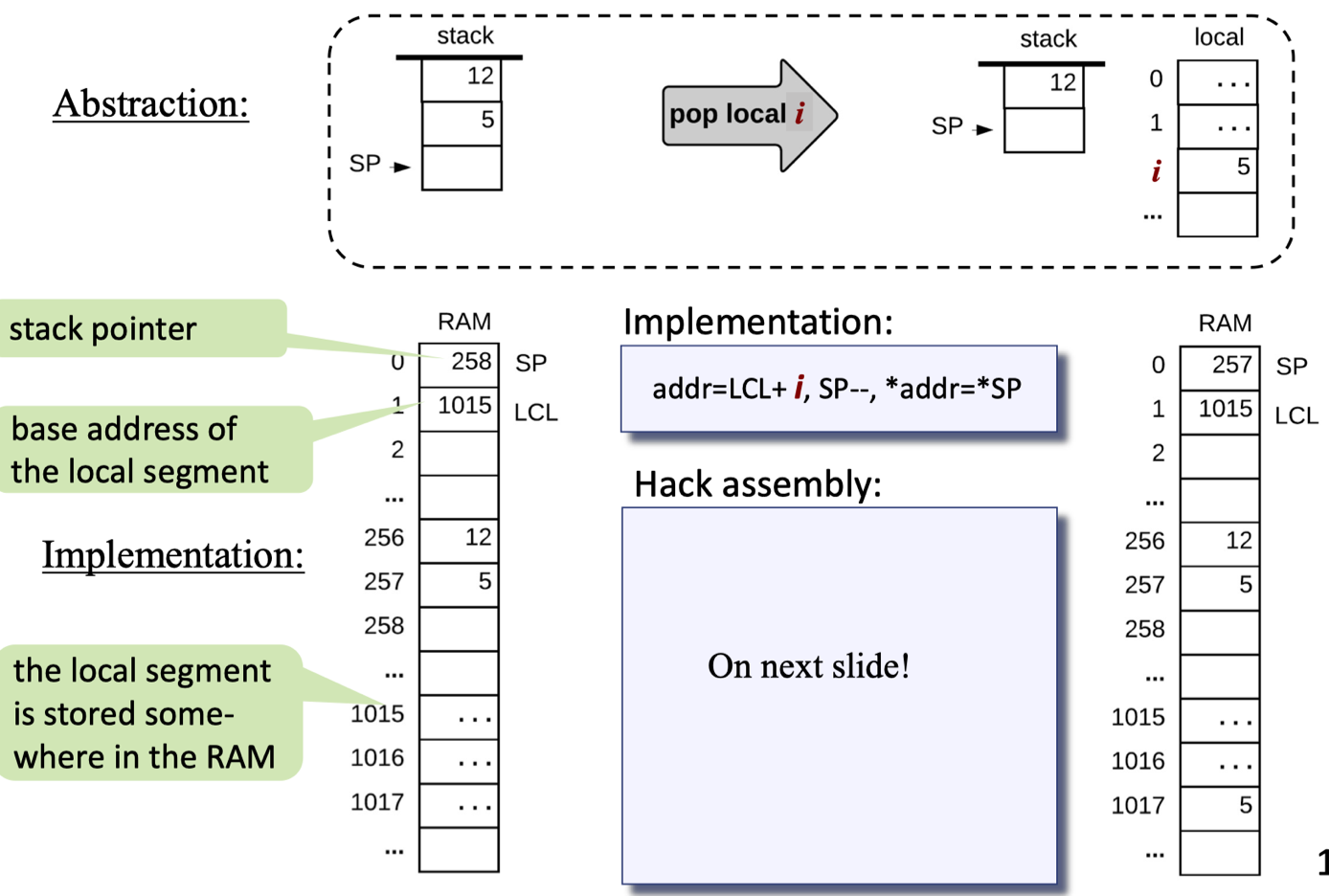
1. VM Translator

* A program that translates VM code into machine language
* Each VM command generates several assembly commands

应用程序, 形状, 箭头

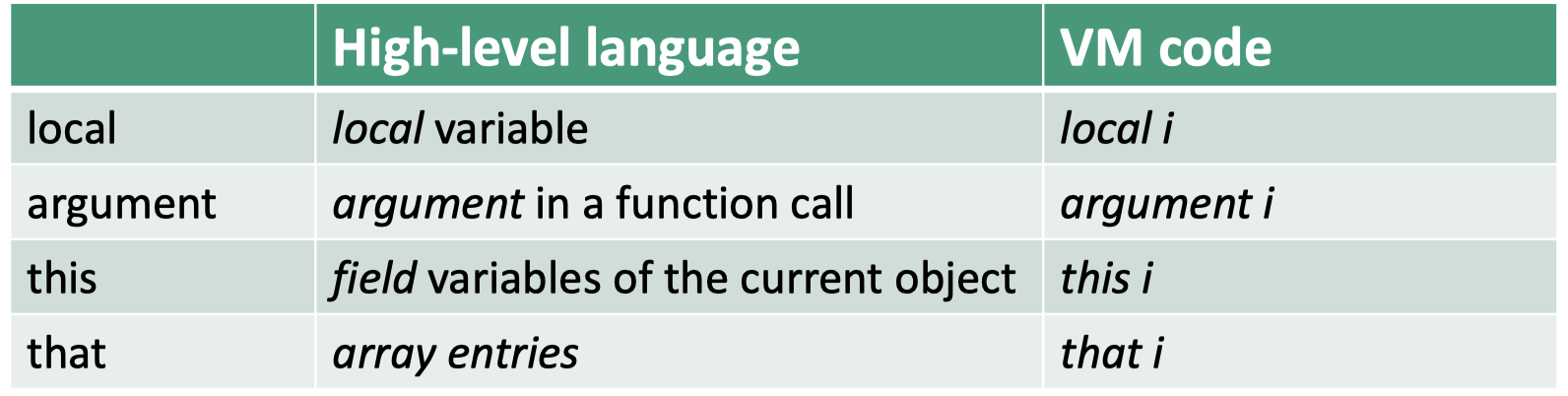
描述已自动生成

Unit 1.5: VM Implementation: Memory Segments

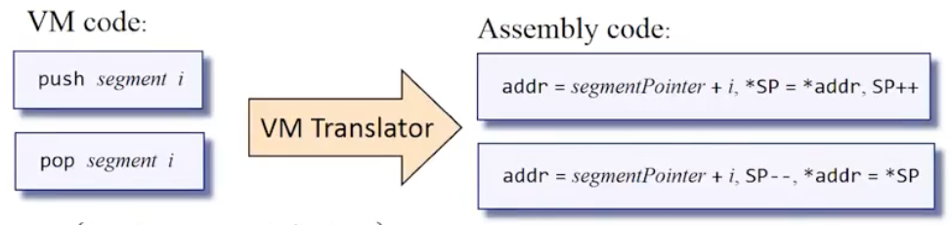


形状, 箭头

描述已自动生成



1. Implementing local, argument, this, that

图示, 文本, 聊天或短信

描述已自动生成

1. Implementing constant

形状, 箭头

描述已自动生成

1. Implementing static

图形用户界面, 文本, 应用程序

描述已自动生成

1. Implementing temp

图示, 文本

描述已自动生成

1. Implementing Pointer

图示

描述已自动生成

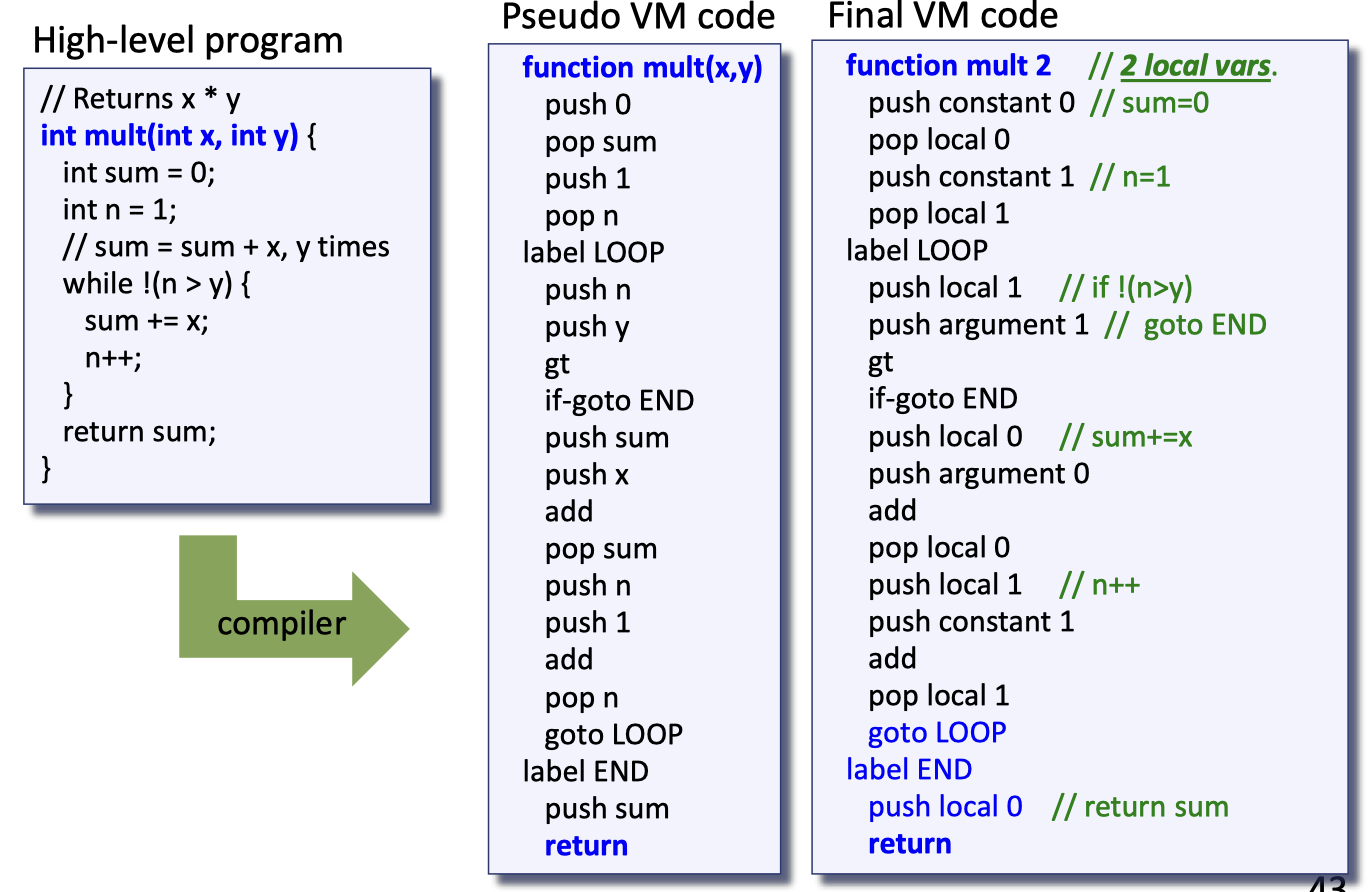
Unit: VM: Branching Commands and Function Commands

1. Branching Commands

图形用户界面, 文本, 应用程序, 聊天或短信

描述已自动生成

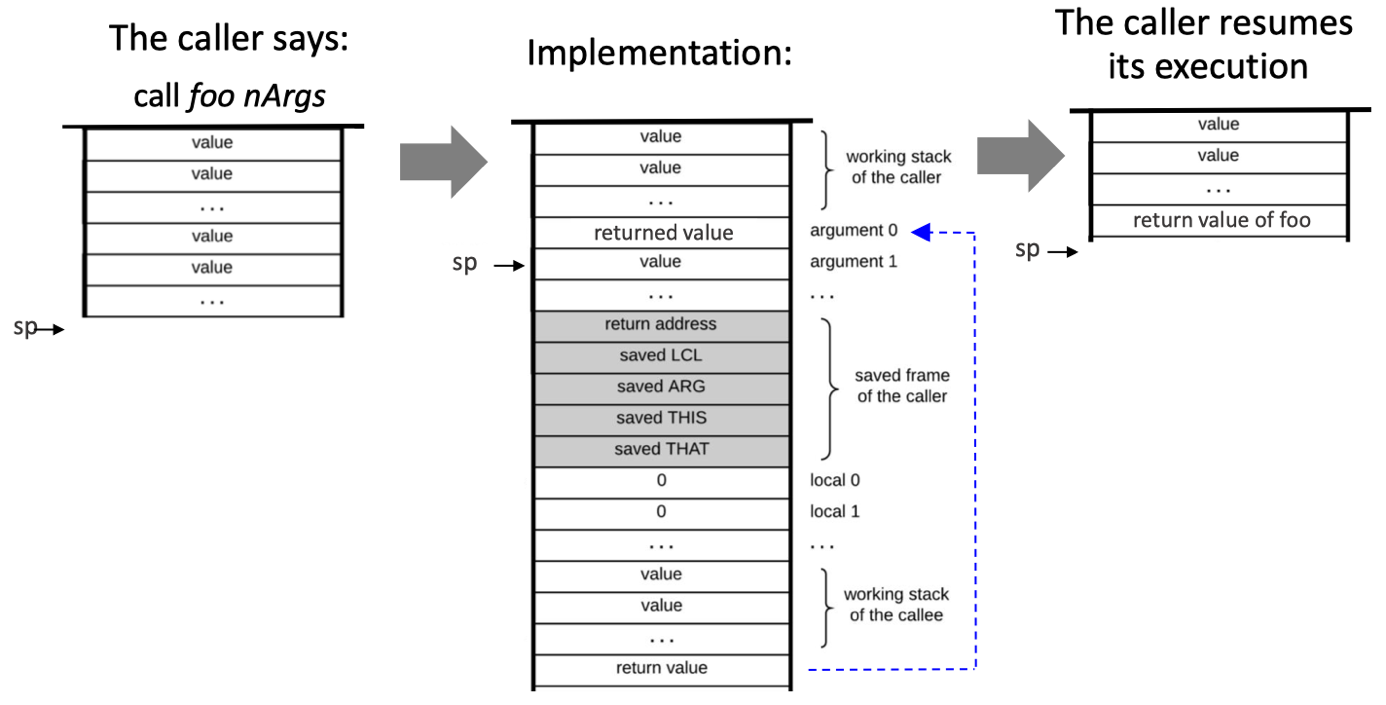
1. Abstraction of Function Command



图示

描述已自动生成

1. Implementation of Function Commands



VM implementation (handling call):

1. Set ARG

2. Save the caller ’s frame

3. Jump to execute foo

VM implementation (handling function):

• Set up the local segment of the called function

VM implementation (handling return):

1. Copy return value onto argument 0.

2. Set SP for the caller.

3. Restore segment pointers of the caller.

4. Jump to the return address within the caller ’s code.

(note that the stack space below sp is recycled)

