Insper

Elementos de Sistemas

Aula 18 – Operações em Pilhas

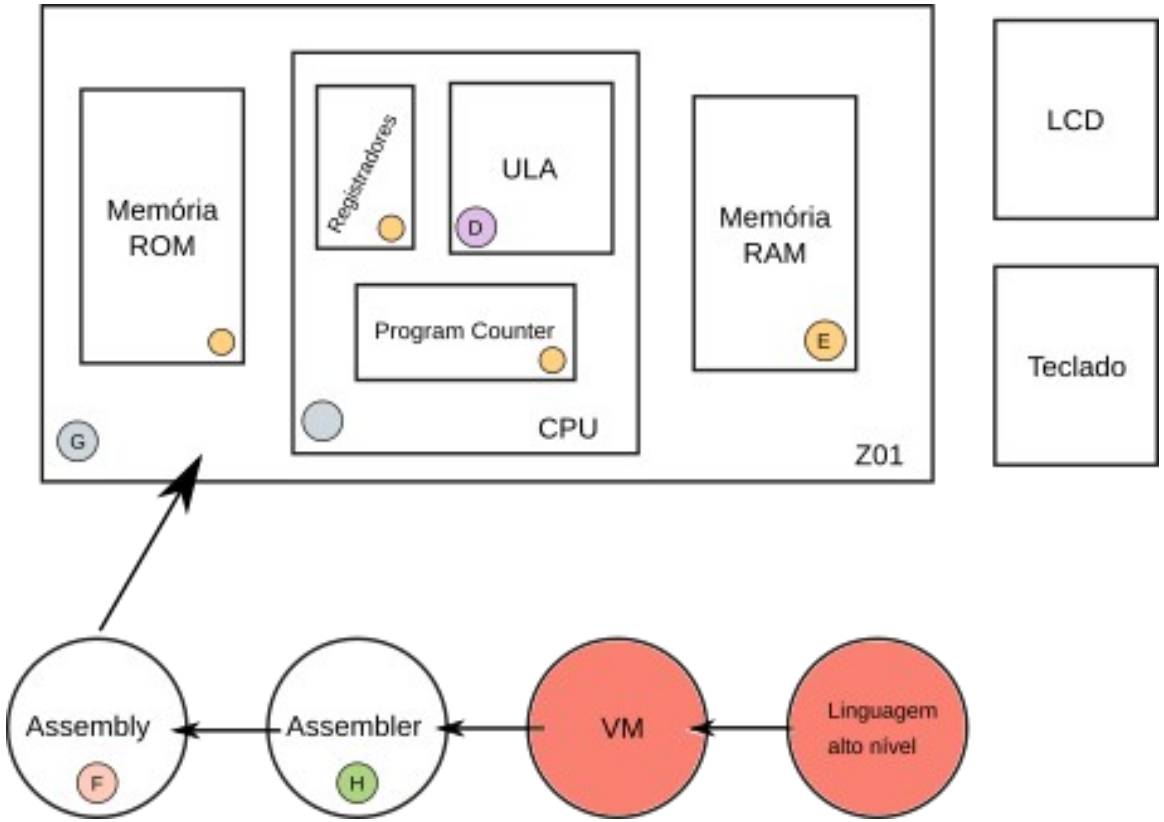
"Qualquer tecnologia suficientemente avançada é indistinguível de mágica."
"Any sufficiently advanced technology is indistinguishable from magic."
Arthur C. Clarke (1917-2008), Escritor Britânico

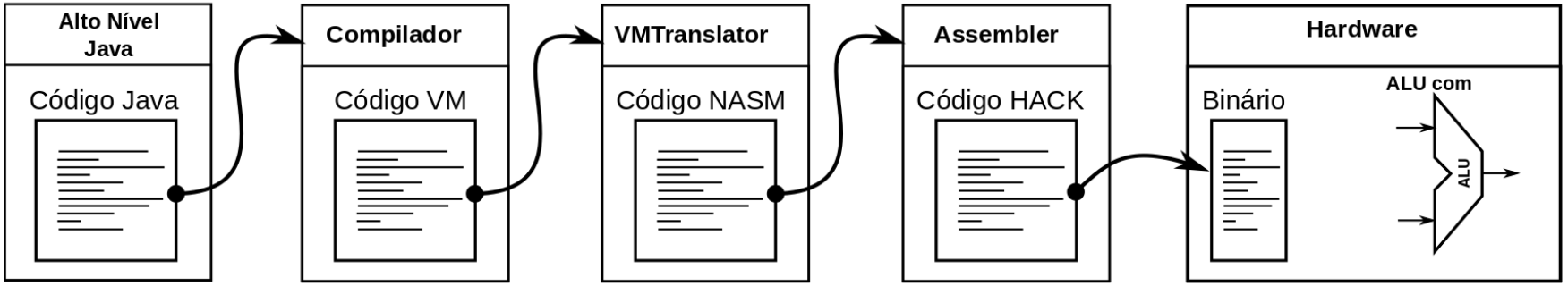
Engenharia - Luciano Soares <lpsoares@insper.edu.br>

Objetivos de Aprendizado da Aula

- Operar em Pilhas;
- Notação Polonesa Reversa;
- Gerenciar Ponteiros de Memória.

Conteúdo(s): Gerenciamento de Memória.


Memória
ROM
Registradores
ULA
D
Program Counter
CPU
Memória
RAM
E
G
Z01
LCD
Teclado
Assembly
F
Assembler
H
VM
Linguagem
alto nível


Alto Nível
Java
Código Java
Compilador
Código VM
VMTranslator
Código NASM
Assembler
Código HACK
Hardware
Binário
ALU com
ALU

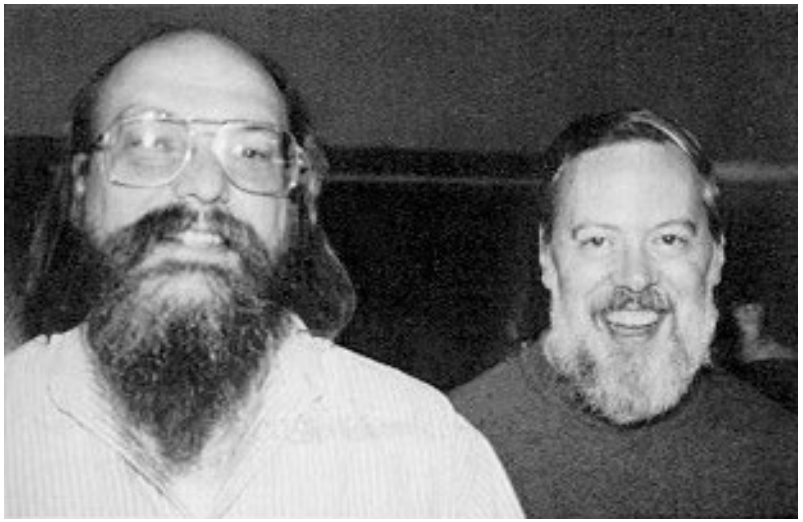
Máquina Virtual do Z0

Para o desenvolvimento do Z0 será adotada uma Maquina Virtual como fase intermediária. Essa máquina virtual pode ser vista como:

- Uma máquina com seu próprio ambiente (um computador virtual).
- Uma representação conveniente de um programa de computador entre o código de alto nível e a linguagem de máquina.

Ken Thompson e Dennis Ritchie

Ken Thompson e Dennis Ritchie enquanto trabalhavam no Bell Labs desenvolveram diversas das tecnologias usadas atualmente em computadores como a linguagem de programação C, o sistema operacional Unix.



Ken Thompson Dennis Ritchie

LUA

Lua é uma linguagem de script compacta, que suporta programação procedural, programação orientada a objetos, programação funcional, dentre outras.

O Lua é executado através da interpretação de um bytecode com uma máquina virtual baseada em **registradores**. A linguagem é muito usada para arquivos de configuração, scripts e prototipagem rápida.

Waldemar, Roberto, Luiz

Motivação

```
class Main {
  static int x;

  function void main() {
    // Input and multiply 2 numbers
    var int a, b, x;
    let a = Keyboard.readInt("Enter a number");
    let b = Keyboard.readInt("Enter a number");
    let x = mult(a,b);
    return;
  }
}

  // Multiplies two numbers.
  function int mult(int x, int y) {
    var int result, j;
    let result = 0; let j = y;
    while not(j = 0) {
      let result = result + x;
      let j = j - 1;
    }
    return result;
  }
}
```

Traduzir programas em linguagem de alto nível para um código executável.

Compilar

```
    ...
    leaw $a,%A
    movw %D,(%A)
    leaw $b,%A
    movw $0,(%A)
  LOOP:
    leaw $a,%A
    movw (%A),%D
    leaw $b,%A
    subw %D,%A,%D
    leaw $END,%A
    jg
    leaw $j,%A
    movw (%A),%D
    leaw $temp,%A
    addw %D,(%A),%D
    movw %D,(%A)
    leaw $j,%A
    movw (%A),%D
    incw %D
    movw %D,(%A)
    leaw $LOOP,%A
    jmp
    ...
```

Insper

Comandos Suportados pela VM do Z0

Comandos Aritméticos e Booleanos

- **`add`** (soma)
- **`sub`** (subtração)
- **`neg`** (negação)
- **`eq`** (igual)
- **`gt`** (maior que)
- **`lt`** (menor que)
- **`and`** (e)
- **`or`** (ou)
- **`not`** (não)

Comandos de Acesso de Memória

- **`push`** **`segmento`** **`x`** (empilha)
- **`pop`** **`segmento`** **`x`** (desempilha)

Comandos de Fluxo de Execução

- **`label`** (declaração)
- **`goto`** (marcadores)
- **`if-goto`** (marcadores)

Comandos de Chamada de Funções

- **`function`** (declaração da função)
- **`call`** (invoca uma função)
- **`return`** (retorno da função)

Insper

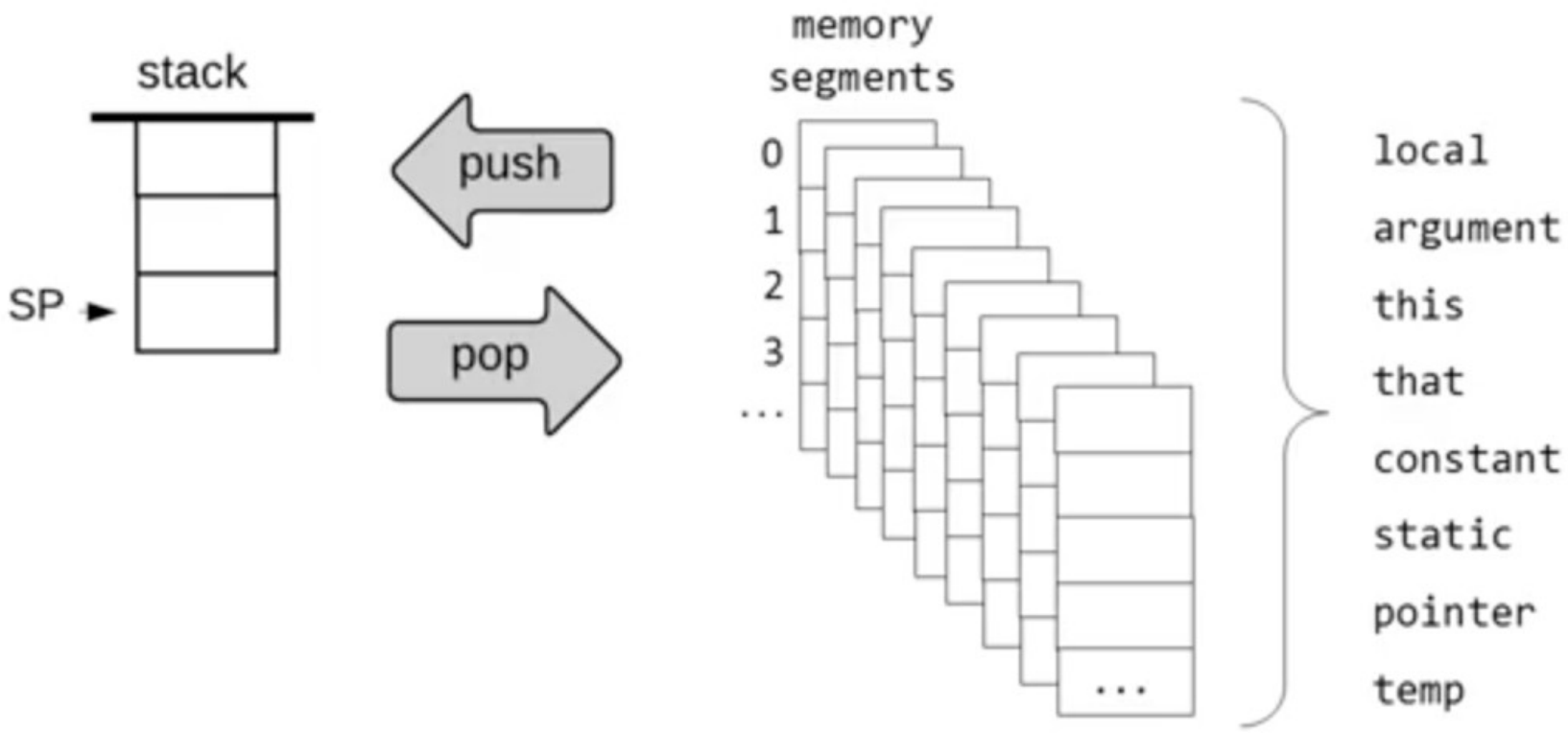
Aritmética por Pilha

Revendo:

Insper

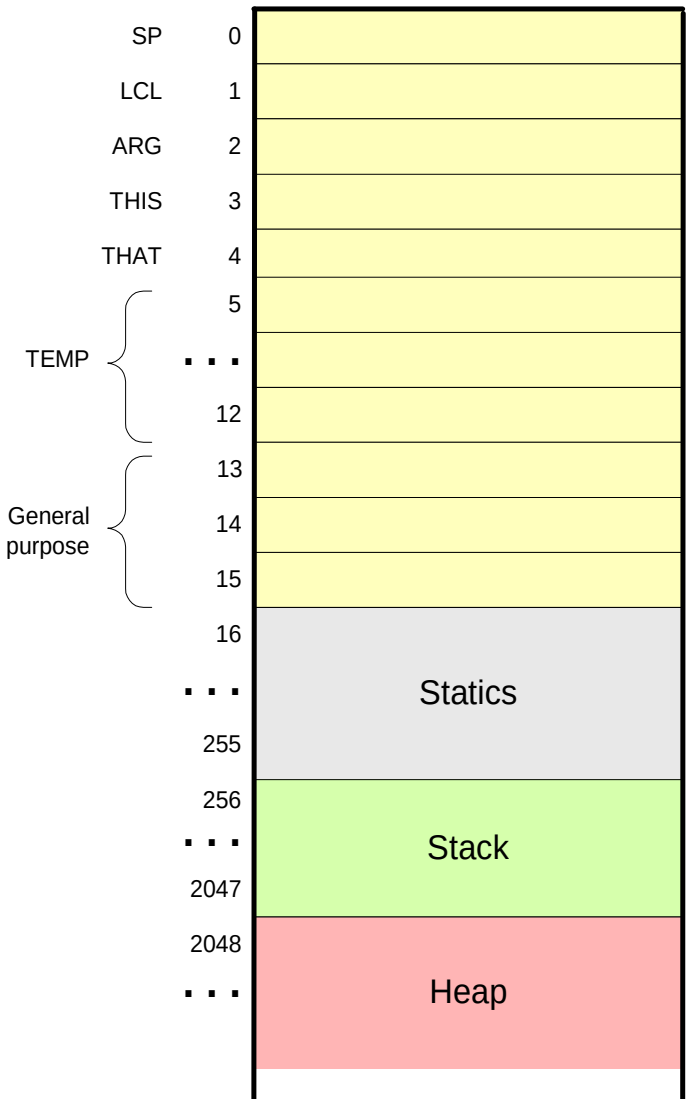
Segmentos de Memória

O usuário pode fazer o PUSH ou POP para algum segmento de memória.



Insper

Organização de memória na VM do Z0



- O **SP** (**S**tack **P**ointer) aponta a posição para a inserção de um novo elemento na pilha (Stack), ele é definido no endereço 0 (Zero) da RAM
- **LCL** (Local), **ARG** (Arguments), **THIS** e **THAT** são ponteiros usados para controlar a posição dos dados nas chamadas de funções.
- A região **Statics** é uma área de memória compartilhada por todas as funções.
- O **Stack** é a pilha de dados do computador. O espaço de memória entre os endereços 256 e 2047 é reservado para armazenar os dados da pilha.
- O **Heap** é a memória dinâmica de dados e pode ser acessada de diversas formas. Essa memória é reservada entre os endereços 2048 e 16383.

Stack, SP e Heap

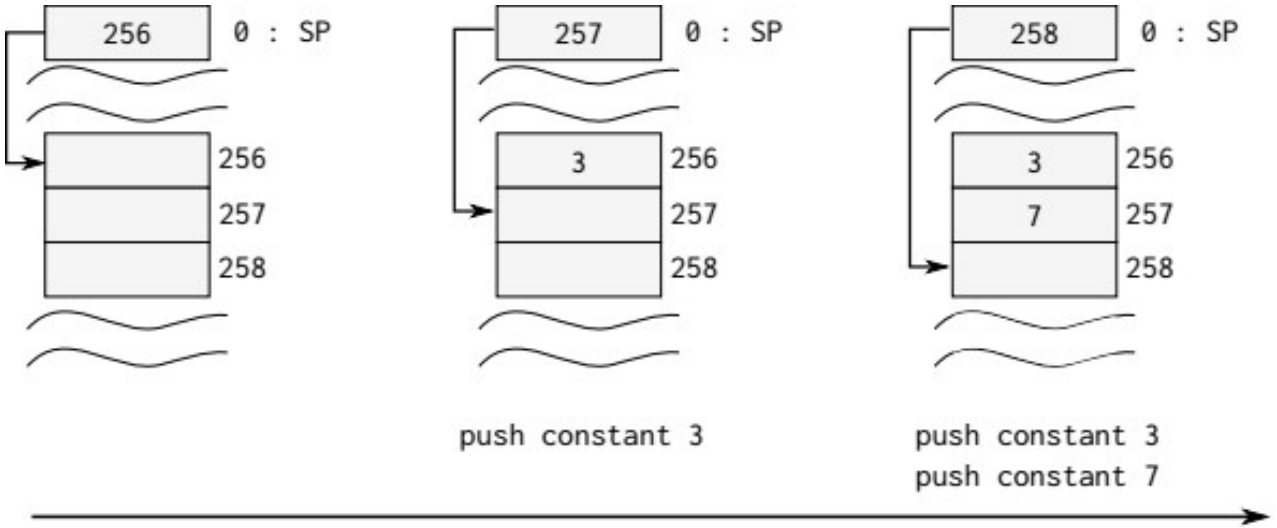
Stack (pilha) é a pilha da máquina virtual, ou seja, o espaço da memória que armazena os dados dos pushs e pops.

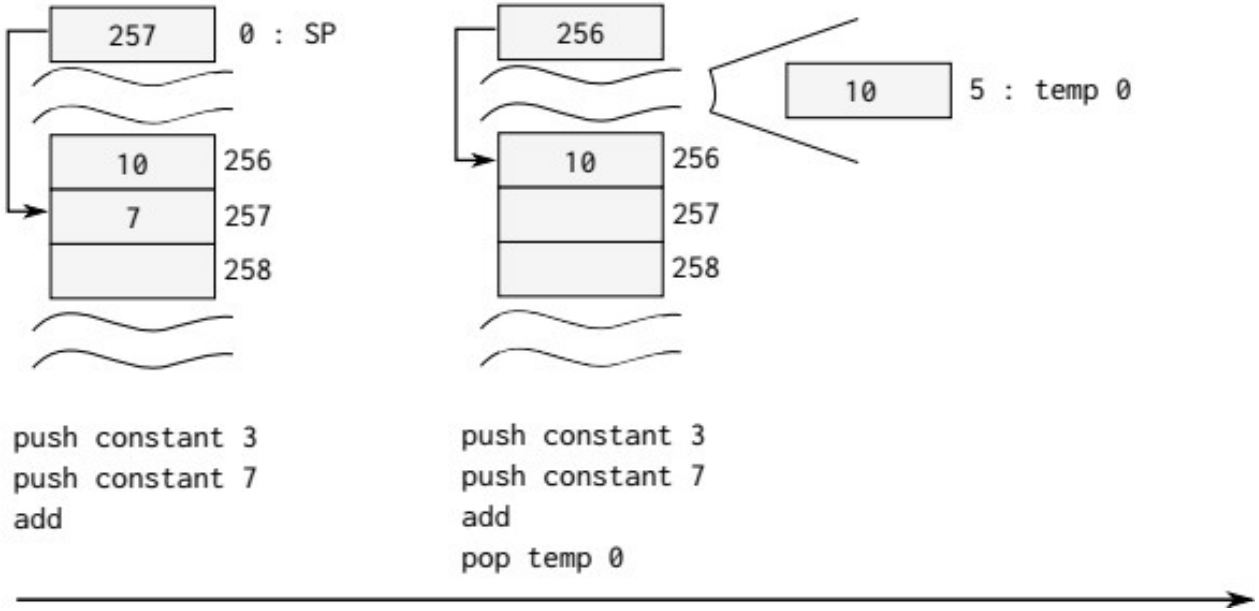
SP é um ponteiro que mostra o próximo local de memória disponível para armazenar um valor. As operações de push e pop fazem esse ponteiro aumentar e diminuir.

Heap é a área usada para armazenar dados dinâmicos diversos dos programas. Os programas alocam e desalocam memória para seu funcionamento aqui.

Exemplo de Memória

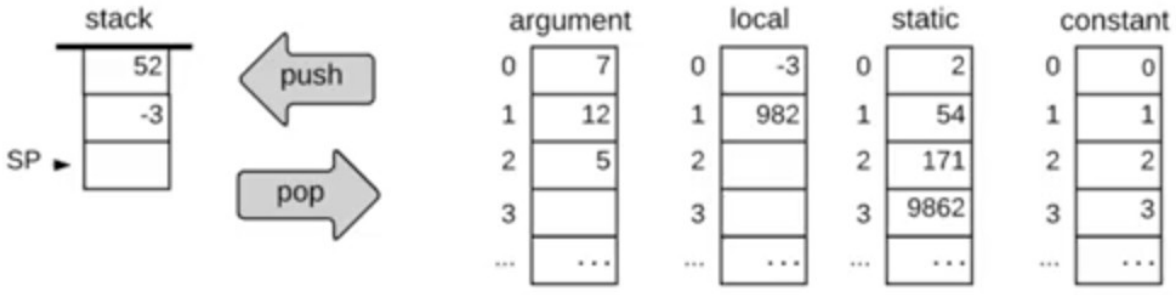
Quando uma constante é inserida por um PUSH, ela é armazenada diretamente na pilha.


256
0 : SP
256
257
258
257
0 : SP
3
256
257
258
258
0 : SP
3
256
7
257
258
push constant 3
push constant 3
push constant 7


257
0 : SP
10
256
7
257
258
256
10
5 : temp 0
10
256
257
258
push constant 3
push constant 7
add
push constant 3
push constant 7
add
pop temp 0

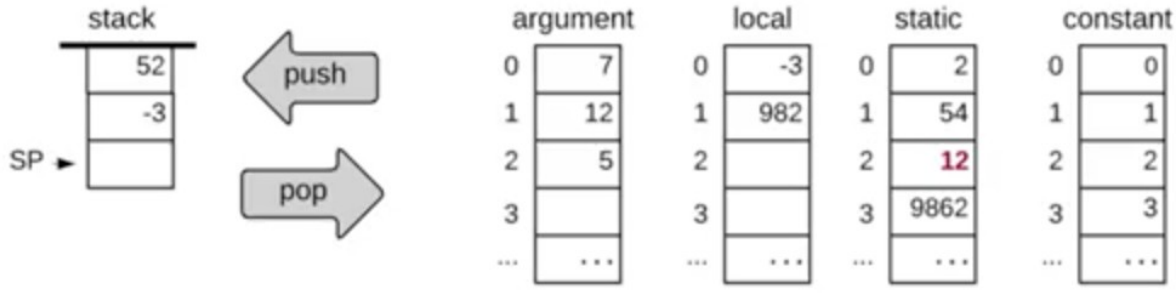
Pergunta

Qual é o código que realiza a operação abaixo?




```
let static 2 = argument 1
```

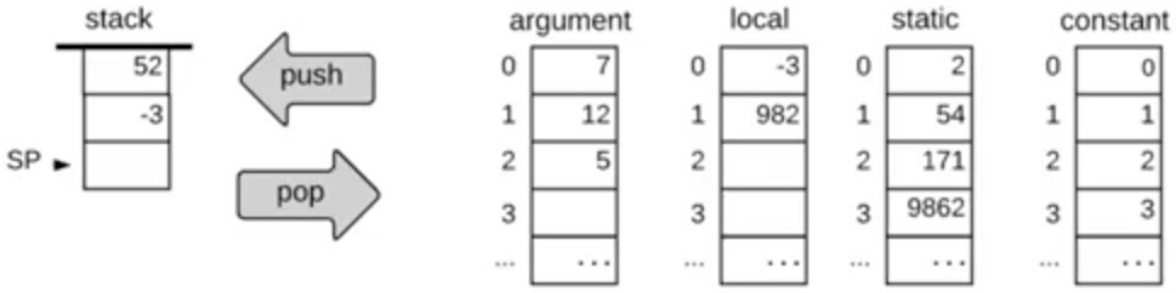
?



Insper

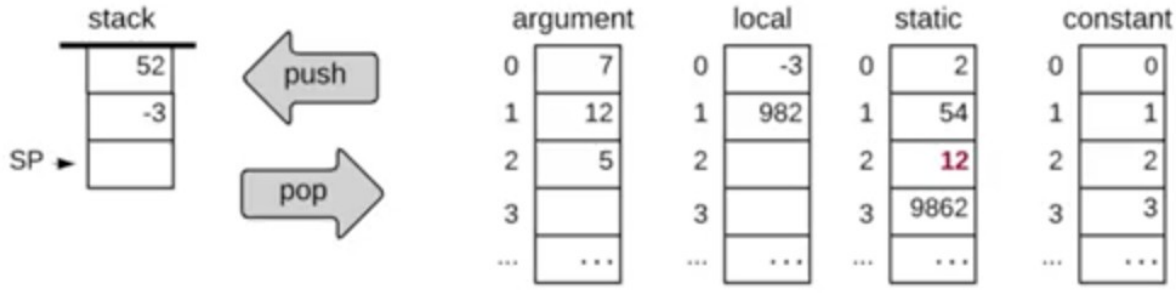
Pergunta

Qual é o código que realiza a operação abaixo?



let static 2 = argument 1

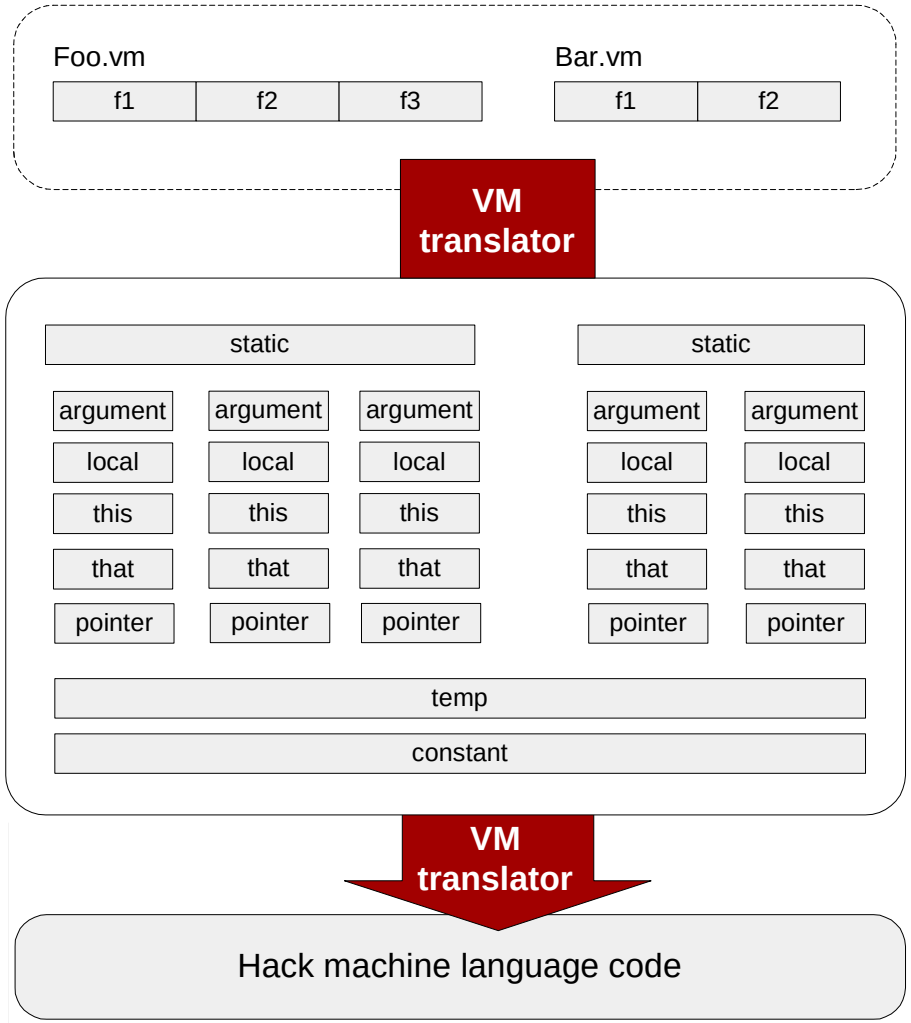
```
push argument 1
pop static 2
```



Insper

Organização dos arquivos

Vários arquivos .vm com suas funções gerando um só .asm



VM files

(one set of virtual segments for each instance of a running function)

Insper

Manipulação de Ponteiros

Pseudocódigo:

D = *p

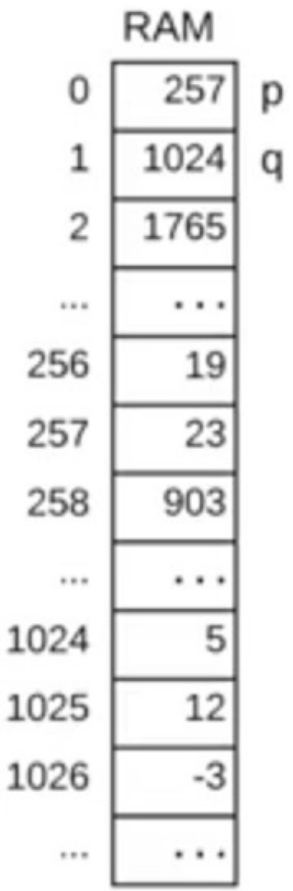
* significa a notação para a localização na memória apontada por p

Em Assembly:

```
leaw $SP,%A
movw (%A),%A
movw (%A),%D
```

Resultado:

D recebe o valor 23.



Insper

Funções e Memória

Os programas na máquina virtual são organizados em funções, com cada função organizando seus dados e formas de acesso.

```
function NomeDaFunção nLocals

call NomeDaFunção nArgs

return
```

Formar Duplas

Os Srs. Dennis Ritchie e Ken Thompson estão avaliando criar uma nova linguagem usando um máquina a pilha como linguagem intermediária. Você pode ajudar eles.

Formem duplas para resolver as atividades.

Mantenha um ambiente em que todos participem de tudo.

* Isso é uma obra de ficção, Dennis Ritchie e Ken Thompson não escreveram esse documento.

Próxima Aula

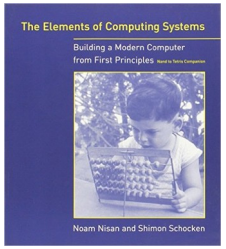
- Ver estudo para aula ? sobre **????**
- Estudar Lista de Exercícios Aula ? (opcional):
- Ler (opcional)

Capítulo 8

The Elements of Computing Systems

Building a Modern Computer from First Principles

Noam Nisan e Shimon Schocken



Insper

Insper

www.insper.edu.br