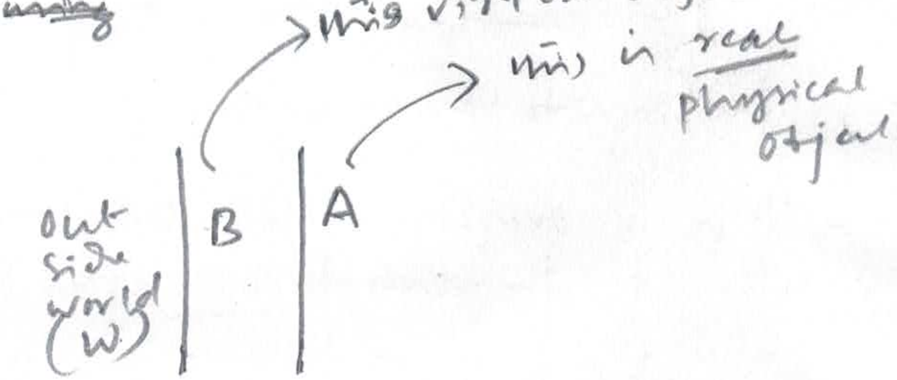
- Lecture-15 - <6/11/2019>

Virtual Machine :-

The concept of virtual :-

- Virtual Memory
- Virtual Machine
- Virtualization (virtual Box)

Virtual :- To mimic the structure and behaviour of A by B. ~~using~~



——— (not physically existing as such but made by software to appear to do so)

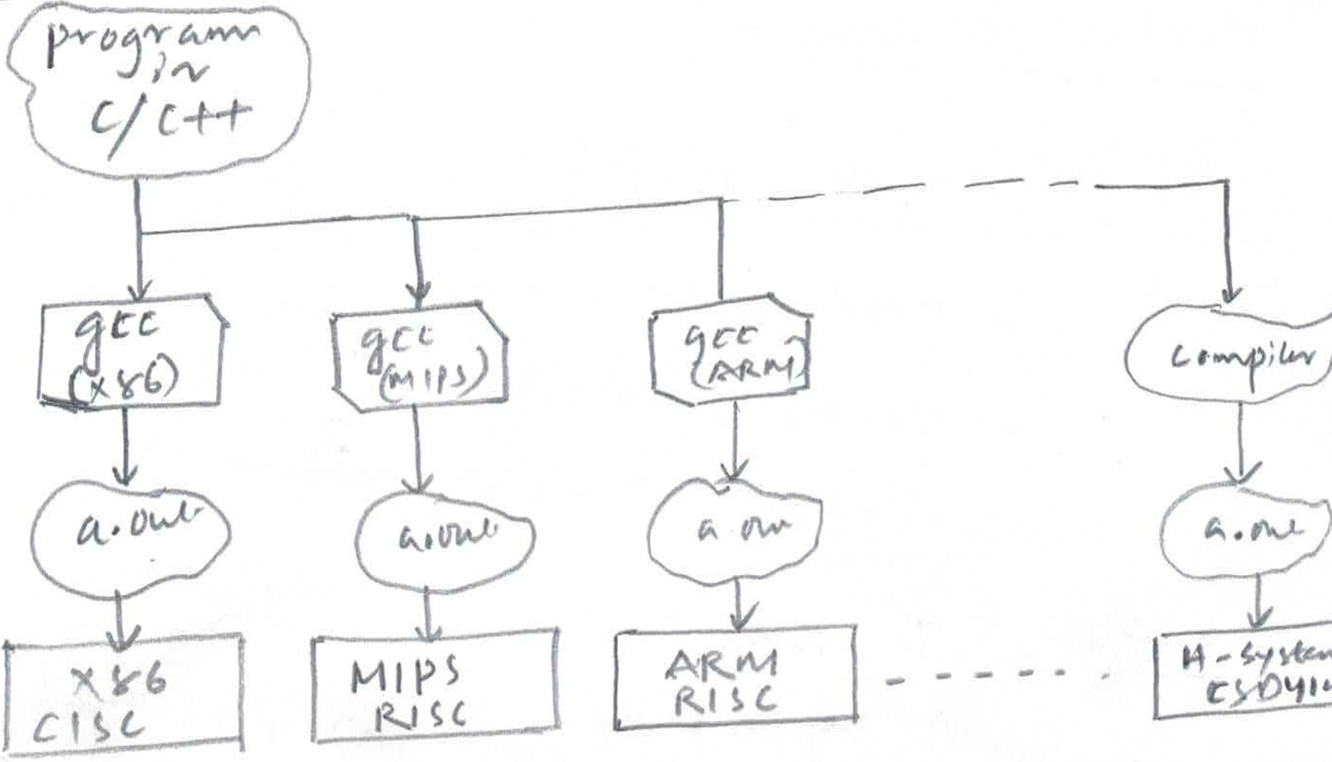
Virtual Machine :-

A - Hardware architecture

B - Virtual Machine (VM)

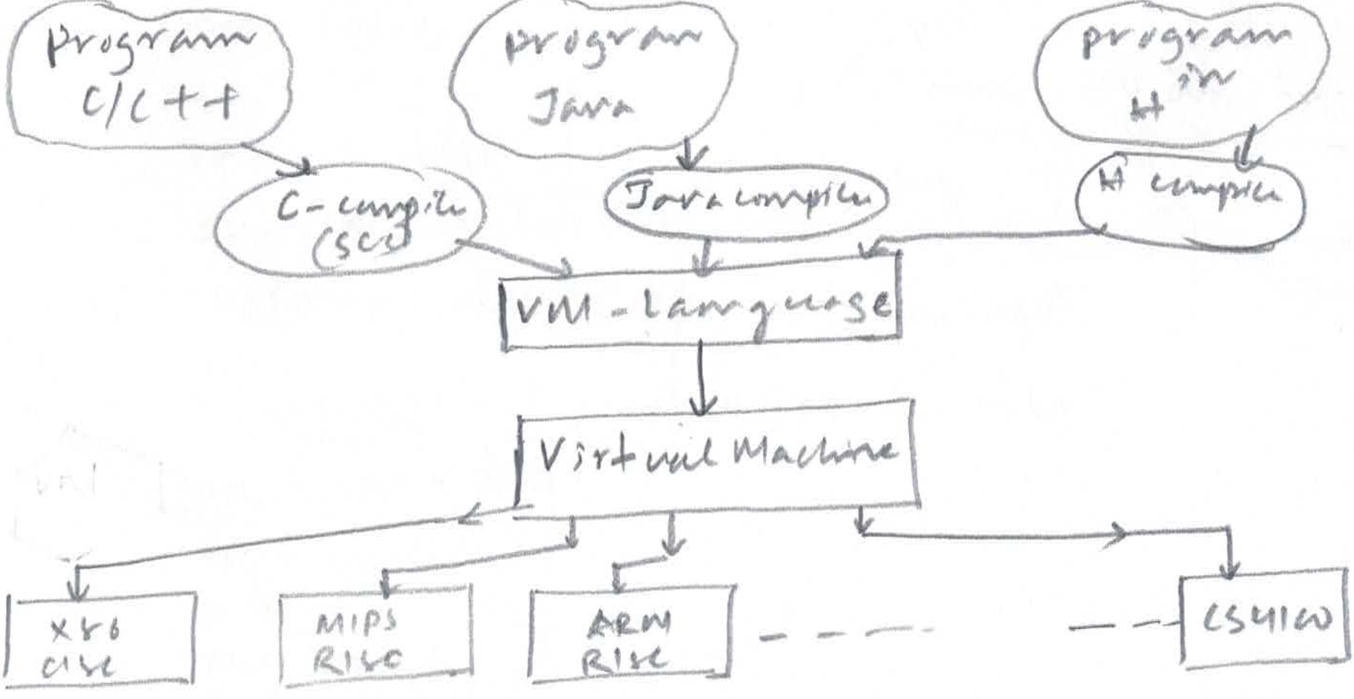
W - Programming Language & (compiler)

Scenario



This limitation — Code portability!

- How to overcome this limitation?
  - Some how you have to have a common hardware platform.
  - If you can't have physical hardware just create one using software.



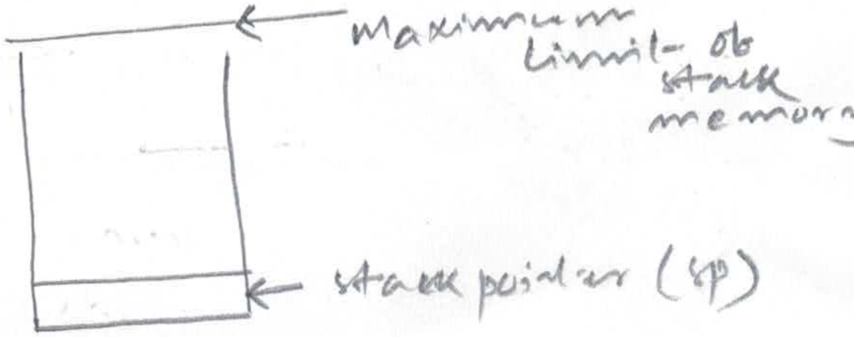
Virtual Machine Architecture :-

- Stack based Architecture — Example
  - Java (JVM)
  - .Net (CLR)
- Register based architecture
  - Dalvik virtual Machine (DVM) (Targetted for Mobile platform)

Review of stack operation :-

operation

- push
- pop



Example :-

push A

$[sp] \leftarrow A$

$sp \leftarrow sp-1$

pop A

~~$A \leftarrow sp$~~

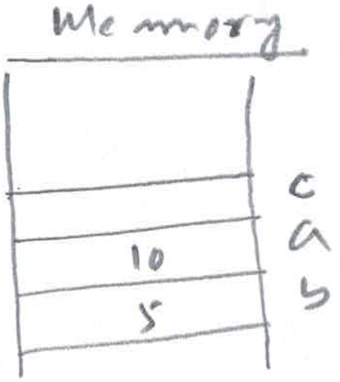
$sp \leftarrow sp+1$

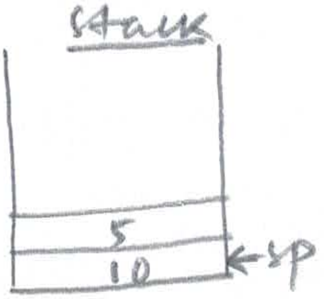
$A \leftarrow [sp]$

Stack based expression evaluation :-

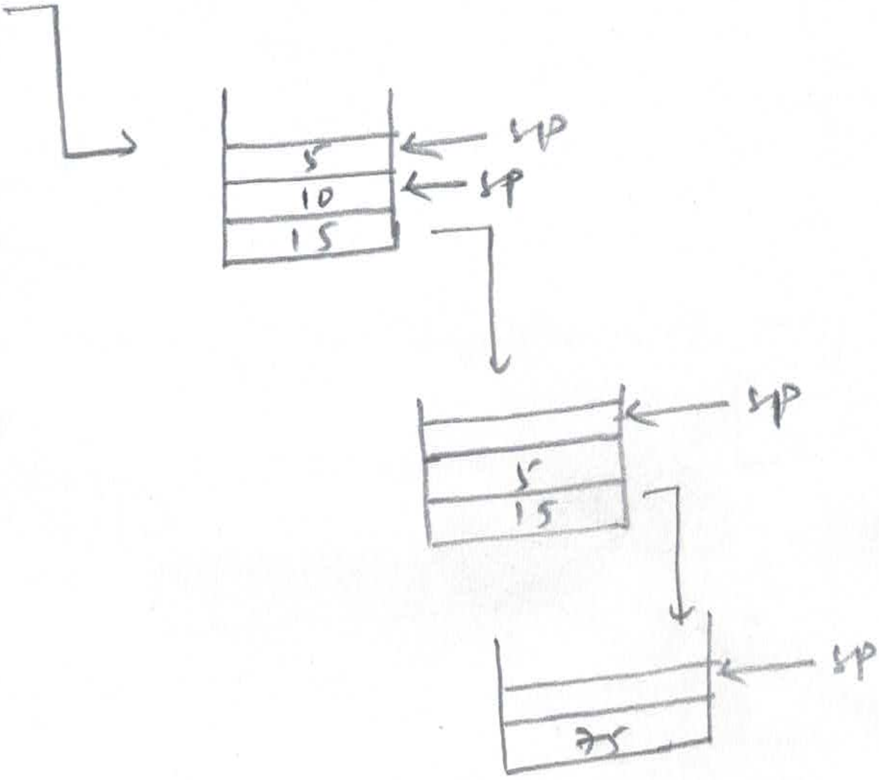
Example - $c = (a+b) * (a-b)$

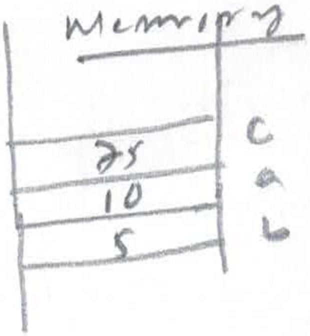
Stack based execution engine





```
Push a
Push b
add
Push a
push b
sub
mul
pop c
```



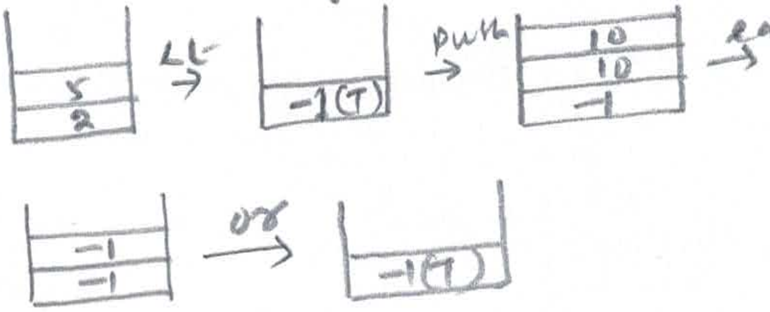


Example (2) -

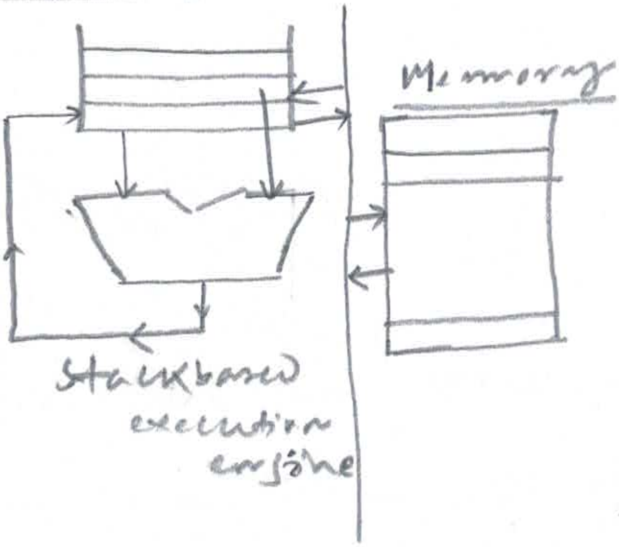
if (x<5) or (y=10) the final out-put of this statement be <u>true</u> or <u>false</u>

```
Push x
Push 5
lt
push y
push 10
eq
or
```

Let x=2, y=10



Architecture:-



→All these architecture will be implemented in software.

Programming in virtual Machine:-

- Instruction set (Command)
- Memory management

Command in

- Arithmetic command
- Memory acces command
- program/control flow command
- function calling command.

Arithmetic & Logic Command:-

add	$x+y$
sub	$x-y$
neg	$-x$
eq	if $x=y$ then -1 else 0
gt	if $x>y$ then -1 else 0
lt	if $x<y$ then -1 else 0
and	$x \& y$
or	$x \mid y$

Memory Access command :-

- Push segment index
  - [sp] ← segment[in
  - sp ← sp-1
- pop segment index
  - sp ← sp+1
  - segment[index] ← [sp]

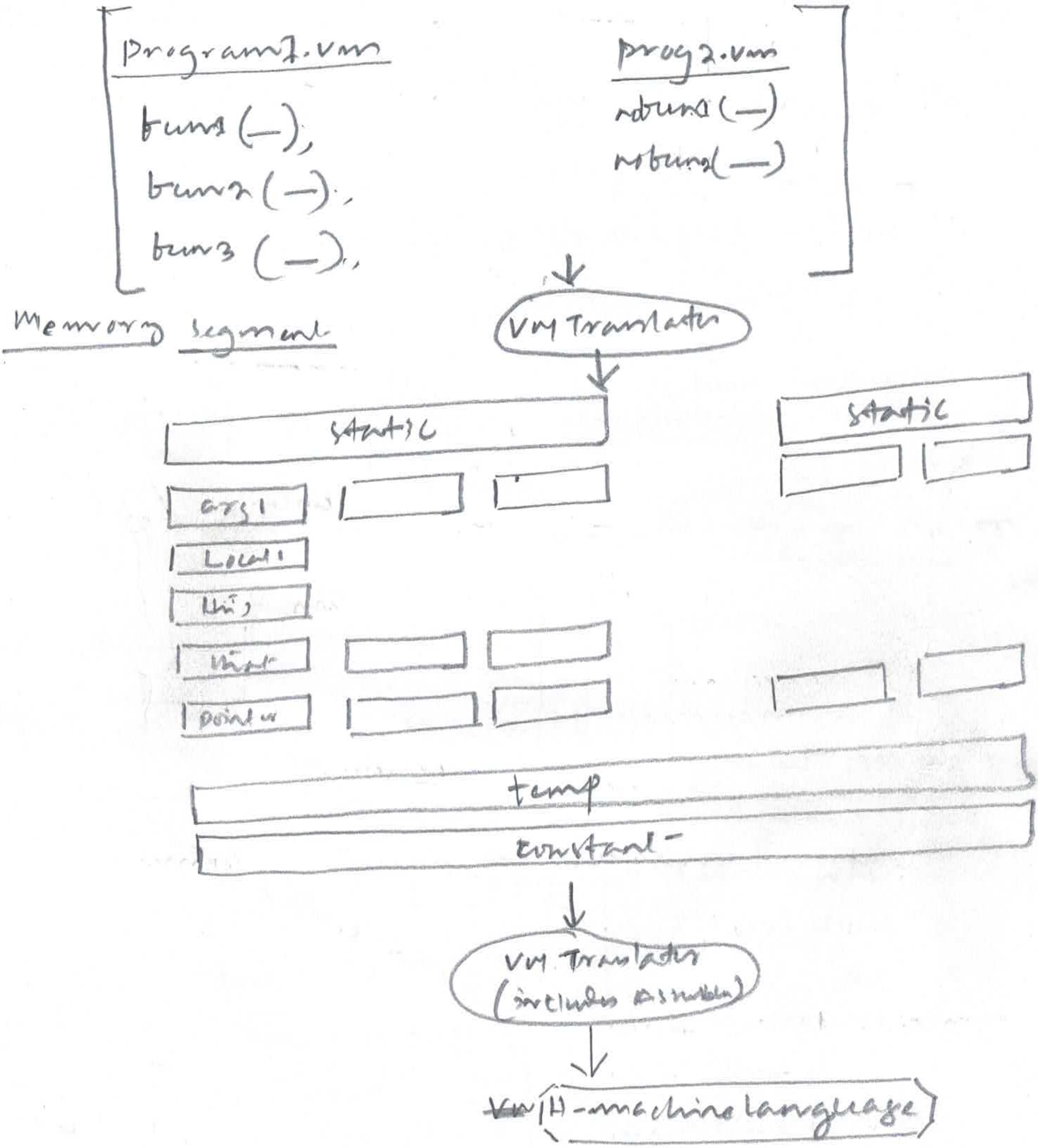
Partitioning memory into different segment :-

- To manage the program (multiple program) precisely.

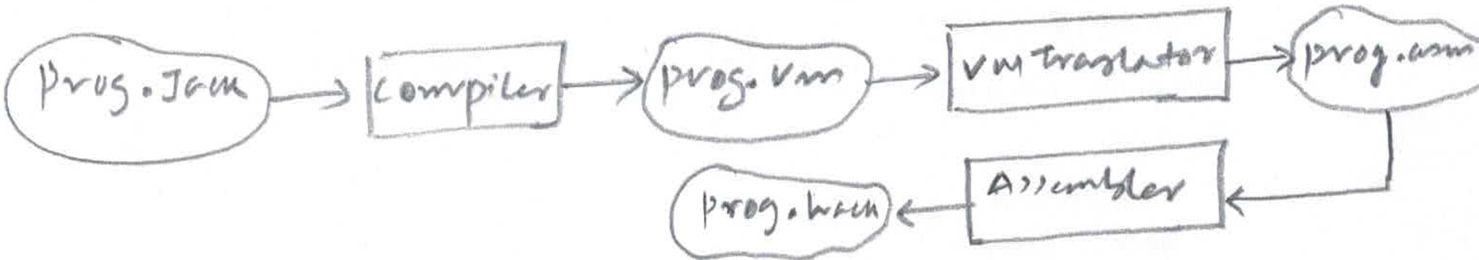
Memory segment (H-VM) [virtual memory of VM]

Segment	Need
argument	parameters
Local	Local variables
static	Global variables
constant	storing constant values range (0 - 32717)
this/that	Heap kind of ~~store~~ memory (Dynamic memory allocation
pointer	Base address of this/that
temp	temporary variables (Dynamic)

Virtual Memory Scenario or VMachine :-



Program Elements :-



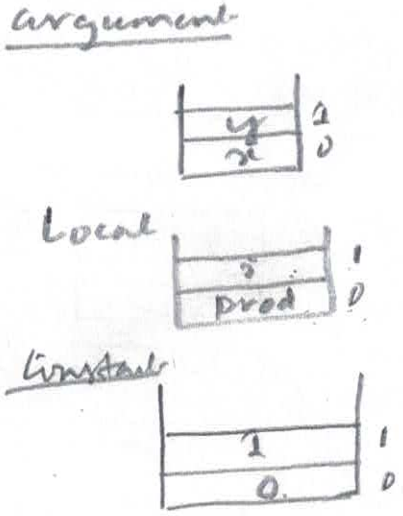
(8)

Example of program execution :-

C-program :

```
int  mult( int x, inty) {
        int prod = 0 ;
    for ( int i = y ; ~~i~~ i != 0 ; i--)
            prod = prod + x ;

    return prod ;
}
```



corresponding VM-code :-

```
    push constant 0    ]
    pop  local  0      ]
    push argument 1    ]  variable
    pop  local 1       ]      assignment
Label loop
    push constant 0    ]
    push local 1       ]  -> i != 0
    eq                 ]
    if-goto end        ]
    push local 0       ]  -> prod = prod + x
    push argument 0    ]
    add                ]
    pop local 0        ]
    push local 1       ]
    push constant 1    ]  i = i-1
    sub                ]
    pop local 1        ]
    goto  loop
```

```
label end
    push local 0  ]  return
    return        ]    prod,
```