Analysis of vector operations

Christophe Rhodes

Motivation

- understand consequence of data structure design decisions
- simple example of random-access model and big-O notation

Implementation details

Here we are thinking as the data structure implementor, not the data structure user

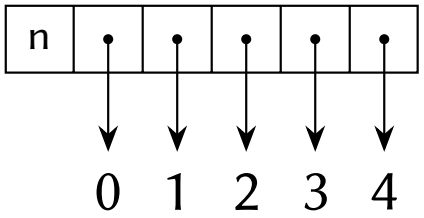
- look "behind the curtain" of the data structure
- implementing operations, so we can't use them!

Primitives:

- MREF(n) $\Longrightarrow \mathbb{Z}$
- ALLOC(n) $\Longrightarrow \mathbb{Z}$ (and allocates memory)
- arithmetic

Constructor

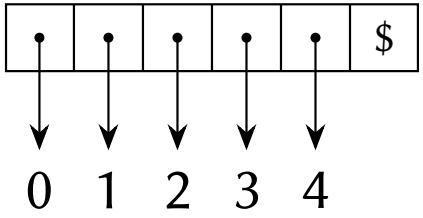
length-data



function VNEW(n)
    v ← ALLOC(n+1) ▷ 2
    MREF(v) ← n ▷ 1
    return v ▷ 1
end function

$\Rightarrow \Theta(1)$

sentinel

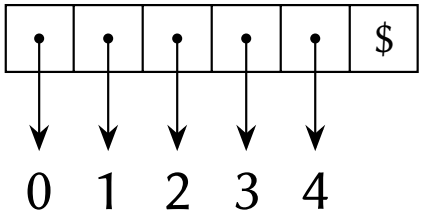


function VNEW(n)
    v ← ALLOC(n+1) ▷ 2
    MREF(v+n) ← $ ▷ 2
    return v ▷ 1
end function

$\Rightarrow \Theta(1)$

Dereference

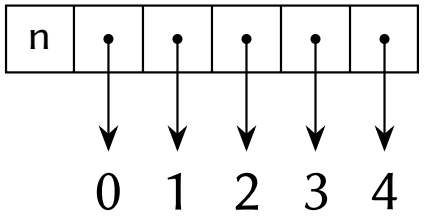
sentinel




```
function [](v,i)
    return MREF(v+i)        ▷ 2
end function
```

$\Longrightarrow \Theta(1)$

length-data

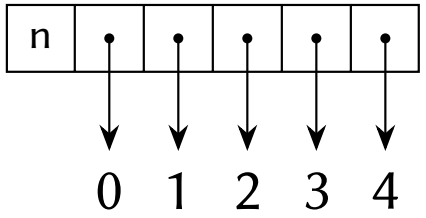



```
function [](v,i)
    return MREF(v+i+1)      ▷ 3
end function
```

$\Longrightarrow \Theta(1)$

Length

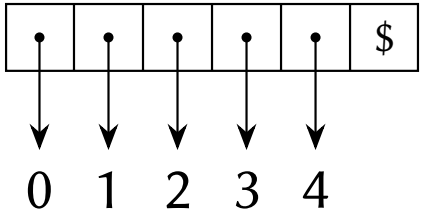
length-data




```
function LENGTH(v)
    return MREF(v)   ▷ 1
end function
```

$\Longrightarrow \Theta(1)$

sentinel




```
function LENGTH(v)
    l ← 0                                  ▷ 1
    while MREF(v+l) ≠ $ do                 ▷ 3n+3
        l ← l + 1                          ▷ 2n+2
    end while
    return l
end function
```

$\Longrightarrow \Theta(n)$