EÖTVÖS LORÁND
UNIVERSITY | BUDAPEST

Custom Enumerators
Solving Concrete Problems

Object Oriented Programming | 2024 Spring

Practice 7

By Tarlan Ahadli

Problem 1

Find the longest word in a given sequential input file that contains character 'w'. Words are separated by spaces from each other.

Specification:

A = (x:infile($\mathbb{K}$), l:$\mathbb{L}$, longest:$\mathbb{S}$)

Pre = ($x = x_0$)

Problem 1 | Enumerator Based Specification

Idea:

Enumerate the words along with their length and a boolean value indicating whether the word contains 'w'.

Conditional maximum search

f(e)	~	|e.word |
cond(e)	~	e.w
H, >	~	$\mathbb{N}$, >

New specification:

A = (t:enor(WORD), l:$\mathbb{L}$, longest:$\mathbb{S}$)

WORD = rec(word:$\mathbb{S}$, w:$\mathbb{L}$)

Pre = (t = t_0)

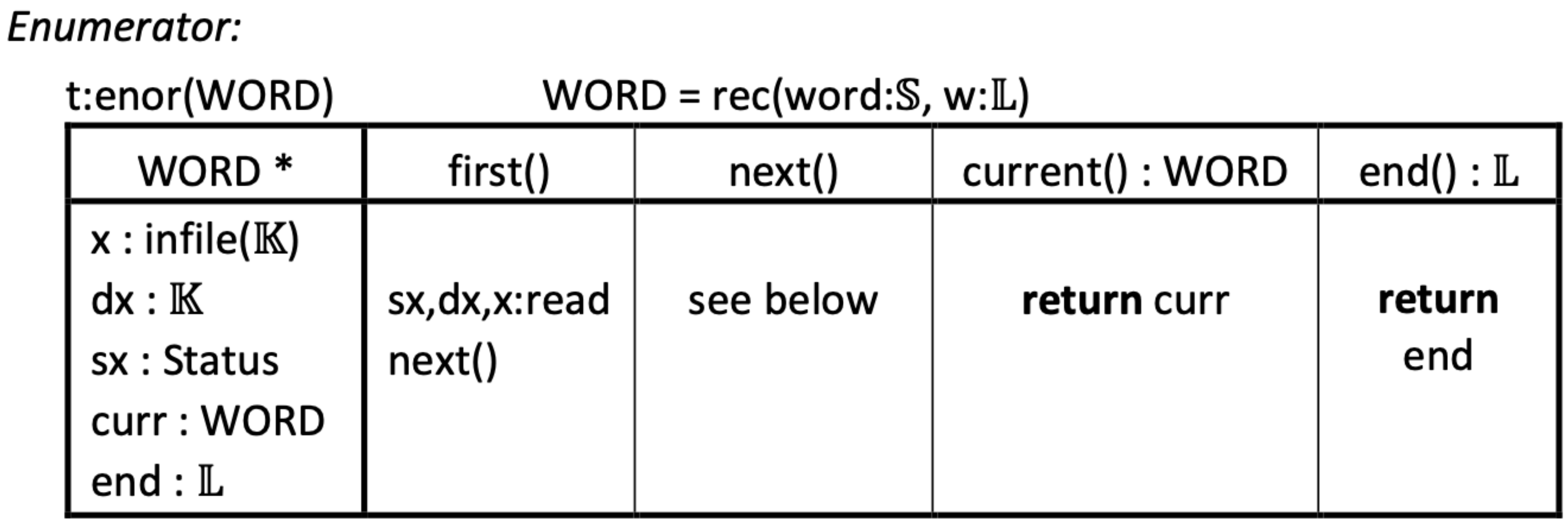
Post = ((l, max, elem) = $\mathbf{MAX}_{\substack{e\in t_0 \\ e.w}}$ |e.word | $\wedge$ l $\rightarrow$ longest =elem.word)

Problem 1 | Structogram

Algorithm:

```
l := false; t.first()
¬t.end()
    ¬ t.current().w
        —
    t.current().w ∧ l
        |t.current().word | > max
            max, longest :=
            |t.current().word |,
            t.current().word
        else
            —
    t.current().w ∧ ¬l
        l, max, longest := true,
        |t.current().word |,
        t.current().word
    t.next()
```

Problem 1 | Enumerator



Enumerator:

t:enor(WORD) WORD = rec(word:$\mathbb{S}$, w:$\mathbb{L}$)

WORD *	first()	next()	current() : WORD	end() : $\mathbb{L}$

x : infile($\mathbb{K}$)
dx : $\mathbb{K}$
sx : Status
curr : WORD
end : $\mathbb{L}$
	sx,dx,x:read
	next()
		see below	**return** curr	**return**
				end

Problem 1 | Enumerator | Next

next() method

A = (x:infile($\mathbb{K}$), dx:$\mathbb{K}$, sx:Status, curr:WORD, end:$\mathbb{L}$)

Pre = (x = x' $\wedge$ dx = dx' $\wedge$ sx = sx')

$Post$ = ((dx'',(sx'',dx'',x''))=$\text{SELECT}_{dx\in(dx',x')}$ (sx=abnorm $\vee$ dx≠' ')

$\wedge$ end =(sx''=abnorm)

$\wedge$ ($\neg$end $\rightarrow$ (curr.word, (sx,dx,x)) = $\bigoplus_{dx\in(dx'',x'')}^{dx \neq ' '}$<dx> $\wedge$ (curr.w, (sx,dx,x)) = $\bigvee_{dx\in(dx'',x'')}^{dx \neq ' '}$dx='w'))

Problem 1 | Enumerator | Next | Algorithm

Selection

t:enor(E)	~	x:infile($\mathbb{K}$) (sx,dx,x:read) without first()
cond(e)	~	$sx=abnorm \vee dx \neq ' '$

Two summations (concatenation and OR'ing)

t:enor(E)	~	x:infile($\mathbb{K}$) (sx,dx,x:read) without first(), as long as: $dx \neq ' '$
f(e)	~	(<dx>, dx='w')
s	~	(curr.word, curr.w)
H, +, 0	~	$(\mathbb{K}^*, \mathbb{L}), (\oplus, \vee)$, (<>, false)

```
while $sx=norm \wedge dx=' '$
    sx,dx,x:read
end := sx=abnorm
if $\neg end$
    curr.word, curr.w := <>, false
    while $sx=norm \wedge dx \neq ' '$
        curr.word, curr.w :=
        $curr.word \oplus <dx>$,
        $curr.w \vee (dx='w')$
        sx,dx,x:read
else
    –
```

Problem 2

Given a file containing data of huntings. Each line of the file consists of the name of the hunter, the date of the hunting, the species and weight of the animal shot by the given hunter at the given hunting. The file is sorted by hunter and then by date. Decide, whether every hunter has shot a bear at any of his/her hunting.

Specification:

$A = (x:\text{infile}(\text{Trophy}), l:\mathbb{L})$

$\text{Trophy} = \text{rec}(\text{name}:\mathbb{S}, \text{date}:\mathbb{S},$
$\text{species}:\mathbb{S}, \text{weight}:\mathbb{N})$

$Pre = (x=x_0 \land x\nearrow_{(\text{name},\text{date})})$

Problem 2 | Enumerator Based Specification

New specification:

$A = (t:enor(\mathbb{L}), l:\mathbb{L})$

Pre $= (t=t_0)$

Post $= (l = \forall SEARCH_{e \in t_0}\, e)$

linear search

cond(e) ~ e

Idea:

Enumerate as many boolean values as the number of the hunters, a boolean value is true in case the related hunter has shot a bear.

Algorithm:

```
l := true
t.first()
l ∧ ¬t.end()
    l := t.current()
    t.next()
```

Problem 2 | Enumerator

Enumerator:

t:enor($\mathbb{L}$)

$\mathbb{L}$*	first()	next()	current() : $\mathbb{L}$	end() : $\mathbb{L}$
x : infile(Trophy) dx : Trophy sx : Status curr : $\mathbb{L}$ end : $\mathbb{L}$	sx,dx,x:read next()	see below	**return** curr	**return** end

Trophy = rec(name:$\mathbb{S}$, date:$\mathbb{S}$, species:$\mathbb{S}$, weight:$\mathbb{N}$)

Problem 2 | Enumerator | Next

next() method

A = (x:infile(Trophy), dx: Trophy, sx:Status, curr:$\mathbb{L}$, end:$\mathbb{L}$)

Pre = ($x = x' \wedge x\nearrow_{(name,date)} \wedge dx = dx' \wedge sx = sx'$) $dx.name = dx'.name$

$Post$ = ($end = (sx'=abnorm) \wedge (\neg end \rightarrow (curr, (sx,dx,x)) = \bigvee_{dx\in(dx',x')} curr.species=\text{"bear"})$)

Summation (OR'ing)

t:enor(E)	~	x:infile(Trophy) (sx,dx,x:read) without first(), as long as: dx.name=dx'.name
f(e)	~	dx.species="bear"
s	~	curr
H, +, 0	~	$\mathbb{L}$, $\vee$, false

```
end := sx=abnorm
if ¬end
    curr := false
    n := dx.name
    while sx=norm ∧ dx.name=n
        curr := curr ∨ (dx.species="bear")
        sx,dx,x:read
else
    –
```