Assignment 2

Neptun: ULLBQG **Name: Rizwan Hussain**

Assignment No: 8

Task

The results of the National Angler's Championship is stored in a text file. Each line of the file contains the identifier of the participant and the championship (strings without whitespace), and the list of the caught fish, which are stored as pairs: (the kind of the fish, the size of the fish). The kind of the fish is a string without whitespace, its size is a natural number. The data in a line are separated by whitespace. The lines of the text file are sorted according to the name of the participants. You can assume that the text file is correct.
An example for a line of the text file:
James BigLakeChampionship Tuna 50 Salmon 20 Sardine 5 Tuna 100

Plan of the main program

A = (t : Enor(Angler), angler : String, cnt : Integer)
*Angler = **rec**(name:String, tuna_cnt: Integer)*
Pre =(t = t')
Post =($\text{angler}, \text{cnt} = \underset{e\in t'}{\text{MAX}}(e.\text{tuna\_cnt})$*)*

```
angler:= “”
cnt:= 0
t.first()
¬t.end()
    t.current().tuna_cnt > cnt
        cnt, angler:=
        t.current().tuna_cnt,
        t.current().name
    else
        –
    t.next()
```

Maximum Selection(Analogy)

t: Enor(E)	~	t: Enor(Angler)
f(e)	~	e.tuna_cnt
H, <	~	N, <

Enumerator of Anglers

enor(Angler)	*first(), next(), current(), end()*

ED :*enor*(*Data*)
cur : *Angler*
end : $\mathbb{L}$
	first() ~ *ED.first(); ED.next()*
	next() ~ *see below*
	current() ~ *cur*
	end() ~ *end*

Operation *next()* of *Enor(*Angler*)* has to solve the following problem:

Get the next angler's name and then add number of Tuna it has killed into the tuna_cnt which is initialized at 0. The anglers are enumerated along with number of Tuna they have caught. Which results in a data = rec(name: String, champ : String , tunas : Integer). The first competition is stored in ED.current(), so there is no need for ED.first() or ED.next() at first. ED.next() operation is used to read the championships of the same angler and enumeration lasts as long as this reading is not finished.

A^{next} = (ED:enor(Data), end:$\mathbb{L}$, cur:Angler)

Pre^{next} = (ED = ED^1)

$Post^{next}$ = $(end = ED.end() \wedge \neg end \rightarrow \text{cur.tuna\_cnt} = \sum_{e \in t'}^{\text{cur.name=e.name}} (\boldsymbol{e}.\text{tunas}))$)

next()

```
$end := ED.end()$
$\neg end$
    $cur.name := ED.current().name$
    $cur.tuna\_cnt := 0$
    $\neg ED.end() \wedge ED.current().name = cur.name$
        $cur.tuna\_cnt := cur.tuna\_cnt + ED.current().tunas$
        $ED.next()$
    SKIP
```

Summation(Analogy)

t: Enor(E)	~	ED : Enor(Data) without first(), as long as cur.name = ED.current().name
f(e)	~	e.tunas
s	~	cur.tuna_cnt
H , + ,0	~	N, +, 0

Enumerator of Data

enor(Data)	*first(), next(), current(), end()*
f : *infile*(*Line*) *curr* : *Data* *end* : $\mathbb{L}$	*first()* ~ *see below* *next()* ~ *see below* *current()* ~ *curr* *end()* ~ *end*

Operations *first()* and *next()* of *Enor(Data)* are the same and they have to solve the following problem: Read the next line of the input file *f*. If there are no more lines, then variable *end* should be true. If there are more lines, then get the name of the angler and the name of the championship and count the word „Tuna”.

A^{next} = (f: infile(Line), end:$\mathbb{L}$, curr:Data)
Line = seq(Word)
Pre^{next} = ($f = f^1$)
$Post^{next}$ = ($sf, df, f = read(f^1) \wedge end=(sf=abnorm) \wedge$
$\neg end \rightarrow curr.name$ = “first word of df” $\wedge$
curr.champ = “second word of df” $\wedge$
curr.tunas = “number of word ’Tuna’ in df”)

In the implementation, the two classes of the two above enumerator objects (*t* and *ED*) are placed into separate compilation units.

Testing plan

Three algorithmic patterns are used in the solution: maximum search, summation, and counting.

A. Test cases for searching the angler who caught most tuna (maximum search): based on the **length of the interval**:

1. Empty file.
2. One angler.
3. More anglers.

based on the **beginning and the end of the interval**:

4. First angler has caught most Tuna.
5. Last angler has caught most Tuna.

based on the **pattern**:

1. There are more than one anglers who caught same number of Tuna.
2. No angler has caught a Tuna.

B. Test cases of summing up the Tunas of an angler(summation):
based on the **length of the interval**:

1. No catch.
2. One competition of one angler.
3. More competitions of one angler.

based on the **beginning and the end of the interval**:

4. An angler did not catch Tuna on his first competition, but he did catch on the rest.
5. An angler did not catch Tuna on his last competition, but he did catch on the rest.

based on the **pattern**:

4. He did not catch Tuna on any competition.
5. He caught Tuna on one of the competitions.
6. He caught Tuna on some of the competitions.
7. He caught Tuna on every competition.

C. Test cases of counting the Tunas on the competition of one angler (counting):
based on the **length of the interval**:

1. A line without catches.
2. A line of one catch.
3. A line of more catches.

based on the **beginning and the end of the interval**:

4. A line the first catch of which is a Tuna.
5. A line the last catch of which is a Tuna.

based on the **pattern**:

7. A line without caught Tuna.
8. A line containing one caught Tuna.
9. A line containing more caught Tunas.