Fundamentals of Programming I

3E Sequence Examples

SUPPLEMENT II

Grado en Ingeniería Informática

Luis Hernández Yáñez
Facultad de Informática
Universidad Complutense

Index

Fundamentals of Programming I

Traversal

Luis Hernández Yáñez

A Parking Lot

Sequence of characters I and O in every line of a text file

I = A car enters the parking lot; O = A car exits the parking lot

How many cars remain in the parking lot at the end of the day?

Several cases, each one in one line and ending with a period

End of the cases: a line with only a period

parking.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
IIIIIIIIOIIOIIIIIOIIOOIOOIIIOIIOOOIOOOIIIOOOOIOOIIIOOIOOOOO.
IIIOOIOOOIIOOOOOOIIIOIOOOOIOOIIOIIIOO.
IIIIOIIIOIOIIIOOIOOOIOOOOOIIOOOIIIIOOIIIOOOO.
.
```

Luis Hernández Yáñez

A Parking Lot

```
#include <iostream>
#include <fstream>
using namespace std;

int main() {
   int cars;
   char c;
   bool end = false;
   ifstream file;
   file.open("parking.txt");
   if (!file.is_open())
      cout << "Couldn't open the file!" << endl;
   else
      // Traversal...
      file.close();
   return 0;
}
```


A Parking Lot (Traversal)

```
while (!end) {
   file >> c;
   if (c == '.') // . (sentinel) as first character?
      end = true;
   else {
      cars = 0;
      while (c != '.') { // Sequence traversal...
         cout << c;
         if (c == 'I')
            cars++;
         else if (c == 'O')
            cars--;
         file >> c;
      }
```

.../...

A Parking Lot (Traversal)

parking.cpp

```
        if (cars >= 0)
           cout << endl << cars << " cars left";
        else
           cout << endl << "Error: More outputs than inputs!";
        cout << endl;
     }
  }
```


Well-Matched Parentheses?

Every parenthesis with its partner

Character sequence ending with # and with open/close parenthesis:

a b (c (d e) f g h ((i (j k)) l m n) o p) (r s) #

Counter for nesting level:

When '(' is found, increment it – When ')' is found, decrement it

At the end the counter must have the value 0

Errors:

- Counter gets -1: An open parenthesis is missing → abc)de(fgh(ij))#
- Counter ends with a positive value: Close parentheses missing
  (a(b(cd(e)f)gh(i))jk#

Well-Matched Parentheses?

An error should interrupt the traversal:

```
char c;
int nesting = 0, pos = 0;
bool error = false;
cin >> c;
while ((c != '#') && !error) {
   pos++;
   if (c == '(')
       nesting++;
   else if (c == ')')
      nesting--;
   if (nesting < 0)
      error = true;
   if (!error)
      cin >> c;
}
```

.../...

Well-Matched Parentheses?

parenthesis.cpp

```
if (error)
   cout << "Error: Closing without an opening (pos. " << pos
        << ")";
else if (nesting > 0)
   cout << "Error: Opening without a closing";
else
   cout << "Correct";
cout << endl;
```

```
C:\FP1\Lesson3>parenthesis
ab(c(de)fgh((i(jk))lmn)op)(rs)#
Correct

C:\FP1\Lesson3>parenthesis
)ab(((cd)ef))#
Error: Closing without an opening (pos. 1)

C:\FP1\Lesson3>parenthesis
(((abc(d)e(fg(h)))#
Error: Opening without a closing
```

Are Two Sequences the Same?

equal.cpp

```
bool equal() {
   bool areEqual = true;
   double d1, d2;
   ifstream seq1, seq2;
   bool final = false;
   seq1.open("sequence1.txt");
   seq2.open("sequence2.txt");
   seq1 >> d1;
   seq2 >> d2; // At least the files contain the sentinels (0)
   while (areEqual && !final) {
      areEqual = (d1 == d2);
      final = ((d1 == 0) || (d2 == 0));
      if (!final) {
         seq1 >> d1;
         seq2 >> d2;
      }
   }
   seq1.close();
   seq2.close();
   return areEqual;
}
```

Change `sequence2.txt` to `sequence3.txt` or to `sequence4.txt` to test other cases

Prime Numbers Less Than N

primes.cpp

Calculated sequence: Numbers divisible only by 1 and themselves (< *N*)

```
#include <iostream>
using namespace std;
bool prime(int n);
int main() {
   int num, candidate;
   cout << "Integer to stop (>1): ";
   cin >> num;
   if (num > 1) {
      candidate = 2; // 1 is not considered a prime number
      while (candidate < num) {
         cout << candidate << " "; // Show prime number
         candidate++;
         while (!prime(candidate)) // Next prime number
            candidate++;
      }
   }
   return 0;
}
```

Prime Numbers Less Than N

```
bool prime(int n) {
   bool isPrime = true;

   for (int i = 2; i <= n - 1; i++)
      if (n % i == 0)
         isPrime = false; // Divisible by i

   return isPrime;
}
```


Prime Numbers Less Than N

primes2.cpp

Improvements:

Test only odd numbers

Only divisible by odd numbers

Can't be divisible by numbers higher than its half

```
candidate = 2;
cout << candidate << " ";
candidate++; // We follow with 3 (prime)
while (candidate < num) {
   cout << candidate << " ";
   candidate = candidate + 2;
   while (!prime(candidate))
      candidate = candidate + 2;
} ...
```

```
bool prime(int n) {
   bool isPrime = true;
   for (int i = 3; i <= n / 2; i = i + 2)
      if (n % i == 0)
         isPrime = false;
   return isPrime;
}
```

Prime Numbers Less Than N

primes3.cpp

Another improvement: stop when the first divisor is found

```
bool prime(int n) {
   bool isPrime = true;

   int i = 3;
   while ((i <= n / 2) && isPrime) {
      if (n % i == 0)
         isPrime = false;
      i = i + 2;
   }

   return isPrime;
}
```


Fundamentals of Programming I

Searching

Searching for a Number in a File

searchfile.cpp

```
#include <iostream>
#include <fstream>
using namespace std;

int search(int n);
// Returns the line where found, or -1 if not found

int main() {
   int num, line;

   cout << "Value to locate: ";
   cin >> num;
   line = search(num);
   if (line != -1)
      cout << "Found in line " << line << endl;
   else
      cout << "Not found!" << endl;

   return 0;
}
```


Searching for a Number in a File

```
int search(int n) {
   int i, line = 0;
   bool found = false;
   ifstream file;
   file.open("integers.txt");
   if (!file.is_open())
      line = -1;
   else {
      file >> i;
      while ((i != 0) && !found) {
         line++;
         if (i == n)
            found = true;
         file >> i;
      }
      if (!found)
         line = -1;
      file.close();
   }
   return line;
}
```

integers.txt:

Archivo Edic

194
78
2
159
41
73
123
175
82
24
265
16
153
9
164
15
231
5
0

Sentinel

Searching in Ordered Sequences

Luis Hernández Yáñez

Searching in Ordered Sequences

searchord.cpp

Ordered sequence (lesser to greater): Stop when one equal to or greater is found

The remaining are higher: *it can't be the one searched!*

```
cout << "Value to locate: ";
cin >> num;
file >> i;
while ((i != 0) && (i < num)) {
   cont++;
   file >> i;
}
if (i == num)
   cout << "Found (pos.: " << cont << ")";
else
   cout << "Not found!";
cout << endl;
file.close();
```

ordered.txt:

Archivo Edici

2
5
9
15
16
24
41
73
78
82
123
153
159
164
175
194
231
265
0

Luis Hernández Yáñez

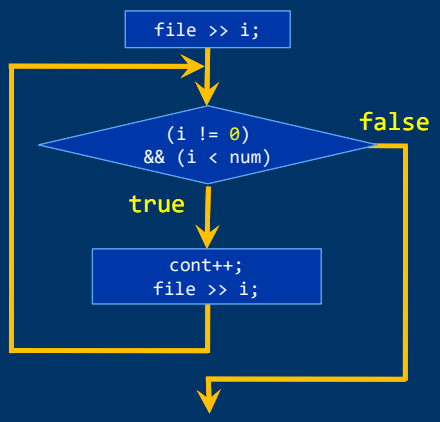
Ordered Sequences

If the element is in the sequence: Same as non-ordered sequences

2 5 9 15 16 24 41 73 78 82 123 153 159 ...

num 9

i 9



Ordered Sequences

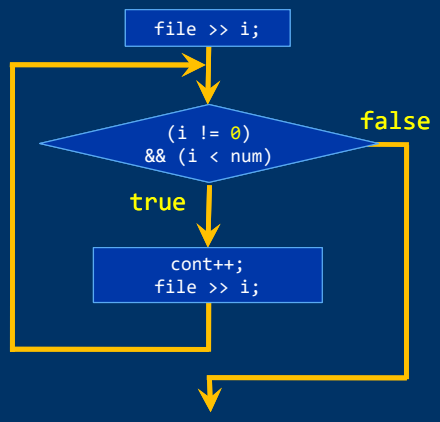
If the element is not in the sequence: Avoid searching the rest

2 5 9 15 16 24 41 73 78 82 123 153 159 ...

The rest of the sequence is not processed

num 10

i 15



Promote Open Culture!

Creative Commons License