Coding Conventions

Panos Louridas

Department of Management Science and Technology
Athens University of Economics and Business

Greek Research and Education Network

17/07/2008

Contents

Contents

Contents

Contents

Contents

Java Layout Example #1

```
1  /* Use the insertion sort technique to sort the "data" array in ascending order.
2  This routine assumes that data[ firstElement ] is not the first element in data and
3  that data[ firstElement-1 ] can be accessed. */ public void InsertionSort( int[]
4  data, int firstElement, int lastElement ) { /* Replace element at lower boundary
5  with an element guaranteed to be first in a sorted list. */ int lowerBoundary =
6  data[ firstElement-1 ]; data[ firstElement-1 ] = SORT_MIN; /* The elements in
7  positions firstElement through sortBoundary-1 are always sorted. In each pass
8  through the loop, sortBoundary is increased, and the element at the position of the
9  new sortBoundary probably isn't in its sorted place in the array, so it's inserted
10 into the proper place somewhere between firstElement and sortBoundary. */ for (
11 int sortBoundary = firstElement+1; sortBoundary <= lastElement; sortBoundary++ )
12 { int insertVal = data[ sortBoundary ]; int insertPos = sortBoundary; while (
13 insertVal < data[ insertPos-1 ] ) { data[ insertPos ] = data[ insertPos-1 ];
14 insertPos = insertPos-1; } data[ insertPos ] = insertVal; } /* Replace original
15 lower-boundary element */ data[ firstElement-1 ] = lowerBoundary; }
```

Source: Steve McConnell, Code Complete, 2^{nd} edition.

Java Layout Example #2

```
1  /* Use the insertion sort technique to sort the "data" array in ascending order.
2  This routine assumes that data[ firstElement ] is not the first element in data and
3  that data[ firstElement-1 ] can be accessed. */ public void InsertionSort( int[]
4  data, int firstElement, int lastElement ) { /* Replace element at lower boundary
5  with an element guaranteed to be first in a sorted list. */ int lowerBoundary =
6  data[ firstElement-1 ]; data[ firstElement-1 ] = SORT_MIN; /* The elements in
7  positions firstElement through sortBoundary-1 are always sorted. In each pass
8  through the loop, sortBoundary is increased, and the element at the position of the
9  new sortBoundary probably isn't in its sorted place in the array, so it's inserted
10 into the proper place somewhere between firstElement and sortBoundary. */ for (
11 int sortBoundary = firstElement+1; sortBoundary <= lastElement; sortBoundary++ )
12 { int insertVal = data[ sortBoundary ]; int insertPos = sortBoundary; while (
13 insertVal < data[ insertPos-1 ] ) { data[ insertPos ] = data[ insertPos-1 ];
14 insertPos = insertPos-1; } data[ insertPos ] = insertVal; } /* Replace original
15 lower-boundary element */ data[ firstElement-1 ] = lowerBoundary; }
16 /* Use the insertion sort technique to sort the "data" array in ascending
17 order. This routine assumes that data[ firstElement ] is not the
18 first element in data and that data[ firstElement-1 ] can be accessed. */
19 public void InsertionSort( int[] data, int firstElement, int lastElement ) {
20 /* Replace element at lower boundary with an element guaranteed to be first in a
21 sorted list. */
22 int lowerBoundary = data[ firstElement-1 ];
23 data[ firstElement-1 ] = SORT_MIN;
24 /* The elements in positions firstElement through sortBoundary-1 are
25 always sorted. In each pass through the loop, sortBoundary
26 is increased, and the element at the position of the
27 new sortBoundary probably isn't in its sorted place in the
28 array, so it's inserted into the proper place somewhere
29 between firstElement and sortBoundary. */
30 for (
31 int sortBoundary = firstElement+1;
32 sortBoundary <= lastElement;
33 sortBoundary++
34 ) {
35 int insertVal = data[ sortBoundary ];
36 int insertPos = sortBoundary;
37 while ( insertVal < data[ insertPos-1 ] ) {
38 data[ insertPos ] = data[ insertPos-1 ];
39 insertPos = insertPos-1;
40 }
41 data[ insertPos ] = insertVal;
42 }
43 /* Replace original lower-boundary element */
44 data[ firstElement-1 ] = lowerBoundary;
45 }
```

Source: Steve McConnell, Code Complete, 2^{nd} edition.

Java Layout Example #3

```
1   /* Use the insertion sort technique to sort the "data" array in ascending
2   order. This routine assumes that data[ firstElement ] is not the
3   first element in data and that data[ firstElement-1 ] can be accessed.
4   */
5
6   public void InsertionSort( int[] data, int firstElement, int lastElement ) {
7      // Replace element at lower boundary with an element guaranteed to be
8      // first in a sorted list.
9      int lowerBoundary = data[ firstElement-1 ];
10     data[ firstElement-1 ] = SORT_MIN;
11
12     /* The elements in positions firstElement through sortBoundary-1 are
13     always sorted. In each pass through the loop, sortBoundary
14     is increased, and the element at the position of the
15     new sortBoundary probably isn't in its sorted place in the
16     array, so it's inserted into the proper place somewhere
17     between firstElement and sortBoundary.
18     */
19     for ( int sortBoundary = firstElement + 1; sortBoundary <= lastElement;
20        sortBoundary++ ) {
21        int insertVal = data[ sortBoundary ];
22        int insertPos = sortBoundary;
23        while ( insertVal < data[ insertPos - 1 ] ) {
24           data[ insertPos ] = data[ insertPos - 1 ];
25           insertPos = insertPos - 1;
26        }
27        data[ insertPos ] = insertVal;
28     }
29
30     // Replace original lower-boundary element
31     data[ firstElement - 1 ] = lowerBoundary;
32  }
```

Source: Steve McConnell, Code Complete, 2nd edition.

Java Layout Example #4

```
1  \u0070\u0075\u0062\u006c\u0069\u0063\u0020\u0020\u0020\u0020
2  \u0063\u006c\u0061\u0073\u0073\u0020\u0055\u0067\u006c\u0079
3  \u007b\u0070\u0075\u0062\u006c\u0069\u0063\u0020\u0020\u0020
4  \u0020\u0020\u0020\u0020\u0073\u0074\u0061\u0074\u0069\u0063
5  \u0076\u006f\u0069\u0064\u0020\u006d\u0061\u0069\u006e\u0028
6  \u0053\u0074\u0072\u0069\u006e\u0067\u005b\u005d\u0020\u0020
7  \u0020\u0020\u0020\u0020\u0061\u0072\u0067\u0073\u0029\u007b
8  \u0053\u0079\u0073\u0074\u0065\u006d\u002e\u006f\u0075\u0074
9  \u002e\u0070\u0072\u0069\u006e\u0074\u006c\u006e\u0028\u0020
10 \u0022\u0048\u0065\u006c\u006c\u006f\u0020\u0077\u0022\u002b
11 \u0022\u006f\u0072\u006c\u0064\u0022\u0029\u003b\u007d\u007d
```

Source: Joshua Bloch and Neal Gafter, Java Puzzler: Traps, Pitfalls, and Corner Cases.

Java Layout Example #5

```
1   public
2   class Ugly
3   {public
4       static
5   void main(
6   String[]
7       args){
8   System.out
9   .println(
10  "Hello w"+
11  "orld");}}
```

Source: Joshua Bloch and Neal Gafter, Java Puzzler: Traps, Pitfalls, and Corner Cases.

Java Layout Example #6

```
1  public class Ugly {
2      public static void main(String[] args) {
3          System.out.println("Hello w" + "orld");
4      }
5  }
```

Source: Joshua Bloch and Neal Gafter, Java Puzzler: Traps, Pitfalls, and Corner Cases.

A Part of Internet Lore

```
#include <stdio.h>
main(t,_,a)char *a;{return!0<t?t<3?main(-79,-13,a+main(-87,1-_,
main(-86,0,a+1)+a))1,t<_?main(t+1,_,a):3,main(-94,-27+t,a)&&t==2?_<13?
main(2,_+1,"%s %d %d\n"):9:16:t<0?t<-72?main(_,t,
"@n'+,#'/*{}w+/w#cdnr/+,{}r/*de}+,/*{*+,/w{%+,/w#q#n+,/#{l,+,/n{n+,/+#n+,/#\
;#q#n+,/+k#;*+,/'r :'d*'3,}{w+K w'K:'+}e#';dq#'l \
q#'+d'K#!/+k#;q#'r}eKK#}w'r}eKK{nl]'/#;#q#n')(){}#}w'){){nl]'/+#n';d}rw' i;# \
){nl]!/n{n#'; r{#w'r nc{nl]'/#{l,+'K {rw' iK{;[{nl]'/w#q#n'wk nw' \
iwk{KK{nl]!/w{%'l##w#' i; :{nl]'/*{q#'ld;r'}{nlwb!/*de}'c \
;;{nl'-{}rw]'/+,}##'*}#nc,',#nw]'/+kd'+e}+;#'rdq#w! nr' ') }+}{rl#'{n' ')# \
}'+}##(!!/")
:t<-50?==*a?putchar(31[a]):main(-65,_,a+1):main((*a=='/')+t,_,a+1)
  :0<t?main(2,2,"%s"):*a=='/'||main(0,main(-61,*a,
"!ek;dc i@bK'(q)-[w]*%n+r3#l,{}:\nuwloca-O;m .vpbks,fxntdCeghiry"),a+1);}
```

Source: Wikipedia

A Part of Internet Lore, Output

On the first day of Christmas my true love gave to me
a partridge in a pear tree.

On the second day of Christmas my true love gave to me
two turtle doves
and a partridge in a pear tree.

On the third day of Christmas my true love gave to me
three french hens, two turtle doves
and a partridge in a pear tree.

On the fourth day of Christmas my true love gave to me
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the fifth day of Christmas my true love gave to me
five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the sixth day of Christmas my true love gave to me
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the seventh day of Christmas my true love gave to me
seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the eighth day of Christmas my true love gave to me
eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the ninth day of Christmas my true love gave to me
nine ladies dancing, eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the tenth day of Christmas my true love gave to me
ten lords a-leaping,
nine ladies dancing, eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the eleventh day of Christmas my true love gave to me
eleven pipers piping, ten lords a-leaping,
nine ladies dancing, eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the twelfth day of Christmas my true love gave to me
twelve drummers drumming, eleven pipers piping, ten lords a-leaping,
nine ladies dancing, eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

A Maze Generator

```
1  char M[2],A,Z,E=40,J[40],T[40];main(C){for(*J=A=scanf("%d",&C);
2  --E;J[E]=T[E]=E)printf(". ");for(;(A-=Z=!Z)||(printf("\n|"
3  ),A=39,C--);Z||printf(M))M[Z]=Z[A-(E=A[J-Z])&&!C&A==T[A]
4  |6<<27<rand()||!C&!Z?J[T[E]=T[A]]=E,J[T[A]=A-Z]=A," _." :"_|"];}
```

Source: Wikipedia

A Prime Number Finder

```
1  _(__,___,____){___/__<=1?_(__,___+1,____):!(___%__)?_(__,___+1,0):___%__==___/
2  __&&!____?(printf("%d\t",___/__),_(__,___+1,0)):___%__>1&&___%__<___/__?_(__,1+
3  ___,____+!(___/__%(___%__))):___<__*__?_(__,___+1,____):0;}main(){_(100,0,0);}
```

Source: Wikipedia

A Prime Number Finder, Decyphered

```
1   void primes(int cap) {
2     int i, j, composite;
3     for(i = 2; i < cap; i++) {
4       composite = 0;
5       for(j = 2; j < i; j++)
6         composite += !(i % j);
7       if(!composite)
8         printf("%d\t", i);
9     }
10  }
11
12  int main() {
13    primes(100);
14  }
```

Source: Wikipedia

Abandon All Sanity, Ye Who Enter Here: INTERCAL

- Designed in 1972 at Princeton by two students, Don Woods and James Lyon
- Goal: "...to have a compiler language which has nothing at all in common with any other major language. By major we meant anything with which the authors were at all familiar, e.g., FORTRAN, BASIC, COBOL, ALGOL, SNOBOL, SPITBOL, FOCAL, SOLVE, TEACH, APL, LISP and PL/I".
- "It is a well-known and oft-demonstrated fact that a person whose work is incomprehensible is held in high esteem. For example, if one were to state that the simplest way to store a value of 65536 in a 32-bit INTERCAL variable is:
  `DO :1 <- #0$#256`
  Any sensible programmer would say that that was absurd. Since this is indeed the simplest method, the programmer would be made to look foolish in front of his boss, who would of course have happened to turn up, as bosses are wont to do. The effect would be no less devastating for the programmer having been correct."

Source: Wikipedia and the INTERCAL Programming Language Revised Reference Manual

An Example Program in C

```
1  #include <stdio.h>
2
3  int main()
4  {
5      puts("Hello, world!");
6      return 0;
7  }
```

Equivalent in INTERCAL

```
DO ,1 <- #13
PLEASE DO ,1 SUB #1 <- #234
DO ,1 SUB #2 <- #112
DO ,1 SUB #3 <- #112
DO ,1 SUB #4 <- #0
DO ,1 SUB #5 <- #64
DO ,1 SUB #6 <- #194
DO ,1 SUB #7 <- #48
PLEASE DO ,1 SUB #8 <- #22
DO ,1 SUB #9 <- #248
DO ,1 SUB #10 <- #168
DO ,1 SUB #11 <- #24
DO ,1 SUB #12 <- #16
DO ,1 SUB #13 <- #214
PLEASE READ OUT ,1
PLEASE GIVE UP
```

Source: Wikipedia

Brainfuck

- Created in 1993 by Urban Müller, as a language that could be implemented with the smallest possible compiler.
- Several brainfuck compilers less than 200 bytes exist.
- Has only eight commands, the following:

Character	*Meaning*
>	increment the pointer (to point to the next cell to the right)
<	decrement the pointer (to point to the next cell to the left)
+	increment (increase by one) the byte at the pointer
−	decrement (decrease by one) the byte at the pointer
.	output the value of the byte at the pointer
,	accept one byte of input, storing its value in the byte at the pointer
[	jump forward to the command after the corresponding] if the byte at the pointer is zero
]	jump back to the command after the corresponding [if the byte at the pointer is nonzero

Source: Wikipedia

“Hello, World!” in brainfuck

```
++++++++++
[                    vector initialization
   >+++++++>++++++++++>+++>+<<<<-
]
>++.                 print   'H'
>+.                  print   'e'
+++++++.                     'l'
.                            'l'
+++.                         'o'
>++.                         space
<<+++++++++++++++.           'W'
>.                           'o'
+++.                         'r'
------.                      'l'
--------.                    'd'
>+.                          '!'
>.                           new line
```

Source: Wikipedia

“Hello, World!” in brainfuck, revised

```
++++++++++[>+++++++>++++++++++>+++>+<<<<-]>++.>+.+++++++..+++.>++.<<+++++++++++++++.>.+++.------.--------.>+.>.
```

Source: Wikipedia

The Shakespeare Programming Language

- Designed by Jon Åslund and Karl Hasselström.
- Intent is to make programs look like Shakespeare plays.
- Each variable name must be the name of a character from a Shakespeare play.
- Code is broken in Acts and Scenes
- Reserved words are adjectives and nouns, positive, negative, and neutral.

“Hello, World!” in Shakespeare

```
The Infamous Hello World Program.

Romeo, a young man with a remarkable patience.
Juliet, a likewise young woman of remarkable grace.
Ophelia, a remarkable woman much in dispute with Hamlet.
Hamlet, the flatterer of Andersen Insulting A/S.


                    Act I: Hamlet's insults and flattery.

                    Scene I: The insulting of Romeo.

[Enter Hamlet and Romeo]

Hamlet:
 You lying stupid fatherless big smelly half-witted coward!
 You are as stupid as the difference between a handsome rich brave
 hero and thyself! Speak your mind!

 You are as brave as the sum of your fat little stuffed misused dusty
 old rotten codpiece and a beautiful fair warm peaceful sunny summer's
 day. You are as healthy as the difference between the sum of the
 sweetest reddest rose and my father and yourself! Speak your mind!

 You are as cowardly as the sum of yourself and the difference
 between a big mighty proud kingdom and a horse. Speak your mind.

 Speak your mind!

[Exit Romeo]

                    Scene II: The praising of Juliet.

[Enter Juliet]

Hamlet:
 Thou art as sweet as the sum of the sum of Romeo and his horse and his
 black cat! Speak thy mind!

[Exit Juliet]

                    Scene III: The praising of Ophelia.

[Enter Ophelia]

Hamlet:
 Thou art as lovely as the product of a large rural town and my amazing
 bottomless embroidered purse. Speak thy mind!

 Thou art as loving as the product of the bluest clearest sweetest sky
 and the sum of a squirrel and a white horse. Thou art as beautiful as
 the difference between Juliet and thyself. Speak thy mind!

[Exeunt Ophelia and Hamlet]


                    Act II: Behind Hamlet's back.

                    Scene I: Romeo and Juliet's conversation.

[Enter Romeo and Juliet]

Romeo:
 Speak your mind. You are as worried as the sum of yourself and the
 difference between my small smooth hamster and my nose. Speak your
 mind!

Juliet:
 Speak YOUR mind! You are as bad as Hamlet! You are as small as the
 difference between the square of the difference between my little pony
 and your big hairy hound and the cube of your sorry little
 codpiece. Speak your mind!

[Exit Romeo]

                    Scene II: Juliet and Ophelia's conversation.

[Enter Ophelia]

Juliet:
 Thou art as good as the quotient between Romeo and the sum of a small
 furry animal and a leech. Speak your mind!

Ophelia:
 Thou art as disgusting as the quotient between Romeo and twice the
 difference between a mistletoe and an oozing infected blister! Speak
 your mind!

[Exeunt]
```

Further Reading

- Steve McConnell, Code Complete, 2^{nd} edition.
- Joshua Bloch and Neal Gafter, Java Puzzler: Traps, Pitfalls, and Corner Cases.
- Brian W. Kernighan and J. P. Plauger: The Elements of Programming Style, 2^{nd} edition.
- Mateas, Michael; Nick Montfort. "A Box, Darkly: Obfuscation, Weird Languages, and Code Aesthetics". Proceedings of the 6th Digital Arts and Culture Conference, IT University of Copenhagen, 1-3 Dec 2005: 144-153