CS1110 Introduction to Systematic Programming

First Practical Class in Week 7 -- Arrays

Copy the files

```
/usr/local/staffstore/cs1110/unit-programs/opinion11.adb
/usr/local/staffstore/cs1110/unit-programs/opinion.dat
```

to you own UNIX area. These are a copy of the example program from Unit 11 on arrays. and a suitable data file for use with the program.

Open the Ada file with Emacs and study it -- trying to understand what it does. Compile the file for debugging by using the **Build** command from the Emacs **Ad**a menu (or from the UNIX command-line using `gnatmake` with the `-g` option. Then start the debugger GDBTK from a terminal window using the command

```
gdbtk opinion11
```

and start the program. Display the contents of the variable `ResponseCode` and of the array `Count` in the Expressions window by double-clicking on these variables. Then single-step the program (using the **Step** button). You will need to type the name of the data file (`opinion.dat`) in the terminal window in response to the `OpenInput` command. Now single-step through the first `FOR` loop several times observing how the contents of these variables change as program execution proceeds. Before doing this it may help to open the data file in Emacs so that you can see the data that is being input by the program.

When you are confident that you understand how the program works set a breakpoint on the line immediately after the end of the first `FOR` loop, i.e. the one beginning

```
Put(Item => "Category ........
```

(do this by clicking on the number to the left of this line whereupon a red cross should appear). Now run the program at full speed until the breakpoint is reached by clicking once on **Cont** button. Observe the contents of the array and then clear the breakpoint (by clicking on the red cross) and then single-step the program to its conclusion watching in the terminal window to see the program output appearing.

Note when the main Ada program is complete, the source code for the bind file will be displayed. There is no need, of course, to step through this (unless you are curious) so you can run the program through the bind code at full speed by clicking the **Cont** button.

Exercise
The algorithm below is an alternative solution to the second problem on Problem Sheet 6.
It uses the following two arrays:

```
TYPE BallList IS ARRAY(1 .. 49) OF Integer;
TYPE SelectedList IS ARRAY(1 .. 6) OF Integer;
Ball : BallList;
Drawn : SelectedList;
```

Algorithm:

```
Initialise the elements of the Ball array to 1 through to 49
FOR Count IN 1 .. 6 LOOP
   Generate a random number N in the range 1 .. (50 - Count)

   Add Nth number in the Ball array to the Drawn list

   Remove Nth element from the Ball array by moving all elements
     with indices I where I > N "one place to the left".  Thus
     the element at position I should be moved to position I - 1.

END LOOP;

Output the Drawn list
```

PTO ...

Convert the algorithm into a complete Ada program, enter it into the computer and compile, debug and run it.

What are the advantages/disadvantages of this method compared to the one proposed on the problem sheet? How would your conclusion be affected if substantially more balls (say 20 or even all 49) were to be drawn.

If you don't have a hard copy of the problem sheet with you then download a copy from the CS1110 module web site.

Second Coursework
Please note that the second coursework problem will be distributed later this week. It depends on material covered in Units 1–14. The submission dead-line is Friday 5th December.