Lecture C3: Ada: Files

Response to 'Muddiest Part of the Lecture Cards'

(15 respondents)

1) *Are there standard conventions for the naming of files*?
Yes there are some standard conventions. A file name is essential for both the OS and the user. Meaningful/friendly names of files make it easier for the user or program developer to manage and process large numbers of files quickly.
The filename is composed of two parts, the name and its extension. The extension tells us what kind of file it is. In general we have two different kinds of files, executable (binary) files and ASCII files such as .txt.
Filenames are usually limited in size and on what characters can exist in the file name. DOS has some of the most restrictive limitations for filenames; the rules have been relaxed for the 95 and NT generations of operating systems. A Windows filename can be up to 253 characters long (11 for DOS) and a filename character can be any one of 245 ASCII codes (52 (subset of the 245 ASCII codes) for DOS). UNIX file names can be upto 255 characters (with the exception of '/' and '\0').

The Ada Style Guide says the following: http://www.iste.uni-stuttgart.de/ps/ada-doc/style_guide/sec_3a.html#3.2

2) *What kind of files besides .txt can our compiler handle*?
Our compiler can only handle .ada .adb and .ads files. It is possible to generate files of any name / contents. You can generate file names via the create command to be what you want them to be. If you want to generate output for e.g., a Microsoft Word document (file.doc) you can use:

```
Ada.Text_Io.Create (File_Word,Ada. Text_Io.Out_File, "my_file.doc");
```

This will only create a file of the given name and extension. The format however is controlled by the user (also see 3 and 4).

3) *Are these files the same sort of files that we wrote in Microsoft Notepad and used in the beam lab or do you have to create all your files in Ada*?
No you don't have to create all the files in Ada. For the beam lab, you created files using notepad and knew the format with which the data was stored. This allowed you to read the data in the files and write into the files with the required format.

4) *Can you save text to any program (i.e., word or excel)*?
There are two interpretations to this question:

a. Can I directly send the contents of the file to a word or excel program that is already running? Yes, provided you understand the communication mechanisms

between the different programs. Cut and Paste operations work across Word, Excel and Powerpoint through a mechanism called OLE (object linking and editing). You can use a similar mechanism for your Ada program to talk to another program.

b. Can I create a file that can be read by another program? Yes, if you know the details of naming convention for the program (e.g., .doc .xls) and all the details of how data is represented by the other programs, you should be able to create files that can be read and manipulated by them.

5) *When/how do you specify what folder you want an output file to be created in*?
If you take a look in the file multfileout.adb that was used in class today (program 3) Instead of creating all three files in the same folder as the multfileout.adb file did, I could have done the following:

```
   Create(Numbers,Out_File, "C:\Documents and Settings\Kristina\My
Documents\Undervisning\Unified 03-04\Spring 2004\Ada\3\INTDATA.TXT");
   Create(Some_Text, Out_File, "TEST.TXT");
   Create(More_Text, Out_File, "CHARACTS.TXT");
```

And the file INTDATA.TXT would have been created in the folder "3" on my computer, and the other two files are created in "the current folder".

6) *Is* standard_Output *always to the monitor (assuming we do not change defaults)*?
Calls using Standard_Output as the file are routed to the device that the operating system considers to be standard_output. In the case of most computing systems which have some form of human interaction, the standard output is the monitor (terminal in unix jargon). Often times for embedded systems, it is a log file which can be analyzed during maintenance.

7) *Do we need two separate commands to output the same data to standard_output and to file*?
Yes. As far as Ada is concerned, standard_output is also an external file. Since you want to write to two files, you will need two separate subprogram calls.

8) *If every input or output of your program is a file, including user input, is there a way to preserve this "file" without explicitly storing each line of input*?
Interesting question. It is possible to create an area in memory where every interaction of your program is stored, which can later be analyzed, converted into a file, or simply ignored. As the number of interactions between the program and the external environment grows, the amount of memory needed to log all the operations increases, which then becomes a bottleneck on the systems performance.

10) *Could you explain:*

```
type My_Fixed is delta 0.01 range 20.0 .. 42.0;
```

just a type of numbers 20.0, 20.01, 20.02, ?

Correct. The type definition says that the smallest number in the type is 20.0, the largest is 42.0 and the granularity is 0.01 i.e. the numbers 20.011 and 20.009 are indistinguishable.

11) *Could you explain the 24.3283 / delta .01 thing? Why does it show those numbers*? (3 students)

The 24.3283 appears because of the `Put(Size,5,5),` which specifies that the data has to be displayed with atmost 5 digits of decimal precision. 24.3283, is the greatest representation precision that you can get for 24.33. Try changing the value in the program to 24.328 and see what the output of the program is.

12) *Is the primary function of a file simply to store and provide data for a program*?
Yes and No. A file can be used as the source/sink of data. It can also be used to share data across multiple programs and as a communication medium. For instance, your code files are looked at by multiple people and act as a means of communicating your thought process/ problem solving approach to other people.

13) *This is the 3rd lecture and I am already completely lost.*
Some things one can try: read through the reading instructions before attending class, work your way through examples from class and in Feldman (or online tutorials), set up tutoring with the TAs, or book a time with me.

14) *No mud (very good lecture)*. ☺ (3 students)