Programming Languages Recitation

Ada - procedures, packages and tasks

Arpit Jain

Computer Science Department
Courant Institute of Mathematical Sciences
New York University

arpit.jain@cs.nyu.edu

September 11, 2014

Overview

1. What is Ada and Motivation
2. Basics
3. Subprograms
   - Procedures
   - Functions
4. Packages
5. Tasks

What is Ada

Features

- Structured
- Statically Typed
- Object-oriented high level programming language

Users/Uses

- U.S. Department of Defense (DoD) for real-time embedded systems
- Large-scale information systems
- Distributed systems
- Scientific computation
- Safety-critical systems

Ada and NYU Courant

- Jack Schwartz (founder of CS dept.) developed SETL (High level based on mathematical theory of sets). SETL was then used for the first valid implementation of Ada called NYU Ada/ED

Ada and NYU Courant

- Jack Schwartz (founder of CS dept.) developed SETL (High level based on mathematical theory of sets). SETL was then used for the first valid implementation of Ada called NYU Ada/ED
- Adacore Executive Team
    - Robert Dewar (President Adacore, Emeritus Professor CS dept. NYU)
    - Edmond Schonberg (Professor CS dept. NYU)
    - Richard Kenner (Researcher CS dept. NYU)
    - Franco Gasperoni (PhD from CS dept. NYU)

Compilation Units

An Ada program is composed of one or more units of

- Subprograms - Procedures or Functions
- Packages

Additionally there are other units like

- Task
- Protected
- Generic

Procedures

- A procedure call is a statement and does not return any value
- Subprogram parameters modes:
  - ‘in’ - value may be used but not changed (Default mode)
  - ‘out’ - value may be changed but not used
  - ‘in out’ - value may be changed and/or used
  - ‘access’
- procedure Average(A, B : in Integer; Result : out Integer);

```
with Ada.Text_IO;
procedure Hello is
begin
 Ada.Text_IO.Put_Line("Hello,␣world!");
end Hello;
```

Functions

- Unlike procedure a Function returns value
- Parameters mode remain same as procedures
- function Average_Two(A, B : in Integer) return Integer;

```
function Sum(A, B : in Integer) return Integer is
 Total : Integer := A;
begin
 Total := Total + B;
 return Total;
end Sum;
```

Packages

Each Program unit consists of two parts :

- Declaration/Specification - interface of the unit. Analogous to '.h' file in C
- Body - Implementation details of the unit. Analogous to '.c' file in C

What is a Package :
Collection of logically related entities

Package Declaration/Specification

- Package Specification go in the .ads file

```
package stack is
 procedure push(x:integer);
 function pop return integer;
end stack;
```

- Package body go in .abd file

```
package body stack is
 procedure push(x:integer) is
  begin
    -- Do something here
  end;
 function pop return integer is
 begin
   -- Do something here
 end;
end stack;
```

Tasks

- An independent execution of the same static code, having a stack, program counter and local environment but shared memory
- Ada task communicated through
  - Rendezvous - message passing
  - Shared Variables
  - Protected objects

```
with text_io;
use text_io;

procedure hello is
   task Foo;
   task body Foo is
   begin
      Put(”In␣foo”);
      New_Line;
   end Foo;
begin
  Put(”Hello␣World”); New_Line;
end Hello;
```