16.070

Introduction to Computers & Programming

Ada III

Introduction to scalar types, subtypes, enumeration types

Guest Lecturer: Tony Bogner
Charles Stark Draper Laboratory

Scalar Types

- A scalar is a single unit of data. Scalar data types are single-valued data types, that can be used for individual variables, constants, etc.
- Recall the elementary types that we saw earlier:
  - INTEGER
  - FLOAT
  - CHARACTER
  - BOOLEAN

Scalar Types

- Ada programmers are not limited to the above built-in types. You can **define your own types**. They can be **structured types**, which combine multiple data items into a larger unit, or they can be scalar (single-valued) types.
- Scalar types can be:
  - a new type **derived** from an existing type
  - an **enumeration type**
  - a **subtype** (subrange of values) of an existing type
- Each distinct type needs its own I/O library

Derived Types

- Motivation
  - Using pure numbers, we can write nonsense:
    - age := -20;
    - height := age - class_size;
    - shoe_size := 2 * no_on_bus;
  - Types help program values reflect the real world.

Integer Types

- New data types can be derived from `INTEGER`:

```
TYPE ages IS NEW Integer RANGE 0 .. 110;
age : ages;
voting_age : CONSTANT ages := 18;

TYPE heights IS RANGE 0 .. 230;
height : heights;

max_channels : CONSTANT := 12;
TYPE rcvr_channel_t IS RANGE 1.. max_channels;

current_channel : rcvr_channel_t;
```

- These types are distinct from each other, and from `INTEGER`.
- Types cannot be mixed in Ada, so nonsense can be avoided.

Floating Point Types

- New data types can be derived from `FLOAT`:

```
TYPE measurement_t IS NEW Float;
pseudorange_measurement : measurement_t;

TYPE pressure_t IS DIGITS 7;
atmospheric_pressure : CONSTANT pressure_t := ;

max_voltage : CONSTANT := 10.0;
TYPE voltage_t IS DIGITS 5 RANGE 0.0.. max_voltage;

voltage_reading : voltage_t;
```

- These types are distinct from each other, and from `Float`.
- Types cannot be mixed in Ada, so nonsense can be avoided.

Type conversion

- Ada has *strong typing:* different types cannot be mixed
- Explicit type conversion is permitted:

```
type length is digits 5 range 0.0 .. 1.0E10;
type area is digits 5 range 0.0 .. 1.0E20;

function area_rectangle (L,H : length) return area is
begin
   return area(L) * area(H);
end;
```


Benefits of derived types

- Nonsense rejected by compiler
  `height := age - current_channel;`
- "Out of range" rejected by compiler
  `age := -20;`
- “Out of range” run time error
  `current_channel := current_channel + 12;`
- Enforce distinct nature of different objects
- Robust, elegant, effective programs

Subtypes

- A subtype is a subrange of a larger type. Subtypes are appropriate whenever there are ranges of allowed values.
- Subtypes of the same larger type are not distinct types. A subtype and the larger type are also not distinct types. Thus subtypes of the same thing are assignment-compatible.
- The benefit of subtypes is that range checks avoid some nonsense.

Subtype Example

- Two useful sub-types of the integers are built into Ada:
  - `subtype POSITIVE is INTEGER range 1..INTEGER'LAST;`
    `subtype NATURAL is INTEGER range 0..INTEGER'LAST;`
- A distinct type could be defined for the number of people on a bus (making it a distinct type prevents other integers, like shoe sizes, from being added to it). Within that type, different limits apply to the numbers of people that are allowed to be sitting or standing on the bus. Subtypes are appropriate whenever there are ranges of allowed values.

```
min_on_bus : constant := 0;
max_on_bus : constant := 80;
type no_on_buses is range min_on_bus .. max_on_bus;

max_seated : constant no_on_buses := 50;

subtype seated_on_buses is no_on_buses
   range min_on_bus .. max_seated;
subtype standing_on_buses is
   range min_on_bus .. (max_on_bus - max_seated);
```


Subtypes

```
subtype Natural is Integer range 0..Integer'Last;

subtype Positive is Integer range 1..Integer'Last;

subtype NonNegativeFloat is Float range 0.0 .. Float'Last;


subtype SmallInt is Integer range -50..50;
subtype CapitalLetter is Character range 'A'..'Z';
X, Y, Z      : SmallInt;
NextChar     : CapitalLetter;
Hours_Worked : NonNegFloat;


X := 25;
Y := 26;
Z := X + Y;
```


Enumeration Types

- A data type whose values are a collection of allowed words

```
type Class is
  (Freshman, Sophomore, Junior, Senior);

type days is (Mon, Tue, Wed, Thu, Fri, Sat, Sun);
type colours is (white, red, yellow, green, blue, pink, black);
type traffic_colours is (green, yellow, red);
type suits is (clubs, diamonds, hearts, spades);
```

The values in an enumeration type are ordered:

these conditions are all **true**

```
clubs < diamonds
white < pink
colours(green) > colours(red)
traffic_colours'(green) < traffic_colours'(red)
```

Enumeration Types

- Note that the same words can be elements in different enumeration types, even in the same program. They are distinguished by reference to the type as well as the element. The last examples show how this is done.
- Enumeration types have the following benefits:
  - readable programs
  - avoid arbitrary mapping to numbers (e.g. better to use "Wed" than 3 for a day of the week)
  - they work well as selectors in case statements

Enumeration Type Attributes and Operations

```
type Days is
  (Monday, Tuesday, Wednesday, Thursday, Friday,
   Saturday, Sunday);

Today    : Days; --current day of the week
Tomorrow : Days; --day after Today

Today := Friday;
Tomorrow := Saturday;

Days'First           is Monday
Days'Last            is Sunday
Days'Pos(Monday)     is 0
Days'Val(0)          is Monday
Days'Pred(Wednesday) is Tuesday
Days'Pred(Today)     is Thursday
Days'Succ(Tuesday)   is Wednesday
Days'Succ(Today)     is Saturday
```

You must ensure the result is legal. A CONSTRAINT_ERROR will occur at run-time otherwise. For example, `Days'SUCC(Sun)` is illegal.

I/O Libraries

- Each distinct type needs its own I/O library.
- General form:
  - **package** type_io **is new** TEXT_IO.basetype_io (typename);
- Subtypes don't need separate I/O libraries, but the I/O libraries for the base types must be present.

```
package int_io is new TEXT_IO.INTEGER_IO (INTEGER);

type ages is new INTEGER range 0 .. 110;
package ages_io is new TEXT_IO.INTEGER_IO (ages);

type measurement is digits 10;
package measurement_io is new TEXT_IO.FLOAT_IO (measurement);

type suits is (clubs, diamonds, hearts, spades);
package suits_io is new TEXT_IO.ENUMERATION_IO (suits);

type colours is (white, red, yellow, green, brown, blue, pink, black);
package colours_io is new TEXT_IO.ENUMERATION_IO (colours);
```


Input/Output Operations

```
type Days is
  (Monday, Tuesday, Wednesday, Thursday, Friday,
   Saturday, Sunday);

package Day_IO is new Ada.Text_IO.Enumeration_IO(Enum=>Days);

if this_day in weekend_days then
   put("Holliday!");
end if;

Day_IO.Get(Item => Today);
Day_IO.Put(Item => Today, Width => 10);
```

What colours are produced when primary colours are mixed?

```
with TEXT_IO; use TEXT_IO;

procedure mix_colours is

   type colour is (red, yellow, blue, green, orange, purple);

   subtype primary_colour is colour range red .. blue;

   package colour_io is new enumeration_io (colour);
   use colour_io;

   colour1, colour2 : primary_colour; -- input colours
   colour_mix : colour;               -- result of mixture
```


What colours are produced when primary colours are mixed?

```
begin -- mix_colours

  PUT ("Enter two of the primary colours ");
  PUT_LINE ("(RED, YELLOW, BLUE)");
  GET (colour1);
  GET (colour2);
  if     ((colour1 = red) and (colour2 = yellow)) or
         ((colour2 = red) and (colour1 = yellow)) then
     colour_mix := orange;
  elsif  ((colour1 = red) and (colour2 = blue)) or
         ((colour2 = red) and (colour1 = blue)) then
     colour_mix := purple;
  elsif  ((colour1 = blue) and (colour2 = yellow)) or
         ((colour2 = blue) and (colour1 = yellow)) then
     colour_mix := green;
  else -- same colours
     colour_mix := colour1;
  end if;

  PUT ("The colour mixture will be ");
  PUT (colour_mix);
  NEW_LINE;
end mix_colours;
```

Sample Run

```
Enter two of the primary colours (RED, YELLOW, BLUE)
red
blue
The colour mixture will be PURPLE
```