Movie Recommendation System

—

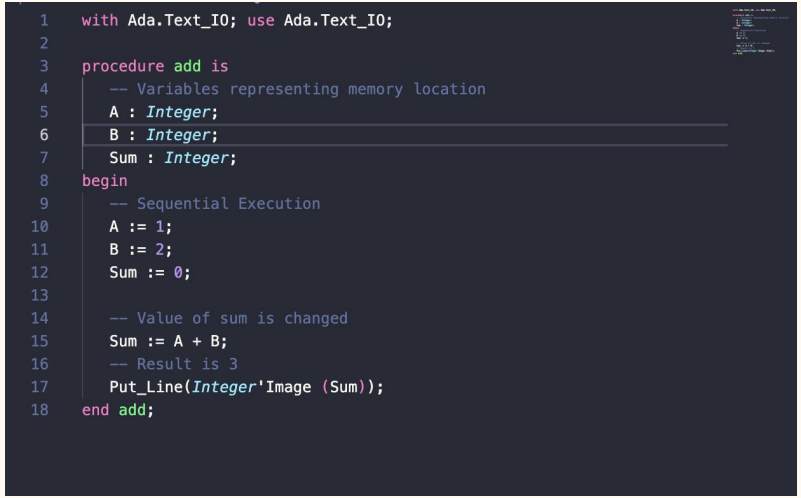
The Evil Residents

What is the language paradigm?

Ada is a multi-paradigm language but it is primarily an Imperative Language.

Imperative language: a language with three properties

- Sequential execution of instructions
- Use of variables representing memory locations
- Use of assignment to change the values of variables



```ada
with Ada.Text_IO; use Ada.Text_IO;

procedure add is
   -- Variables representing memory location
   A : Integer;
   B : Integer;
   Sum : Integer;
begin
   -- Sequential Execution
   A := 1;
   B := 2;
   Sum := 0;

   -- Value of sum is changed
   Sum := A + B;
   -- Result is 3
   Put_Line(Integer'Image (Sum));
end add;
```

How does Ada fit into the historical evolution of programming languages?

- The concept of Ada was introduced in the late 1970s to address the rise of programming languages used within the U.S Department of Defense.
- In 1983, Ada was published and became one of the first widely used language on the PC platform
- Today Ada is used primarily within critical softwares and embedded systems

https://www.adacore.com/about-ada/timeline-of-ada

What did you find different about the language from other languages in the same paradigm?

Other languages (Java, C++)

- Syntax
- Errors

Ada

- Uses an expository syntax to make reading code easier.
  - Instead of using curly braces like other languages begin and end blocks are used.
- Designed to catch errors at compilation time not runtime.
  - Helps eliminate bugs and errors early on.

What features did you find helpful?

Encapsulation and Modularity

- Nested procedures keep related logic grouped together, functions within our code including; Load_Movies, Display_Movie, and all the filters reduce global namespace pollution

```
procedure Display_Movie(Movie : Movie_Record) is
begin
   Ada.Text_IO.Put_Line("Title: " & ASU.To_String(Movie.Title));
   Ada.Text_IO.Put_Line("Genre: " & ASU.To_String(Movie.Genre));
   Ada.Text_IO.Put_Line("Duration: " & Integer'Image(Movie.Duration) & " minutes");
   Ada.Text_IO.Put_Line("Service: " & ASU.To_String(Movie.Service));
   Ada.Text_IO.Put_Line("Rating: " & Integer'Image(Movie.Rating));
   Ada.Text_IO.New_Line;
end Display_Movie;
```

Dynamic and Flexible String Handling

- Unbounded Strings avoid fixed-length buffers, ultimately, eliminating an entire class of buffer-overflow bugs

What does this language offer to the programmer that makes it a language you would want to use?

- Strong Typing:

  Mixing up strings/integers, or passing the wrong data type is caught at compile time before the code ever runs

- Generics:

  Very simple generic packages, making it easy to write and instantiate a fully tested data-structure for any element without any runtime overhead.

- Readability:

  Due to Ada's explicit syntax, the control flow is unambiguous and easy to navigate.

Is the language best suited for one particular type of application? If so, what?

Avionics & Air Traffic Control:

- The FAA safety and mobility systems are enhanced due to Ada's high availability and domain reliability
- The FAA's core En Route air traffic control systems were developed at an unprecedented short-time and within the budget all at the mercy of Ada's versatility

Military & Defense Systems:

- In the mid 1970's, the U.S. DoD and UK Ministry of Defense replaced hundreds of their specialized programming languages within their embedded systems solely for Ada
- Ada is integrated into essential parts of military projects including; missile systems (SM2 and SM3), naval ship control systems (CG 47), and engine control systems (DDG-79)

Demonstration

Thank You!