PL Recitation 14

Ang Sun
asun@cs.nyu.edu

Topics

- HW 13
  - Exception handling
- Scoping review
- Concurrency in Ada
- List manipulation in Scheme and ML
  - length
  - reverse
Write a program in your favorite language in which f calls g calls h, an exception is raised in h, and the exception is handled in f.

Ada example:

- f.adb
- f2.adb

```ada
with Text_IO; use Text_IO;

procedure f is
  Overflow : exception;

  procedure h is
    begin
      raise Overflow;
    end;

  procedure g is
    begin
      h;
    end;

  begin
    g;
    exception when Overflow ->
      Put_Line("Wrong!!! overflow");
  end;
```
Scoping review

- What's wrong with f2.adb?
  - Ada use static scoping; exception is also statically scoped. However, exception propagation is via the dynamic call chain.

- Static vs Dynamic scoping
  - The wiki example

```ada
with Text_IO; use Text_IO;

procedure f2 is
  procedure h is
    Overflow : exception;
    begin
      raise Overflow;
    end;

  procedure g is
    Overflow : exception;
    begin
      h;
      exception when Overflow =>
        Put_Line("Wrong!!!! overflow here..."个百分比);
    end;

    begin
      g;
    end;
```

```plaintext
int x = 0;
int f() { return x; }
int g() { int x = 1; return f(); }
```
Most languages we use are statically scoped

static scoping - the body of a procedure is executed in the environment in which the procedure was defined.

dynamic scoping - the body of a procedure is executed in the environment in which the procedure was called.

So in the wiki example

```c
int x = 0;
int f() { return x; }
int g() { int x = 1; return f(); }
```

- With static scoping, calling `g` will return 0 since it has been determined at compile time that the expression `x` in any invocation of `f` will yield the global `x` binding which is unaffected by the introduction of a local variable of the same name in `g`.
- With dynamic scoping, the binding stack for the `x` identifier will contain two items when `f` is invoked from `g`: the global binding to 0, and the binding to 1 introduced in `g` (which is still present on the stack since the control flow hasn't left `g` yet). Since evaluating the identifier expression by definition always yields the top binding, the result is 1.

---

Concurrent in Ada

- Question: In Ada, write a procedure that declares two tasks, task one and task two. When the procedure is called, task one and task two should print “one” and “two”, respectively, over and over such that the printing of “one” and “two” is perfectly interleaved. Thus, the output should look like:
  - one
  - two
  - one
  - two
  - ...

- How to solve this problem?
  - Mutual exclusion
  - Conditional communication
  - ...
1. (2 Points) Write the (recursive) definition of length in Scheme. Keep your answer to 5 lines of code or less.

   **Answer:**
   
   ```scheme
   (define (length lst)
     (if (null? lst)
         0
         (+ 1 (length (cdr lst))))
   )
   ```

2. (2 Points) Write the (recursive) definition of length in ML. Keep your answer to 5 lines of code or less.

   **Answer:**
   
   ```ml
   fun length nil = 0
     | length (l::t) = 1 + length t;
   ```

(2 Points) What is the static Scheme type of `length` as determined by a Scheme compiler? Keep your answer to one sentence or less.

   **Answer:** There is none; Scheme is dynamically typed.

(2 Points) What is the static ML type of `length` as determined by an ML compiler? Keep your answer to one sentence or less.

   **Answer:** `list -> int`
(2 Points) For each of the following Scheme or ML programs indicate (a) whether the program will compile, (b) if so, what happens when you run it, and (c) why? Keep each answer to one sentence: “The program does x because of y”.

(a) Scheme: (length 1)

**Answer:** The program compiles and terminates with an error because 1 is neither nil nor a pair.

(b) ML: length 1;

**Answer:** The program does not compile because ML’s type checker detects that 1 is not a list.

(c) Scheme: (length '(i "two"))

**Answer:** The program compiles and evaluates to 2 because Scheme lists may contain any (type of) value.

(d) ML: length [1, "two"];

**Answer:** The program does not compile because ML lists must contain elements of a single type but 1 is of type int and "two" is of type string.