CS 342 — Principles of Programming Languages November 3, 1989

Test 2

Please write your name here:

Before you begin, please take a moment to look over the entire test so that you can budget your time.

Answers to essay questions will be judged on their completeness and clarity. However, please do not feel obligated to fill up all the empty space on the page. Conversely, do not feel limited by the space on a page, since you may continue an answer on the back of a page if you wish.

For each of the multiple choice questions, please circle the correct answer. You will lose one point for an incorrect answer on a multiple-choice question, so don't guess randomly.

problem	subject	points	score
1	C calling mechanism	4	
2	CLU calling mechanism	4	
3	static chaining vs. display	6	
4	static chaining vs. display	6	
5	static link	5	
6	draw the stack	15	
7	information hiding	15	
8	closures	7	
9	type checking	10	
10	exception handling	15	
11	procedure parameters	13	
Total		100	

This exam (only) counts for 15% of your final grade.

1. (4 points) The procedure calling mechanism used in C is:
   (a) call by constant,
   (b) call by name,
   (c) call by reference,
   (d) call by value,
   (e) call by value-result.
2. (4 points) The procedure calling mechanism used in CLU is:
   (a) call by constant,
   (b) call by name,
   (c) call by reference,
   (d) call by value,
   (e) call by value-result.
3. (6 points) disregarding optimizations, which variable access technique uses fewer memory references to access a variable at a static distance greater than 3:
   (a) a display,
   (b) static chaining.
4. (6 points) disregarding optimizations, which variable access technique requires fewer memory references to call and return from a local procedure (i.e., at a static distance of 0):
   (a) a display,
   (b) static chaining.
5. (5 points) Which of the following best describes the static link of an activation record for a procedure `P` (for a compiler that uses static chaining):
   (a) `P`'s static link points to the activation record for the most recent activation of `P`'s caller
   (b) `P`'s static link points to the activation record for the most recent activation of the block or procedure that lexically encloses `P`
   (c) `P`'s static link points to the activation record for the most recent activation of the block or procedure that lexically encloses `P`'s caller.

6. (15 points) Using static chaining, draw the run-time stack (growing upwards, please), for the following Pascal program when execution reaches the point marked. Include in your drawing the static links (labeled SL), dynamic links (labeled DL), arrows showing where each link points, variable names (labeled with the variable name), variable values (including parameters), and the EP and SP registers (and arrows for where they point). Use a heavy line between activation records. Be sure to label each activation record with the name of the procedure it is for, and clearly indicate to which activation record each static or dynamic link points. (Other details are not necessary.)

```
program p;

    function r(n: integer): integer;
    begin
        {** draw stack when here **}
        r := n
    end;

    procedure q(m: integer);
        var v: integer;
    begin
        if m = 2
        then q(m-1)
        else v := r(m)
    end;

begin {p}
    q(2)
end.
```

7. (15 points) Describe the significant differences between the facilities provided for information hiding in C and CLU.

8. (7 points) What is a closure? (Give a brief description)

9. (10 points) Describe one method for doing static type checking across module boundaries in a language with separate compilation.

10. (15 points) Describe the significant costs and benefits of a linguistic exception handling mechanism, such as found in CLU.

11. (13 points) Which of the following are significant *benefits* of a language that allows procedures as parameters (such as C or CLU) versus one that does not?

    Circle all of the following that are benefits of having procedure parameters, and give a short explanation (one sentence) of how each is a benefit. (Don't define the terms.)

    (a) abstraction

    (b) automation

    (c) compile-time time efficiency

    (d) expressiveness

    (e) information hiding

    (f) regularity

    (g) run-time time efficiency

    (h) security