Department of Computer Science and Engineering
University of California at San Diego

CSE 230
Winter 1998

Problem Set 5

DUE TUESDAY FEBRUARY 10 IN CLASS.

1. It has been claimed that in Pascal a loop in the main program of the form **while** $\mathcal{E}$ **do** $\mathcal{C}$ is identical in effect to the procedure invocation $whiledo(\mathcal{E},C)$ given the top-level procedure definitions

   procedure C;
   begin
   $\quad\mathcal{C}$
   end;

   procedure $whiledo(E : boolean;$ **procedure** $C)$;
   begin
   $\quad$**if** E **then**
   $\qquad$**begin**
   $\qquad\quad C; whiledo(E, C)$
   $\qquad$**end**
   end;

   Note that in this program text, calligraphic letters such as $\mathcal{E}$ and $\mathcal{C}$ indicate arbitrary well-formed expressions or commands, while italic letters such as E are identifiers.

   Why is this claim incorrect? Suggest a change to the semantics of Pascal that would make the claim correct. How could the program fragment be modified to make the claim correct, mimicking in standard Pascal your proposed change in semantics?

2. Here is a Pascal procedure that is intended to swap the values of its two arguments, which are passed by reference.

   procedure $swap$ (**var** $x, y : integer$);
   begin
   $\qquad x \leftarrow x + y;\;\; y \leftarrow x - y;\;\; x \leftarrow x - y$
   end

   Give examples of two contexts where, for different reasons, this procedure does not work. Explain features of the Ada language which could be used to let a similar Ada procedure be correct in at least one of these contexts. (For information on Ada, see [Kamin, 1990] or [Sethi, 1989].)

3. (a) According to Prashant Jain and Douglas C. Schmidt in their article *Experiences Converting a C++ Communication Software Framework to Java* in the magazine *C++ Report*, January 1997:

> In contrast to other OO languages like C++ and Ada95, Java doesn't support user-specified parameter passing modes. Instead, Java has reference semantics for object parameters, but passes these references by value. For example, in the following code, even after the `someMethod` invocation, `bar` still points to the `String` object associated with the literal string `"original"`.

```
void someMethod (String foo) {
  foo = "changed";
}

void caller () {
  String bar = "original";
  someMethod (bar);
}
```

Invent and explain a systematic way of writing methods like `SomeMethod` differently so that they can update their arguments.

(b) (This problem is due to Paul Phillips.) Another way of describing parameter passing in Java is to say that all objects are passed by reference, but references are passed by value. There is no way to pass references themselves by reference, as in C with `int * a, * b, swap(&a, &b)`.

Consider the Java procedure

```
public static void swap(Object a, Object b) {
    Object temp;

    temp = a;
    a = b;
    b = temp;
}
```

and suppose that `x` and `y` are arbitrary objects. Exactly what will happen when `swap(x,y)` is called?

Is it possible to write a generic `swap` procedure in Java?