Objectives

1. be able to describe how Simula’s classes/objects are in fact procedures/activation records

Outline

1. problem domain: event-based simulation
2. evolution from Algol 60
   (+) class and reference variables, pass-by-reference
   (+) char, text, and input-output features
   (C) default parameter passing mechanism (pass-by-value-result)
3. implementing objects
   (a) class: procedure returning a pointer to its activation record
   (b) object: activation record produced by a call to a class; called an ‘instance’ of that class
   (c) heap allocation of activation records (garbage collection)
4. OO in Simula 67
   (a) dynamic lookup (for virtual procedures)
   (b) abstraction:
      i. no information hiding (public/private) in original version
      ii. but does encapsulate behavior and state together in activation record
   (c) inheritance: prefix classes (= superclass)
   (d) subtyping: values of a class can be assigned to references variables of prefix classes
5. coroutines

Vocabulary

pass-by-reference, instance, class prefixing, prefix class, coroutine
Simula Example

class Prefix (X); Integer X;
    virtual: procedure Banner; label Dest;
begin
    text star;
    text dash;
    star := "*";
    dash := "-";
    Banner;  ! Call a virtual procedure with no match here;
    OutText("Some stuff.");
    OutImage
end++of++Prefix;

Prefix class Stars;
begin
    procedure Banner;
    begin
        Integer i;
        i := 0;
        while i < X do
            begin
                OutText(star);
                i := i + 1
            end;
    OutImage
    end--of--Banner;
end++of++Stars;

Prefix class Dashes(Y); Integer Y;
begin
    procedure Banner;
    begin
        Integer i;
        i := 0;
        while i < X do
            begin
                while j < i + Y do
                    begin
                        OutText(dash);
                        j := j + 1
                    end;
                OutText(" ");
                i := i + Y + 1
            end;
    OutImage
    end--of--Banner;
    OutImage
end++of++Dashes;

! new Prefix would be an error, because there is no Banner in Prefix
ref(Prefix) objref := new Stars;  ! Prefix code calls Banner in Stars
objref.Banner;
! call Banner in Stars directly
new Dashes;  ! Prefix code calls Banner in Dashes

Adapted from Example 18.1 in Rob Pooley's An Introduction to Programming in Simula (http://www.nacs.hw.ac.uk/~rjp/bookhtml/).