Exercise 4.1: Name Binding

Consider the following program:

```
program Namensbindung;
  x, y : integer;

  procedure p1 (y : integer);
  begin
    print (x + y);
  end p1;

  procedure p2 (y : integer);
  x : integer;
  begin
    x := y;
    p1 (y);
  end p2;

begin
  x := 0; y := 2;
  p1 (y); p2 (y);
end.
```

a. What is the output of the program using
   (i) static name binding?
   (ii) dynamic name binding?

b. What is the scope and the lifetime of every variable `x` using
   (i) static name binding?
   (ii) dynamic name binding?

c. What are disadvantages of dynamic name binding compared to static name binding?

Exercise 4.2: Descriptors for Name Binding

For which concepts found in programming languages are runtime descriptors required for name binding? For which concepts can they be avoided?
Exercise 4.3: Accessibility Checks

Consider the following Ada-program:

```ada
procedure A4 is

    type Integer_Access is access all Integer;

    I1, I2 : aliased Integer;
    P      : Integer_Access := I2'Access;

    procedure X (Q : access Integer) is
        V : aliased Integer := 1;
    begin
        P := V'Access;
        P.all := P.all + Q.all;
    end X;

    procedure Y (Q : access Integer) is
    begin
        P := Integer_Access (Q);
    end Y;

    procedure Z is
        I3 : aliased Integer := 3;
    begin
        Y (I3'Access);
    end Z;

begin
    I1 := 1;
    I2 := 2;
    X (I1'Access);
    Z;
    Ada.Integer_Text_IO.Put (P.all);
end A4;
```

Questions:

- Which problem exists in the program?
- What would happen if this program were written in C, compiled and executed?
- How does Ada try to prevent the problem? (Two separate strategies apply to the subprograms X and Y)
- Discuss advantages and disadvantages of the two approaches.
Exercise 4.4: Descriptors for Address Binding

a. Name some situations in which runtime descriptors are required for address binding. Name other relevant contexts in which they are not required.

b. In which contexts are descriptors for Address Binding split into a static and a dynamic part?

c. Do you know any programming languages, which do not require any runtime descriptors for address binding?

Exercise 4.5: Gotos

a. Can goto-statements cause problems for name bindings? for address bindings?

b. Which restrictions apply to goto-statements in common programming languages to avoid these problems?

c. Which properties must be satisfied in a programming language so that these problems do not occur even if goto-statements are unrestricted?