There are 5 problems in this set, each worth as indicated, for a total of 100 points. The harder parts are marked with a single * (average difficulty) or two ** (higher-than-average difficulty). For the easy points, start with the unmarked parts.

**Problem 1** (50 points) Consider the following SML code. The function imp_rev reverses a list imperatively: It uses one reference to scan down the input list and another to accumulate the elements in reverse.

```sml
fun imp_rev lst =
    let val right = ref []
        and left = ref lst
    in while not (null (!left)) do
        ( right := hd (!left) :: !right;
          left := tl (!left));
    !right
    end;
```

1. The implementation of imp_rev uses a while-do, the general form of which in SML is:

```sml
while ⟨E₁⟩ do ⟨E₂⟩
```

where ⟨E₁⟩ is a valid SML expression of type bool and ⟨E₂⟩ is a valid SML expression of type unit. Show that a while-do expression, as just described, can be de-sugared into an equivalent SML expression not containing any while-do. (No actual SML coding is necessary here, give your answer in pseudo-code with a precise English explanation – no more than 2-3 lines.)

2. Based on your answer in part 1, write the SML code of imp_rev again, where the while-do expression is now de-sugared.

** 3. Consider the code of the SML library function rev. (It is the same code as that of function reverse in Handout 22.) Give a precise reason for why imp_rev is more efficient than rev. Your answer need not take more than 3-5 lines of type-written text. Hint: Execute by hand each of "rev [5,6,7]" and "imp_rev [5,6,7]", and compare the two executions.

* 4. Write the SML code of the function imp_filter, with polymorphic type

```sml
(‘a -> bool) -> ‘a list -> ‘a list
```

by making as few changes to imp_rev as possible. The function imp_filter on input arguments pred and lst, each of the appropriate type, filters out all elements in lst not satisfying the predicate pred, i.e., imp_filter is the imperative version of the applicative (or functional) filter.
For the next 4 problems, you need to read carefully Section 8.5 in [P]; pp 331-334. Your answers will be relatively short.

Problem 2 (15 points) Exercise 8.14, page 334 in [P].

Problem 3 (10 points) Exercise 8.15, page 334 in [P].

* Problem 4 (10 points) Exercise 8.16, page 334 in [P].

* Problem 5 (15 points) Exercise 8.17, page 334 in [P].