Homework 6

Homework Due: Wednesday, March 10 at 5 pm.

The second part of the course is to prove the correctness of OCaml programs by induction. The focus of Homework 6 will be on OCaml lists.

Prepare Your Homework 6 Directory

```
cd ~/csci2041
./setup hw6
cd hw6
```

Relevant OCaml code

```
let rec append l1 l2 =
  match l1 with
  | [] -> l2
  | h1 :: t1 -> h1 :: append t1 l2

let rec length l =
  match l with
  | [] -> 0
  | h :: t -> 1 + length t

let rec sum l =
  match l with
  | [] -> 0
  | h :: t -> h + sum t

let rec filter p l =
  match l with
  | [] -> []
  | h :: t -> if p h then h :: filter p t else filter p t
```

Problem 1 (30 points)

Prove the following theorem by induction. Use the lecture notes as a style guide.

Theorem 1. *For any type* `a` *and any value* `l` *of type* `a list`*, we have*

`append l []` = `l`

Problem 2 (30 points)

Prove the following theorem by induction. Use the lecture notes as a style guide.

Theorem 2. *For any two values* `l1` *and* `l2` *of type* `int list`*, we have*

`sum (append l1 l2)` = `sum l1 + sum l2`

Bonus Problem (10 points)

Prove the following theorem by induction. Use the lecture notes as a style guide.

Theorem 3. *For any type* `a` *and any value* `p` *of type* `a -> bool`*, any value* `l` *of type* `a list`*, we have*

`filter p (filter p l)` = `filter p l`

Submission

You need to typeset or scan your homework as one single file `hw6.pdf` within the directory `hw6`, and submit it to GitHub as you did in earlier homework assignments. It is encouraged to use professional typesetting software such as LaTeX and you can reuse the code in `hw6-handout.tex` for your work. Otherwise, your hand-writing must be clear and legible, and it must be scanned and consolidated as one single file `hw6.pdf`. Submissions consisting of multiple image files are not acceptable.

If you plan to hand-write your solutions, please practice scanning your documents *now,* and reserve enough time before the homework deadline for scanning your work. Claiming that you have done the homework before the deadline but somehow cannot scan or typeset your solution will not grant you a special deadline extension. You need to use late days to cover such mistakes.

Grading

Generic rubric for all mathematical proofs

Principles

1. Mathematical correctness is the most important. The format is less an issue.

   We give points even if you are not following the template with "Domain", "Property", and "Inductive order", though it is difficult to achieve mathematical correctness without specifying these components.

2. Any reasoning solely based on intuition (*e.g.,* "the `append` function will concatenate two lists" without proving it) will be completely ignored. The grading is done as if intuitive reasoning was not written. It is still important to have correct intuition to construct a rigorous proof, but the intuition itself is not a proof.

Components of induction and case analysis

1. It is important to say you are working on *values.* (Otherwise, your induction probably would not work.)
2. It is important to write out the property and inductive order correctly. (Otherwise, it is unclear what kinds of induction you are doing.) For case analysis, you still need to write out the property.
3. It is important to explicitly mark or declare the usage of the inductive hypothesis. (Otherwise, the reasoning is considered unclear.)

Three levels of correctness

Correct: Mathematically correct with reasonable steps (using the notes as the guideline), and with correct usage of the lemmas or inductive hypothesis. The following mistakes are ignored:

- Minor mistakes that can be fixed by adjusting parentheses, such as
  - `append (l1, l2)` or `append (l1 l2)` instead of `append l1 l2`

Requiring Changes: Making non-trivial mistakes or missing important steps, but with a correct structure with enough details. It must be similar to a correct answer, in the sense that a TA must be able to reconstruct a correct answer from what was written without significant global changes. The following mistakes belong to this category:

- Using single elements as singleton lists.
  - `append l x` instead of `append l [x]`
- Using `::` for list concatenation.
  - `l1 :: l2` for two lists `l1` and `l2` of the same type.
- Directly state (without a proof) that a `list` value must be of the form `[x1; x2; ...; x3]`.

Wrong: Having severe errors so that a TA could not fix the answer with only local changes.

How to mark the inductive hypothesis

(Either works!)

- Mark its usage right at the spot where it is used. For example, "by the inductivie hypothesis on `t`".
- State clearly the implication of the inductive hypothesis for a particular case as part of the regular mathematical reasoning. For example, it is okay to state the consequence of the inductive hypothesis on the tail of a list in the beginning of the case such as "the inductive hypothesis implies that $P($`t`$)$ holds."

Problems 1 and 2

If the proof does not follow the structure at all, then 30/30 if mathematically correct, 15/30 if "Require Changes", and 0/30 pts if Wrong.

- Domain:
  - Valueness: 2 points
  - Type: 2 points
- Property and inductive order: 8 points
- Base case: 8 points
  - Correct: 8/8
  - Requiring Changes: 4/8
  - Wrong: 0/8
- Inductive case (marking or declaration of inductive hypothesis is graded separately): 8 points
  - Correct: 8/8
  - Requiring Changes: 4/8
  - Wrong: 0/8
- Marking or declaration of the inductive hypothesis: 2 points.
  - Mark the inductive hypothesis clearly: 2/2
  - State the implication of the inductive hypothesis clearly, maybe in the beginning of a case: 2/2
  - Otherwise: 0/2

Bonus Problem

10/10 if mathematically correct, 5/10 if "Require Changes", and 0/10 points if Wrong.