Homework 5

Discrete Structures
Due Tuesday, February 9, 2021

Submit each question separately in .pdf format (except question 5)

---

1. (a) Trace out the values i, j, m for the binary search algorithm (Algorithm 3 on page 206) on the integer 10 and the list $1, 3, 4, 5, 8, 10, 11, 12, 15$.
   (b) Trace out the list values a_i in the list $a_1 = 6, a_2 = 3, a_3 = 8, a_4 = 2, a_5 = 1, a_6 = 4, a_7 = 10$ for the bubble sort algorithm (Algorithm 4 on page 208).
   (c) Trace out the values s, j for the naive string matching algorithm (Algorithm 6 on page 209) on the strings $t =$ `mississippi` and $p =$ `si`.

2. The *ternary search algorithm* locates an element in a list of strictly increasing integers by successively splitting the input list into three sublists of equal size, and restricting the search to the sublist in which the target integer lies.
   (a) Implement the ternary search algorithm in pseudocode, writing an algorithm that locates the given element in a list or reports that it does not exist. Follow the example of the binary search algorithm on page 206.
   (b) What is the worst case for this algorithm? Give an example input.
   (c) How many comparisons does this algorithm need in the worst case?

3. For each function $f(n)$ defined below, find the optimal $g(n)$ such that $f(n)$ is $O(g(n))$, and find C, n_0, such that $|f(n)| < C \cdot |g(n)|$ as long as $n > n_0$.
   (a) $f(n) = 3n^4 + \log_2(n^8)$
   (b) $f(n) = \sum_{k=1}^{n}(k^3 + k)$
   (c) $f(n) = (n + 2)\log_2(n^2 + 1) + \log_2(n^3 + 1)$
   (d) $f(n) = n^3 + \sin(n^7)$

4. Assume that you have n coins; it is known that $n-1$ of these coins have equal weight, but one of them is heavier than the others. The input to the algorithm is a list of n integer variables representing the weights of the coins.
   Note. An algorithm to find the maximum coin in a list of $a_1, a_2, \ldots, a_n$ is given in the textbook (Algorithm 1 on page 203); it needs $n - 1$ comparisons between individual numbers/coins.

   You have a generalized comparison function that behaves like two-sided balance scales:

$$\text{compare}(\text{list}_1, \text{list}_2) = \begin{cases} -1, & \text{if } S_1 < S_2, \\ 0, & \text{if } S_1 = S_2, \\ 1, & \text{if } S_1 > S_2, \end{cases} \quad \text{where } S_1 = \sum_{a_i \in \text{list}_1} a_i, \quad S_2 = \sum_{a_j \in \text{list}_2} a_j.$$

   Namely, you are allowed to compare any two groups of coins (of sizes $1, 2, \ldots, \lfloor n/2 \rfloor$ each); and the scales will tell you, if first group is lighter, same or heavier than the other group.

   (a) Describe an algorithm that shows how to find the heaviest coin among n coins, if all the others have the same weight. You can write pseudocode or just explain precise steps in English.

(b) Find the times you call "compare($\text{list}_1, \text{list}_2$)". Express the number of calls as a function of n (the worst-case estimate).

(c) Show that you used as few calls to "compare($\text{list}_1, \text{list}_2$)" as possible.

5. Complete the proofs in Coq. You may use the non-constructive `classic` and `NNPP` axioms if needed, but try to minimize their use. Submit your file as plain-text `hw5_question5.v`.

```
Section Predicate_Logic_Examples.

(* A is a nonempty set (containing element 'something' *)
Variables A : Set.
Variables something: A.
(* Assume that P,Q are 1-argument predicates defined on A *)
Variables P Q : A->Prop.

(* Can distribute 'exists' quantifier over a disjunction *)
Lemma sample5_1:
    (exists (y:A), (P y)) \/ (exists (y:A), (Q y)) <->
    exists (x:A), (P x) \/ (Q x).
Proof.
  (* Insert a proof; then replace 'Admitted' by 'Qed' *)
  Admitted.

(* A variant of De Morgans law *)
Lemma sample5_2:
    (exists (x:A), ~~(P x)) <-> ~(forall (y:A), ~(P y)).
Proof.
  Admitted.

(* If (P x) always implies (Q x), then the existence
   of some (P x0) leads to existence of some (Q x1) *)
Lemma sample5_3:
    (forall (x:A), P x -> Q x) ->
        ((exists (x:A), (P x)) -> exists (x:A), (Q x)).
Proof.
  Admitted.

(* If P being true sometimes implies that also Q is true sometimes,
   then there is some x0 for which (P x) implies (Q x) *)
Lemma sample5_4: ((exists (x:A), (P x)) -> (exists (x:A), (Q x))) ->
    (exists (x:A), ((P x) -> (Q x))).
Proof.
  Admitted.

(* If P(x) always implies Q(x), and P(x) is always true,
   then Q(x) is always true. *)
Lemma sample5_5: (forall (x:A), ((P x) -> (Q x))) ->
    ((forall (x:A), (P x)) -> forall (x:A), (Q x)).
Proof.
  Admitted.

End Predicate_Logic_Examples.
```