Discrete Structures (CSCI 246)

Homework 4

Purpose & Goals

The following problems provide practice relating to:

- the Pigeonhole principle,
- mathematical induction,
- mathematical definitions (functions), and
- the problem solving process.

Submission Requirements

- **Type or clearly hand-write your solutions** into a pdf format so that they are legible and professional. Submit your pdf to Gradescope. **Illegible, non-pdf, or emailed solutions will not be graded.**
- Each problem should start on a new page of the document. When you submit to Gradescope, associate each page of your submission with the correct problem number. Please post in Discord if you are having any trouble using Gradescope.
- Try to model your formatting off of the proofs from lecture and/or the textbook.
- Submit to Gradescope early and often so that last-minute technical problems don't cause you any issues. Only the latest submission is kept. Per the syllabus, assignments submitted within 24 hours of the due date will receive a 25% penalty and assignments submitted within 48 hours will receive a 50% penalty. After that, no points are possible.

Academic Integrity

- You may work with your peers, but **you must construct your solutions in your own words on your own.**
- Do not search the web for solutions or hints, post the problem set, or otherwise violate the course collaboration policy or the MSU student code of conduct.
- Violations (regardless of intent) will be reported to the Dean of Students, per the MSU student code of conduct, and you will receive a 0 on the assignment.

Tips

- Answer each problem to the best of your ability. Partial credit is your friend!
- Read the hints for where to find relevant examples for each problem.
- Refer to the problem solving and homework tips guide.
- It is not a badge of honor to say that you spent 5 hours on a single problem or 15 hours on a single assignment. Use your time wisely and get help (see "How to Get Help" below).

How to Get Help

When you are stuck and need a little or big push, **please ask for help!**

- Timebox your effort for each problem so that you don't spend your life on the problem sets. (See the problem solving tips guide for how to do this effectively.)
- Post in Discord. If you're following the timebox guide, you can post the exact statement that you produced after spending 20 minutes being stuck.
- Come to office hours or visit the CS Student Success Center.

Problem 1 (12 points)

Let $f : \mathbb{Z} \to \mathbb{Z} \times \mathbb{Z}$ be defined by $f(n) = \langle 3n, n + 4 \rangle$. f is a function. (You can check the three properties for yourself if you would like.)

(a) (6 points) Is f onto?

(b) (6 points) Is f one-to-one?

Grading Notes. Each of the above is graded out of 6 points, broken down as described below.

(1) Correctly decide if the statement is true or false.

(5) Correctly prove your claim.

- A proof requires clearly stated facts and explanations.
- A disproof requires a clearly stated counterexample along with an explanation of why the counterexample is a counterexample.

Problem 2 (8 points)

Suppose we have 20x4 table of 20 length-four binary strings. Note that `1010` is a length-four binary string, whereas `0123` is not (because it includes values other than `0` and `1`) and `10101` is not (because its length is not four).

Use the Pigeonhole Principle to prove that there are is a duplicate in the table.

Hint: When using the Pigeonhole Principle, always

- clearly define your set A (of pigeons),
- clearly define your set B (of pigeonholes),
- clearly define the function $f : A \to B$ that maps each pigeon $a \in A$ to a single pigeonhole $f(a)$ so that $f(a) \in B$ (i.e., f has the three properties of a well-defined function), and
- explain how you're able to apply the Pigeonhole Principle to obtain the desired result.

Grading Notes. While a detailed rubric cannot be provided in advance as it gives away solution details, the following is a general idea of how points are distributed for this problem. We give partial credit where we can.

(7) **Correctness.** If your proof is not correct, this is where you'll get docked. You'll need to

- (1) define the pigeons (or set A),
- (1) define the pigeonholes (or set B),
- (1) define the function $f : A \to B$,
- (1) briefly explain why f is well-defined,
- (1) explain why $|A| > |B|$,
- (1) correctly apply the PHP,
- (1) explain how the resulty of the PHP achieves the desired result of the problem.

(1) **Communication.** We need to see a mix of notation and intuition. If you skip too many steps at once, or we cannot follow your proof, or if your proof is overly wordy or confusing, this is where you'll get docked.

Problem 3 (8 points)

One of the most important data structures in computer science is a *hash table*. Given a set U of possible values and a set $S \subseteq U$ of values from U, a hash table allows us to quickly answer the question "is x in S?" without needing to examine every element of S. A hash table is defined as follows:

- Let $T[1 \dots n]$ be a table with n cells.
- Let $h : U \to \{1, 2, \dots, n\}$ be a function (called a *hash function*).
- Each element $x \in S$ is stored in the cell $T[h(x)]$.

To check that a new value $y \in U$ is in S, we can compute $h(y)$, look at the cell $T[h(y)]$, and see if y is there.

Consider the set $U = \{x \in \mathbb{Z} : -100 \leq x < 100\}$ and a table $T[1 \dots 100]$.

Use the PHP to prove that *any* well-defined function $h : U \to \{1, 2, \dots, 100\}$ will have at least one collision (where two elements from U are stored in the same cell of T).

Hint: When using the Pigeonhole Principle, always

- clearly define your set A (of pigeons),
- clearly define your set B (of pigeonholes),
- clearly define the function $f : A \to B$ that maps each pigeon $a \in A$ to a single pigeonhole $f(a)$ so that $f(a) \in B$ (i.e., f has the three properties of a well-defined function), and
- explain how you're able to apply the Pigeonhole Principle to obtain the desired result.

Grading Notes. While a detailed rubric cannot be provided in advance as it gives away solution details, the following is a general idea of how points are distributed for this problem. We give partial credit where we can.

(7) **Correctness.** If your proof is not correct, this is where you'll get docked. You'll need to

- (1) define the pigeons (or set A),
- (1) define the pigeonholes (or set B),
- (1) define the function $f : A \to B$,
- (1) briefly explain why f is well-defined,
- (1) explain why $|A| > |B|$,
- (1) correctly apply the PHP,
- (1) explain how the resulty of the PHP achieves the desired result of the problem.

(1) **Communication.** We need to see a mix of notation and intuition. If you skip too many steps at once, or we cannot follow your proof, or if your proof is overly wordy or confusing, this is where you'll get docked.

Problem 4 (16 points)

For a given integer $n \geq 0$, consider the sum of the first n squares. That is, $1^2 + 2^2 + \cdots + n^2$. After trying some examples, you might hypothesize that the sum of the first n squares is equal to $n(n+1)(2n+1)/6$.

(a) (1 point) Show that this formula works for $0 \leq n \leq 3$.

Grading Notes. This one is simple: compute the sum of squares and the formula for each one correctly, and note that they are equal.

(b) (15 points) Use mathematical induction to prove that the formula works for any integer $n \geq 0$.

Hint: Try to model your proof off of the proofs by mathematical induction from lecture.

Grading Notes. While a detailed rubric cannot be provided in advance as it gives away solution details, the following is a general idea of how points are distributed for this problem. We give partial credit where we can.

(13) **Correctness.** If your proof is not correct, this is where you'll get docked. You'll need to
- (1) define the predicate $P(n)$ that you are proving,
- (1) state the variable you are performing induction over,
- (1) state the base case,
- (1) prove the base case,
- (1) state the inductive case,
- (7) prove the inductive case
  - (1) assume the inductive hypothesis ($P(n-1)$)
  - (1) start with LHS of $P(n)$ in order to manipulate to RHS (or vice versa). Do not start with LHS=RHS!
  - (1) find a way to get $P(n-1)$ in your algebra somewhere, so you can
  - (1) correctly apply the inductive hypothesis,
  - (1) clearly noting that you have done so.
  - (2) Then, successfully derive RHS of $P(n)$ (if you started with LHS) or LHS (if you started with RHS).
- (1) Finish with a one-sentence conclusion that pulls it all together.

(2) **Communication.** We need to see a mix of notation and intuition. If you skip too many steps at once, or we cannot follow your proof, or if your proof is overly wordy or confusing, this is where you'll get docked.