# Chapter 4 Review

Tuesday, April 20, 2021 2:33 PM

Chapter 4: Elementary Number Theory and Methods of Proof

Section 4.1: Direct Proof and Counterexample 1: Introduction

Even and Odd

| Even | Odd |
|---|---|
| n is even ⇔ ∃ an integer k such that n = 2k | n is odd ⇔ ∃ an integer k such that n = 2k + 1 |

- Def: An integer n is **even** IFF n equals twice some integer
- Def: An integer n is **odd** IFF n equals twice some integer plus one.

Prime and Composite

| Prime | Composite |
|---|---|
| n is prime ⇔ ∀ positive integer r and s, if n = rs then either r = 1 and s = n or r = n and s = 1 | n is composite ⇔ ∃ positive integers r and s such that n = rs and 1 < r < n and 1 < s < n |

- Def: An integer n is **prime** IFF n > 1 and for all positive integers r and s, if n = rs, then either r or s equals n.
- Def: An integer n is **composite** IFF n > a and n = rs for some integers r and s with 1 < r < n and 1 < s < n.

Theorem 4.1.1: The sum of any two even integers is even.

Existential Instantiation

- Def: If the existence of a certain kind of object is assumed or has been deduced the it can be given a name, as long as that name is not currently being used to denote something else.

Proving/Disproving Existential Statements

Proofs of Existential Statements

- Existential Statement: ∃x ∈ D such that Q(x)
  - Constructive Proofs of Existence
    1. Find an x in D that makes Q(x) true.
    2. Give directions for finding an x in D that makes Q(x) true.
  - Nonconstructive Proofs of Existence
    1. Show that the existence of a value of x that makes Q(x) true is guaranteed by an axiom or a previously proved theorem.
    2. Show that the assumption that there is no such x leads to a contradiction.

Disproving Existential Statements

- Prove the negation of the existential statement, which will be a universal statement.

Proving/Disproving Universal Statements

| Proof | Disproof |
|---|---|
| Method of Exhaustion (3.1)<br>Method of Generalizing from the Generic Particular<br>Method of Direct Proof | Counterexample (3.1) |

Disproving a Universal Conditional Statement by Counterexample

- Def: To disprove a statement of the form "∀x ∈ D, if P(x) then Q(x)" find a value of x in D for which the hypothesis P(x) is true and the conclusion Q(x) is false. Such an x is called a **counterexample**.

Proving Universal Statements

Method of Generalizing from the Generic Particular

- Def: To show that every element of a set satisfies a certain property, suppose x is a particular but arbitrarily chosen element of the set, and show that x satisfies the property.

Method of Direct Proof

1. Express the statement to be proved in the form "∀x ∈ D, if P(x) then Q(x)". (This step is done mentally)
2. Start the proof by supposing x is a particular but arbitrarily chosen element of D for which the hypothesis P(x) is true. (This step is often abbreviated "Suppose x ∈ D and P(x)")
3. Show that the conclusion Q(x) is true by using definitions, previously established results, and the rules for logical inference.

Directions for Writing Proofs of Universal Statements

1. Copy the statement of the theorem to be proved on your paper
2. Clearly mark the beginning of your proof with the word Proof.
3. Make you proof self-contained. Meaning explain the meaning of each variable used in the proof in the body of the proof ("Suppose … " or "Let …").
4. Write your proof in complete, grammatically correct sentences.
5. Keep your reader informed about the status of each statement in your proof.
6. Give a reason for each assertion in your proof.
7. Include the "little words and phrases" that make the logic of your arguments clear.
8. Display equations and inequalities.

Section 4.2: Direct Proof and Counterexample 2: Rational Numbers

Rational and Irrational Numbers

| Rational | Irrational |
|---|---|
| r is rational ⇔ ∃ integers a and b such that r = $\frac{a}{b}$ and b ≠ 0 | r is irrational ⇔ ∀ integers a and b such that ~(r = $\frac{a}{b}$ and b ≠ 0) |

- Def: A real number r is **rational** IFF it can be expressed as a quotient of two integers with a nonzero denominator.
- Def: A real number that is not rational is **irrational**.

Zero Product Property: If neither of two real numbers is zero, then their product is also not zero.

Theorem 4.2.1: Every integer is a rational number.

Theorem 4.2.2: The sum of any two rational numbers is rational.

Corollary

- Def: A **corollary** is a statement whose truth can be immediately deduced from a theorem that has already been proved.

Corollary 4.2.3: The double of a rational number is rational.

Section 4.3: Direct Proof and Counterexample 3: Divisibility

Number Theory: Divisibility

- If n and d are integers and d ≠ 0, then n is **divisible by** d IFF n equals d times some integer.
- Instead of "n is divisible by d" we can say that
  - "n **is a multiple of** d"
  - "d **is a factor of** n"
  - "d **is a divisor of** n"
  - "d **divides** n"
- The notation **d|n** is read "d divides n". (d∤n means "d does not divide n")
- Symbolically, if n and d are integers and d ≠ 0: d|n ⇔ ∃ an integer k such that n = dk.
- Symbolically, if n and d are integers and d ≠ 0: d∤n ⇔ $\frac{n}{d}$ is not an integer
- Note: Think of "divides" as a relation, not an operator!

A Positive Divisor of a Positive Integer (Theorem 4.3.1): For all integers a and b, if a and b are positive and a divides b, then a ≤ b.

Divisors of 1 (Theorem 4.3.2): The only divisors of 1 are 1 and -1.

Transitivity of Divisibility (Theorem 4.3.3): For all integers a, b, and c, if a divides b and b divides c, then a divides c.

Divisibility by a Prime (Theorem 4.3.4): Any integer n > 1 is divisible by a prime number.

Unique Factorization of Integers Theorem / Fundamental Theorem of Arithmetic (Theorem 4.3.5)

- Given any integer n > 1, there exists a positive integer k, distinct prime numbers $p_1$, $p_2$,…, $p_k$, and positive integers $e_1$, $e_2$,…, $e_k$ such that
  n = $p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$,
  and any other expression for n as a product of prime numbers is identical to this except, perhaps, for the order in which the factors are written.
- Note that n = $p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$ is the **standard factored form** of n

Section 4.4: Direct Proof and Counterexample 4: Division into Cases and the Quotient-Remainder Theorem

The Quotient-Remainder Theorem (Theorem 4.4.1)

- Given any integer n and positive integer d, there exist unique integers q and r such that n = dq + r and 0 ≤ r < d

Div and Mod

- Def: Given an integer n and a positive integer d,
  - □ **n div d** = the integer quotient obtained when n is divided by d
  - □ **n mod d** = the nonnegative integer remainder obtained when n is divided by d
- Symbolically, if n and d are integers and d > 0, then
  - □ n div d = q and n mod d = r ⇔ n = dq + r

  where q and r are integers and $0 \leq r < d$

Parity Property: Any integer is either even or odd.

The Parity Property (Theorem 4.4.2): Any two consecutive integers have opposite parity.

Method of Proof by Division into Cases

- To prove a statement of the form "If $A_1$ or $A_2$ or … or $A_n$, then C" prove all of the following:

  If $A_1$, then C,
  If $A_2$, then C,
  …
  If $A_n$, then C.
- This process shows that C is true regardless of which $A_1$, $A_2$, … , $A_n$ happens to be the case.

Theorem 4.4.3: The square of any odd integer has the form 8m + 1 for some integer m.

Absolute Value

- Def: For any real number x, the **absolute value of x**, denoted |x|, is defined as follows:

$$|x| = \begin{cases} x \ \ if\ x \geq 0 \\ -x\ if\ x < 0 \end{cases}$$

Lemma 4.4.4: For all real numbers r, $-|r| \leq r \leq |r|$

Lemma 4.4.5: For all real numbers r, |-r| = |r|

The Triangle Inequality (Theorem 4.4.6): For all real numbers x and y, $|x + y| \leq |x| + |y|$

Section 4.5: Direct Proof and Counterexample 5: Floor and Ceiling

Floor ($\lfloor x \rfloor$)

- Def: Given any real number x, the **floor of x**, denoted $\lfloor x \rfloor$, is defined as follows:

  $\lfloor x \rfloor$ = that unique integer n such that $n \leq x < n + 1$
- Symbolically, if x is a real number and n is an integer, then

  $\lfloor x \rfloor = n \Leftrightarrow n \leq x < n + 1$

Theorem 4.5.1: For all real numbers x and all integers m, $\lfloor x + m \rfloor = \lfloor x \rfloor + m$

The Floor of n/2 (Theorem 4.5.2): For any integer n,

$$\left\lfloor \frac{n}{2} \right\rfloor = \begin{cases} \dfrac{n}{2} \ \ if\ n\ is\ even \\ \dfrac{n-1}{2}\ if\ n\ is\ odd \end{cases}$$

Theorem 4.5.3: If n is any integer and d is a positive integer, and if $q = \lfloor n/d \rfloor$ and $r = n - d\lfloor n/d \rfloor$, then n = dq + r and $0 \leq r < d$

How to Calculate Div and Mod with Floor (based on Theorem 4.5.3)

- For any nonnegative integer n and a positive integer d,
  - $n \; div \; d \; = \left\lfloor \frac{n}{d} \right\rfloor$
  - $n \; mod \; d \; = n - d \left\lfloor \frac{n}{d} \right\rfloor$

Ceiling ($\lceil x \rceil$)

- Def: Given any real number x, the **ceiling of x**, denoted $\lceil x \rceil$, is defined as follows:
  $\lceil x \rceil$ = that unique integer n such that n - 1 < x ≤ n
- Symbolically, if x is a real number and n is an integer, then
  $\lceil x \rceil$ = n ⇔ n - 1 < x ≤ n

## Section 4.6: Indirect Argument: Contradiction and Contraposition

Method of Proof by Contradiction

1. Suppose the statement to be proved is false. That is, suppose that the negation of the statement is true.
2. Show that this supposition leads logically to a contradiction.
3. Conclude that the statement to be proved is true.

Method of Proof by Contraposition

1. Express the statement to be proved in the (universal conditional) form "∀x in D, if P(x) then Q(x)". (This step may be done mentally)
2. Rewrite this statement in the contrapositive form "∀x in D, if Q(x) is false then P(x) is false". (This step may be done mentally)
3. Prove the contrapositive by a direct proof.
   a. Suppose x is a (particular but arbitrarily chosen) element of D such that Q(x) is false.
   b. Show that P(x) is false.

Relation between Proof by Contradiction and Proof by Contraposition

- Any proof by contraposition can be recast in the language of proof by contradiction.

| Statement |
|---|
| "∀x in D, if P(x) then Q(x)" |
| Proof by Contraposition |
| Suppose $x$ is an arbitrary element of $D$ such that $\sim Q(x)$. → sequence of steps → $\sim P(x)$ |
| Proof by Contradiction |
| Suppose $\exists x$ in $D$ such that $P(x)$ and $\sim Q(x)$. → same sequence of steps → Contradiction: $P(x)$ and $\sim P(x)$ |

Proposition

- Def: A **proposition** refers to a statement that is somewhat less consequential than a theorem but is nonetheless worth noting.

New Theorems and Propositions

Theorem 4.6.1: There is no greatest integer.

Theorem 4.6.2: There is no integer that is both even and odd.

Theorem 4.6.3: The sum of any rational number and any irrational number is irrational.

Proposition 4.6.4: For all integers n, if $n^2$ is even then n is even.

Section 4.7: Indirect Argument: Two Classical Theorems

Irrationality of $\sqrt{2}$ (Theorem 4.7.1): $\sqrt{2}$ is irrational.

Proposition 4.7.2: $1 + 3\sqrt{2}$ is irrational.

Proposition 4.7.3: For any integer a and any prime number p, if p|a then p∤(a+1)

Infinitude of the Primes (Theorem 4.7.4): The set of prime numbers is infinite.

When to Use Indirect Proof

- In the absence of obvious clues suggesting indirect argument,
  - Try first to prove a statement directly.
  - Then, if that does not succeed, look for a counterexample.
  - If the search for a counterexample is unsuccessful, look for a proof by contradiction or contraposition.

Section 4.8: Application: Algorithms

Algorithmic Language

Variable

- Def: The term **variable** is used to refer to a specific storage location in a computer's memory.

Data Type

- Def: The **data type** of a variable indicates the set in which the variable takes its values.

Assignment Statement

- Def: An **assignment statement** gives a value to a variable.
- It has the form "x := e", where x is a variable and e is an expression. This is read "x is assigned the value e" or "let x be e".
- When an assignment statement is executed, the expression e is evaluated (using the current values of all the variables in the expression), and then its value is placed in the memory location corresponding to x (replacing any previous contents of this location).

Groups of Algorithm Statements

- Generally use indentation to indicate that statements belong together as a unit.
- When ambiguity is possible, we may explicitly bind a group of statements together into a unit by preceding the group with the word "do" and following it with the words "end do"

Conditional Statements

- Uses the current values of program variables to determine which algorithm statement will be executed next.
- Conditional statements can be either "if-then-else" statements or "if-then" statements.

| "if-then-else" statement | "if-then" statement |
|---|---|
| **if** (*condition*)<br>**then** $s_1$<br>**else** $s_2$ | **if** (*condition*) **then** $s_1$ |

  - Where condition (also called a guard) is a predicate involving algorithm variables and where s1 and s2 are algorithm statements or groups of algorithm statements.

Iterative Statements

- Iterative statements are used when a sequence of algorithm statements is to be executed over and over again.
- There are two types of iterative statements: "while" loops and "for-next" loops

| "while" loop | "for-next" loop |
|---|---|
| **while** (*condition*)<br>*[statements that make up the body of the loop]*<br>**end while** | **for** *variable* := *initial expression*<br>**to** *final expression*<br>*[statements that make up the body of the loop]*<br>**next** *(same) variable* |

  - Note that the "for-next" loop will automatically add 1 to the variable after each iteration.
- A loop is said to be iterated each time the statements in the body of the loop are executed. Each execution of the body of the loop is called an iteration of the loop.

Trace Table

- Shows the current values of algorithm variable at various points during execution.
- Lists the variable names on the left-most column and lists the iteration number above the other columns.
- E.g.

| | Iteration Number | | |
|---|---|---|---|
| | 0 | 1 | 2 |
| Variable Name *i* | 1 | 2 | 3 |
| *s* | 0 | 1 | 3 |

What to Include when Describing an Algorithm Formally

- The name of the algorithm, together with a list of input and output variables.
- A brief description of how the algorithm works.
- The input variable names, labeled by data type.
- The statements that make up the body of the algorithm, possibly with explanatory comments.
- The output variable names, labeled by data type.

The Division Algorithm (Algorithm 4.8.1)

**Algorithm 4.8.1 Division Algorithm**

*[Given a nonnegative integer $a$ and a positive integer $d$, the aim of the algorithm is to find integers $q$ and $r$ that satisfy the conditions $a = dq + r$ and $0 \leq r < d$. This is done by subtracting $d$ repeatedly from $a$ until the result is less than $d$ but is still nonnegative.*

$$0 \leq a - d - d - d - \cdots - d = a - dq < d.$$

*The total number of $d$'s that are subtracted is the quotient $q$. The quantity $a - dq$ equals the remainder $r$.]*

**Input:** *$a$ [a nonnegative integer], $d$ [a positive integer]*

**Algorithm Body:**

$r := a, q := 0$

*[Repeatedly subtract $d$ from $r$ until a number less than $d$ is obtained. Add 1 to $q$ each time $d$ is subtracted.]*

**while** $(r \geq d)$

$r := r - d$

$q := q + 1$

**end while**

*[After execution of the* **while** *loop, $a = dq + r$.]*

**Output:** *$q, r$ [nonnegative integers]*

Greatest Common Divisor

- Def: Let a and b be integers that are not both zero. The **greatest common divisor of** a and b, denoted gcd(a, b), is that integer d with the following properties:
  1. d is a common divisor of both a and b. In other words, d|a and d|b.
  2. For all integers c, if c is a common divisor of both a and b, then c is less than or equal to d. In other words, for all integers c, if c|a and c|b, then $c \leq d$.

Lemma 4.8.1: If r is a positive integer, the gdc(r, 0) = r.

Lemma 4.8.2: If a and b are any integers not both zero, and if q and r are any integers such that a = bq + r, then gdc(a, b) = gdc(b, r).

The Euclidean Algorithm (Algorithm 4.8.2)

**Algorithm 4.8.2 Euclidean Algorithm**

*[Given two integers A and B with $A > B \geq 0$, this algorithm computes* gcd$(A, B)$*. It is based on two facts:*

1. gcd$(a, b)$ = gcd$(b, r)$ *if* $a, b, q,$ *and* $r$ *are integers with* $a = b \cdot q + r$ *and* $0 \leq r < b$.
2. gcd$(a, 0) = a$.*]*

**Input:** $A, B$ *[integers with* $A > B \geq 0$*]*

**Algorithm Body:**

$a := A, b := B, r := B$

*[If* $b \neq 0$*, compute* $a$ mod $b$*, the remainder of the integer division of* $a$ *by* $b$*, and set* $r$ *equal to this value. Then repeat the process using* $b$ *in place of* $a$ *and* $r$ *in place of* $b$.*]*

**while** $(b \neq 0)$

$r := a \bmod b$

*[The value of* $a$ mod $b$ *can be obtained by calling the division algorithm.]*

$a := b$

$b := r$

**end while**

*[After execution of the* **while** *loop,* gcd$(A, B) = a$.*]*

gcd $:= a$

**Output:** gcd *[a positive integer]*

1. Let A and B be integers with A > B ≥ 0.
2. To find the greatest common divisor of A and B, first check whether B = 0. If it is, then gcd(A, B) = A by Lemma 4.8.1. If it isn't, then B > 0 and the quotient remainder theorem can be used to divide A by B to obtain a quotient q and a remainder r: A = Bq + r where 0 ≤ r < B. By Lemma 4.8.2, gcd(A, B) = gdc(B, r). Thus the problem of finding the greatest common divisor of A and B is reduced to the problem of finding the greatest common divisor of B and r.
3. Now just repeat the process, starting again at (2), but use B instead of A and r instead of B.

```
gdc(A, B) =
        if (B != 0)
                then
                        r := A mod B  = A − B ⌊A/B⌋
                        return gdc(B, r)
                else
                        return A
```