CMPT-335
Discrete Structures

Spring 2019

Organizational Details

Class Meeting:
12:00pm-1:15pm, Monday, Thursday, Room RLC-205

Instructor: Dr. Igor Aizenberg

Office: RLC-203B
Phone 718-862-7425
e-mail: igor.aizenberg@manhattan.edu

Office hours:
Monday, Thursday 2:00pm – 3:00pm

Class Web Page: http://www.freewebs.com/igora/CMPT-335.htm

Dr. Igor Aizenberg: self-introduction

- MS in Mathematics from Uzhgorod National University (Ukraine), 1982
- PhD in Computer Science from the Academy of Sciences of the Soviet Union, 1986
- Areas of research: Artificial Neural Networks, Pattern Recognition and Image Processing
- More than 100 journal and conference proceedings publications and two research monographs
- Job experience: Academy of Sciences of the Soviet Union(1982-1990); Uzhgorod National University (Ukraine,1990-1996 and 1998-1999); Catholic University of Leuven (Belgium, 1996-1998); Company “Neural Networks Technologies” (Israel, 1999-2002); Dortmund University of Technology (Germany, 2003-2005); Tampere University of Technology (Finland, 2005-2006); Texas A&M University-Texarkana (Texarkana, TX, 2006-2016), Manhattan College (from August, 2016)
- http://www.freewebs.com/igora/ - personal web page
- https://manhattan.edu/campus-directory/iaizenberg01 - official web page

Text Book

- **Kenneth Rosen, Discrete Mathematics and Its Applications, 7/e (2012), McGraw Hill, 2012.**
- **ISBN: 978-0-07-338309-5**

Methods of Evaluation

- **Tests:**

Midterm Test: **March**

Final Exam: **May**

- **Homework** (homework assignments, **which will be due** (not all of them will be due), **will be graded**)

Grading

Grading Method

Midterm Test	25%
Final Exam	35%
Homeworks and preparation	40%

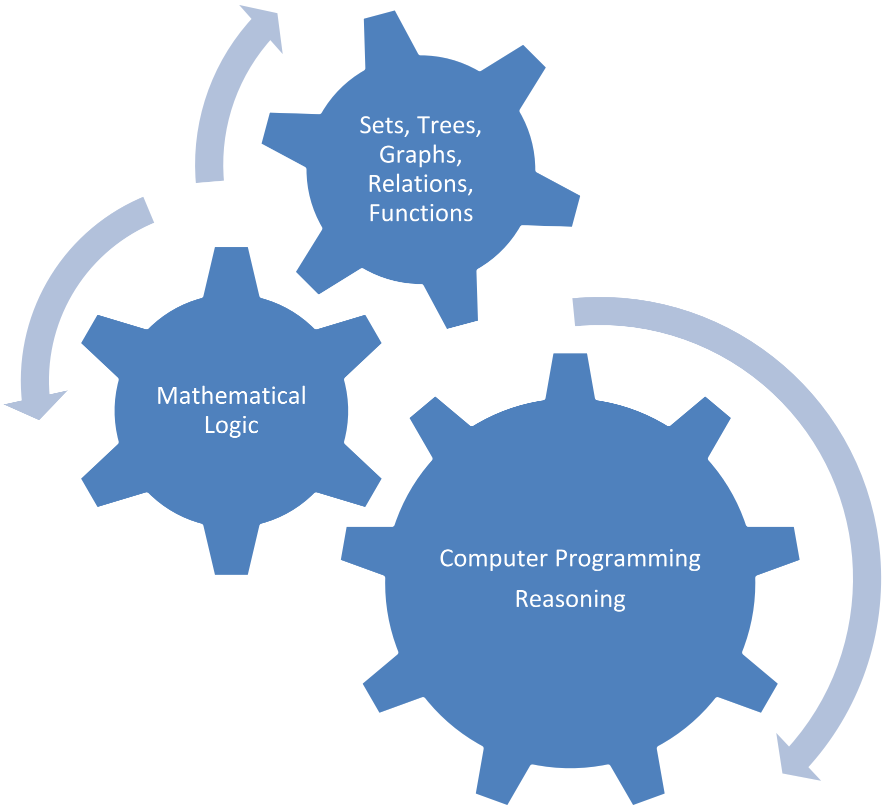
Grading Scale:

93%+ → A
90%+ → A-
85%+ → B+
80%+ → B
78%+ → B-
72%+ → C+
67%+ → C
64%+ → C-
60%+ → D+
58%+ → D
less than 58% → F

Interaction

- Very important!
- The only stupid question is the one that is left unasked.
  ASK!!!
  In class, in my office, after class…
- **Do not hesitate to ask!**

Discrete Structures: What it is?



Discrete Structures: What it is?

- Discrete Structures include those mathematical concepts and mechanisms, which are widely used in the computer programming, modeling, and simulations
- A discrete nature of a digital computer requires consideration of discrete rather than continuous models
- Since to solve any problem using a computer, a proper model must be developed first, discrete structures and methods, which are considered in Discrete Mathematics, are very important.

Discrete Mathematics: What it is?

- Discrete Mathematics is somewhat “opposite” (but not alternative!) to Calculus
- While in Calculus we consider continuous objects and continuity as a fundamental principle, in Discrete Mathematics we consider discrete objects, structures, and their relationships
- Both Continuous and Discrete Mathematics are very important. They compliment each other

Discrete Structures and Methods of Discrete Mathematics: Main Chapters

- Mathematical Reasoning (mathematical logic, methods of proof)
- Discrete Structures (abstract mathematical structures – sets, graphs, trees – that are used to represent discrete objects)
- Algorithmic Thinking (methods that are used for algorithms design and specification, verification of their correctness)
- Applications and Modeling (the use of Discrete Math methods for simulations and for modeling a variety of real-world problems)

What we will study?

- Basic concepts of discrete structures and methods of discrete mathematics, which are used in computer modeling and simulation, in computer programming, computer engineering and systems analysis:
- Elements of Mathematical Logic
- Elements of Sets Theory
- Relations and Function theory
- Mathematical Induction
- Modular Arithmetic and Elements of Cryptography
- Graphs and Trees
- Boolean Functions

PROPOSITIONAL LOGIC

Propositional Logic

- This Chapter is very important for understanding fundamentals of mathematical reasoning, artificial intelligence, algorithm design and programming
- Propositional Logic, which is a part of mathematical logic, is a key tool in algorithms design and programming, verification of the correctness of algorithms and programs
- Mathematical logic is also used in computer circuits design

Proposition

- A **proposition** is a declarative sentence (that is, a sentence that declares a fact) ,which is either **true** or **false**, but not both or uncertain
- Examples of propositions:

➢ Washington DC is a capital of the USA

➢ Today is Thursday

➢ 1+5=3

➢ Yonkers is the biggest city in the world

Proposition

- Exercise. Which statements are propositions and which are not?

➢ x is a student

➢ We are in the Calculus class now

➢ $x+5=7$

➢ Tomorrow will be Friday

➢ It is a cloudy sky now

➢ $2+5=7$

➢ Are you a student?

Proposition

- Exercise. Which statements are propositions and which are not?

x is a student	not
We are in the Calculus class now	yes
$x+5=7$	not
Tomorrow will be Friday	yes
It is a cloudy sky now	yes
$2+5=7$	yes
Are you a student?	not

Propositional Variables. Truth Value

- Propositional variables (statement variables) are variables that represent propositions. We will use small English letters for propositional variables: p = "Today is Tuesday", q="2+3=6".
- The truth value of a proposition is true (1), if it is a true proposition and false (0), if it is false
- Often the truth value true is associated with T, while the truth value false is associated with F.

Propositional Logic

- The area of logic that deals with propositions is called the propositional calculus or propositional logic
- It was first developed by Greek philosopher Aristotle more than 2300 years ago

Logical Operations. Compound Propositions

- Logical operations are operations over propositions (main of them are negation (not), conjunction (and), disjunction (or), exclusive or)

- Compound propositions are formed from existing propositions using logical operations

Negation

- Let p be a proposition. The negation of p, denoted by ^p (also by $\overline{p}$) is the statement "it is not the case that p"
- The proposition $\overline{p}$ is read "not p". The truth value of $\overline{p}$ is the opposite of the truth value of p.

The Truth Table for the Negation of a Proposition	
p	^p
1	0
0	1

Negation

- p= "At least 10 students are in the class today"
- ^p="Less than 10 students are in the class today"
- q="2+3=5"

  ^q="$2+3\neq 5$"
- r="Today is Thursday"
- ^r="Today is not Thursday"

Conjunction

- Let p and q be propositions. The conjunction of p and q, denoted by $p \wedge q$ or p & q is the proposition "p and q". It is true only if both p and q are true and <u>false otherwise</u>

The Truth Table for the Conjunction of Two Propositions		
p	q	$p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

Conjunction

- p= "At least 10 students are in the class today"
- q="2+3=7"
- r="Today is Tuesday"
- p & q= "At least 10 students are in the class today" and "2+3=7"
- p & r= "At least 10 students are in the class today" and "Today is Tuesday"
- q & r = "2+3=7" and "Today is Tuesday"

Disjunction

- Let p and q be propositions. The disjunction of p and q, denoted by $p \vee q$ is the proposition "*p* or *q*". It is true when if only one of p and q is true and false when both p and q are false

The Truth Table for the Disjunction of Two Propositions		
p	q	$p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1

Disjunction

- p= "I am John Smith"
- q="2+3=7"
- r="Today is Tuesday"
- $p \vee q$ = "I am John Smith" or "2+3=7"
- $p \vee r$ = "I am John Smith" or "Today is Tuesday"
- $q \vee r$ = "2+3=7" or "Today is Tuesday"

Exclusive OR (XOR)

- Let p and q be propositions. The exclusive OR of p and q, denoted by $p \oplus q$ (or p xor q) is the proposition, which is true only when exactly one of p and q is true and false otherwise

The Truth Table for the Exclusive OR of Two Propositions		
p	q	$p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0

Exclusive OR

- *p*= “Only History majors may take the Discrete Structures course”
- *q*=“2+3=5”
- *r*=“Computer Science is my major”
- $p \oplus q$ = “Only History majors may take the Discrete Structures course” xor “2+3=5”
- $p \oplus r$= “Only History majors may take Discrete Structures course” xor “Computer Science is my major”
- $q \oplus r$ = “2+3=5” xor “Computer Science is my major”

Conditional Statements

- Conditional statements are used to combine propositions in such a way that one of them depends on another one or they are mutually dependent
- For example:

➢ "If I can teach Discrete Structures, then I know Discrete Structures"

➢ "If I know Discrete Structures, then I can teach Discrete Structures"

Implication

- Let *p* and *q* be propositions. The conditional statement $p \rightarrow q$ (implication) is the proposition "if *p*, then *q*". The implication $p \rightarrow q$ **is <u>false only when *p* is true and *q* is false</u>** and **true otherwise**.
  p is called the hypothesis, *q* is called the conclusion.

The Truth Table for the Implication $p \rightarrow q$		
p	q	$p \rightarrow q$
0	0	1
0	1	1
1	0	0
1	1	1

Implication

- The truth value of implication is determined by the following rule: "something, which is false, **cannot** follow from something, which is true, but something, which is true **can** follow from whatever".
- Example: "If I will be the President of the university, then I will lower tuition fee".

  If I really will be the President, but I will not lower tuition fee, then I break your expectations and my promise.

  In any other case, you should not blame me that my statement was false.

Implication

- Implication $p \rightarrow q$ can be expressed in several ways:
- If p, then q
- p is sufficient for q
- q is necessary for p
- p implies q
- q follows from p
- q unless ^p

Converse, Contrapositive, Inverse Statements

$p \to q$ — ➢ Statement

$q \to p$ — ➢ Converse statement

$\overline{q} \to \overline{p}$ — ➢ Contrapositive statement

$\overline{p} \to \overline{q}$ — ➢ Inverse Statement

The Truth Table for the Implication $p \to q$
and its "derivatives"

p	q	$\overline{p}$	$\overline{q}$	$p \to q$	$q \to p$	$\overline{q} \to \overline{p}$	$\overline{p} \to \overline{q}$
0	0	1	1	1	1	1	1
0	1	1	0	1	0	1	0
1	0	0	1	0	1	0	1
1	1	0	0	1	1	1	1

Biconditional Statement (Bi-Implication)

- Let *p* and *q* be propositions. The biconditional statement (bi-implication) $p \leftrightarrow q$ is the proposition "*p*, if and only if *q*". The biconditional statement $p \leftrightarrow q$ is true only when *p* and *q* have the same truth values, and is false otherwise

The Truth Table for the Bi-Implication $p \leftrightarrow q$		
$\boldsymbol{p}$	$\boldsymbol{q}$	$p \leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1

Biconditional Statement (Bi-Implication)

- Biconditional statement $p \leftrightarrow q$ can also be expressed in other ways:
- *p* is necessary and sufficient for *q*
- *q* is necessary and sufficient for *p*
- *p* iff *q*
- *q* iff *p*

Compound Propositions and Precedence of Logical Operations

- Compound propositions can be built up by connecting "elementary" compound propositions using logical connectives (operations), for example,

$$\left(p \vee \overline{q}\right) \rightarrow \left(p \wedge q\right); \left(p \vee (q \wedge r)\right) \rightarrow \left(\overline{p} \wedge r\right)$$

- It is important to keep the following precedence of logical operations:

  Negation, Conjunction, Disjunction, Implication, Bi-Implication : $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

Logic and Bit Operations

- A bit (binary digit) is a symbol with two possible values, namely 0 and 1.
- A variable x is called a Boolean variable if its value is either 0 or 1.
- A bit can be used to represent a truth value. Traditionally 0 is associated with "False" and 1 is associated with "True".

Boolean (Bit) Operations

- Boolean (bit) operations are logical operations over Boolean variables and constants (Negation, AND, OR, XOR, etc).
- Let x and y be Boolean variables. Then

Table for some of the Bit Operations						
x	y	$x \wedge y$	$x \vee y$	$x \oplus y$	$x \rightarrow y$	$x \leftrightarrow y$
0	0	0	0	0	1	1
0	1	0	1	1	1	0
1	0	0	1	1	0	0
1	1	1	1	0	1	1

Bitwise Boolean Operations

- A bit string is a sequence of zero or more bits. The length of this string is the number of bits in this string.
- Bitwise operation over two bit strings as a Boolean (logical) operations applied to the corresponding elements of both strings element (bit)- wise.

$$\begin{array}{cccc} 0 & 1 & 1 & 1 \\ \wedge & & & \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{array} \qquad \begin{array}{cccc} 0 & 1 & 1 & 1 \\ \vee & & & \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{array} \qquad \begin{array}{cccc} 0 & 1 & 1 & 1 \\ \oplus & & & \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{array}$$

Propositional Equivalences

- A compound proposition that is always true, no matter what the truth values of the propositions occur in it, is called a tautology.
- A compound proposition that is always false, no matter what the truth values of the propositions occur in it, is called a contradiction.
- A compound proposition that is neither a tautology nor a contradiction is called a contingency.

Tautology and Contradiction

- Example of a tautology is $p \vee \overline{p}$
- Example of a contradiction is $p \wedge \overline{p}$

Examples of a Tautology and a Contradiction			
p	$\overline{p}$	$p \vee \overline{p}$	$p \wedge \overline{p}$
0	1	1	0
1	0	1	0

Logical Equivalence

- The compound propositions that have the same truth values in all possible cases are called logically equivalent.
- Let *p* and *q* be propositions. They are logically equivalent if $p \leftrightarrow q$ is a tautology: $p \equiv q$ or $p \Leftrightarrow q$
- $\equiv (\Leftrightarrow)$ are not logical operations (connectives), they just denote a fact that $p \leftrightarrow q$ is a tautology.

De Morgan's Laws

- De Morgan's laws establish two very important equivalences. They are:

1) The negation of the conjunction is logically equivalent to the disjunction of the negations.

$$\overline{(p \wedge q)} \equiv \overline{p} \vee \overline{q}$$

2) The negation of the disjunction is logically equivalent to the conjunction of the negations.

$$\overline{(p \vee q)} \equiv \overline{p} \wedge \overline{q}$$

Proving Logical Equivalences

- A straightforward way to prove a logical equivalence is to construct the truth table for those compound propositions involved in the equivalence. Such a table contains 2^n rows, where n is a number of propositional variables involved in the corresponding propositions
- However, this is not a good way for proving. Such a table for $n>3$ becomes too big
- The best way for proving is to use logical equivalences (standard equivalencies or laws)

Important Logical Equivalences

$p \wedge T \equiv p$

$p \vee F \equiv p$

- Identity laws

$p \vee T \equiv T$

$p \wedge F \equiv F$

- Domination laws

$p \vee p \equiv p$

$p \wedge p \equiv p$

- Idempotent laws

$\overline{\overline{p}} \equiv p$

- Double negation law

$p \wedge q \equiv q \wedge p$

$p \vee q \equiv q \vee p$

- Commutative laws

$(p \vee q) \vee r \equiv p \vee (q \vee r)$

$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

- Associative laws

$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

- Distributive laws

Important Logical Equivalences

$$\overline{(p \wedge q)} \equiv \overline{p} \vee \overline{q}$$

$$\overline{(p \vee q)} \equiv \overline{p} \wedge \overline{q}$$

- De Morgan's laws

$$p \vee (p \wedge q) \equiv p$$

$$p \wedge (p \vee q) \equiv p$$

- Absorption laws

$$p \vee \overline{p} \equiv T$$

$$p \wedge \overline{p} \equiv F$$

- Negation laws

Important Logical Equivalences

$$p \to q \equiv \overline{p} \vee q$$

$$p \to q \equiv \overline{q} \to \overline{p}$$

$$p \vee q \equiv \overline{p} \to q$$

$$p \wedge q \equiv \overline{\left(p \to \overline{q}\right)}$$

$$\overline{\left(p \to q\right)} \equiv p \wedge \overline{q}$$

$$\left(p \to q\right) \wedge \left(p \to r\right) \equiv p \to \left(q \wedge r\right)$$

$$\left(p \to r\right) \wedge \left(q \to r\right) \equiv \left(p \vee q\right) \to r$$

$$\left(p \to q\right) \vee \left(p \to r\right) \equiv p \to \left(q \vee r\right)$$

$$\left(p \to r\right) \vee \left(q \to r\right) \equiv \left(p \wedge q\right) \to r$$

- Logical equivalences involving conditional statements

Important Logical Equivalences

$$p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$$

$$p \leftrightarrow q \equiv \overline{p} \leftrightarrow \overline{q}$$

$$p \leftrightarrow q \equiv (p \wedge q) \vee (\overline{p} \wedge \overline{q})$$

$$\overline{(p \leftrightarrow q)} \equiv p \leftrightarrow \overline{q}$$

- Logical equivalences involving biconditionals

Important Logical Equivalences

- Any proposition in any compound proposition can always be substituted by another compound proposition that is logically equivalent to it without changing the truth value of the original compound proposition.
- This property can be used to prove the logical equivalency of compound propositions instead of checking their truth tables