NYU, Tandon School of Engineering
CS Bridge — Spring 2022

Exam 1

Thursday, May 5, 2022

- This exam has 13 questions, with 100 points total.

- You have **two hours**.

- **You should submit your answers on the Gradescope platform** (not on NYU Brightspace).

- **It is your responsibility to take the time for the exam** (You may use a physical timer, or an online timer: https://vclock.com/set-timer-for-2-hours/).
  Make sure to upload the files with your answers to gradescope BEFORE the time is up, while still being monitored by ProctorU.
  We will not accept any late submissions.

- In total, you should upload 3 '.cpp' files:
  - One '.cpp' file for questions 1-11.
    Write your answer as one long comment (/* … */).
    Name this file 'YourNetID_q1to11.cpp'.
  - One '.cpp' file for question 12, containing your code.
    Name this file 'YourNetID_q12.cpp'.
  - One '.cpp' file for question 13, containing your code.
    Name this file 'YourNetID_q13.cpp'.

- **Write your name, and netID at the head of each file.**

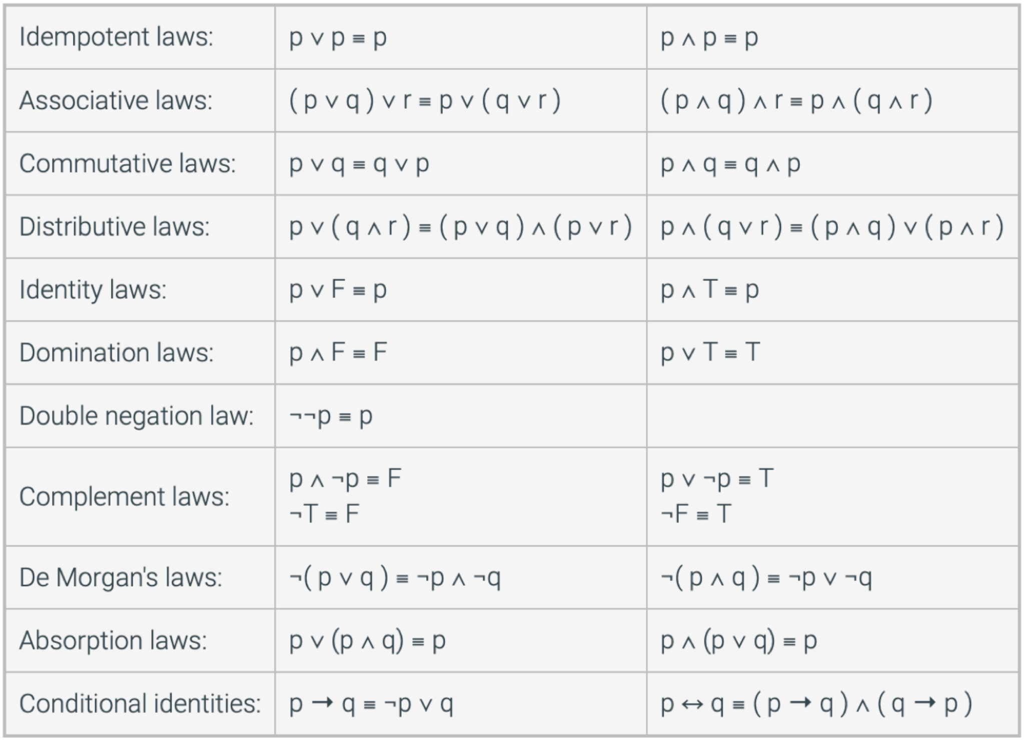
- This is a closed-book exam. However, you are allowed to use:
  - Visual Studio Code (VSCode) or Visual-Studio or Xcode. You should create a new project and work ONLY in it.
  - Two sheets of scratch paper.

  Besides that, no additional resources (of any form) are allowed.

- **You are not allowed to use C++ syntactic features that were not covered in the Bridge program so far.**

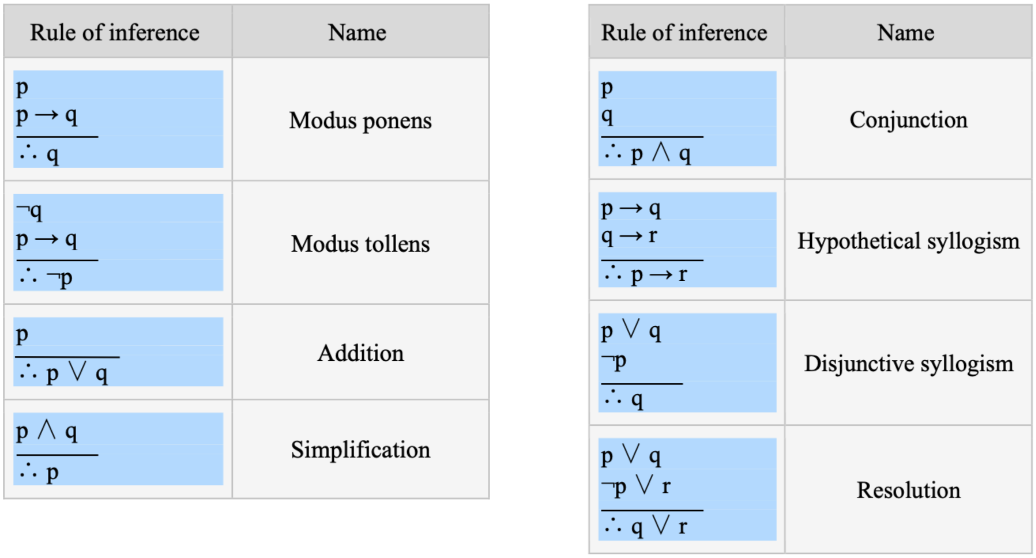
- Read every question completely before answering it.
  Note that there are 2 programming problems at the end**.**
  Be sure to allow enough time for these questions

Table 1.5.1: Laws of propositional logic.



Idempotent laws:	$p \lor p \equiv p$	$p \land p \equiv p$
Associative laws:	$(p \lor q) \lor r \equiv p \lor (q \lor r)$	$(p \land q) \land r \equiv p \land (q \land r)$
Commutative laws:	$p \lor q \equiv q \lor p$	$p \land q \equiv q \land p$
Distributive laws:	$p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$	$p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$
Identity laws:	$p \lor F \equiv p$	$p \land T \equiv p$
Domination laws:	$p \land F \equiv F$	$p \lor T \equiv T$
Double negation law:	$\neg\neg p \equiv p$	
Complement laws:	$p \land \neg p \equiv F$ $\neg T \equiv F$	$p \lor \neg p \equiv T$ $\neg F \equiv T$
De Morgan's laws:	$\neg(p \lor q) \equiv \neg p \land \neg q$	$\neg(p \land q) \equiv \neg p \lor \neg q$
Absorption laws:	$p \lor (p \land q) \equiv p$	$p \land (p \lor q) \equiv p$
Conditional identities:	$p \rightarrow q \equiv \neg p \lor q$	$p \leftrightarrow q \equiv (p \rightarrow q) \land (q \rightarrow p)$

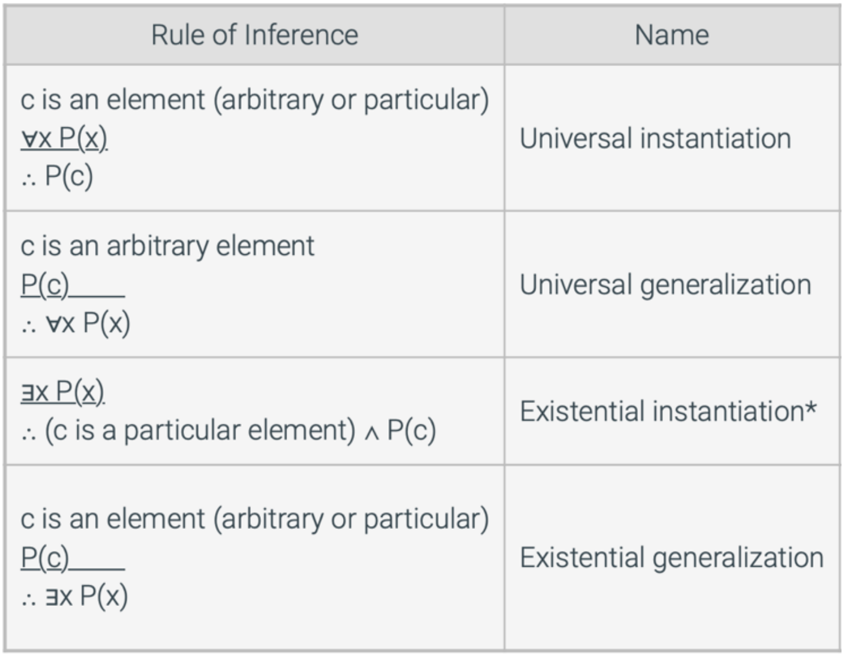
Table 1.12.1: Rules of inference known to be valid arguments.



Rule of inference	Name
p $p \rightarrow q$ $\therefore q$	Modus ponens
$\neg q$ $p \rightarrow q$ $\therefore \neg p$	Modus tollens
p $\therefore p \lor q$	Addition
$p \land q$ $\therefore p$	Simplification

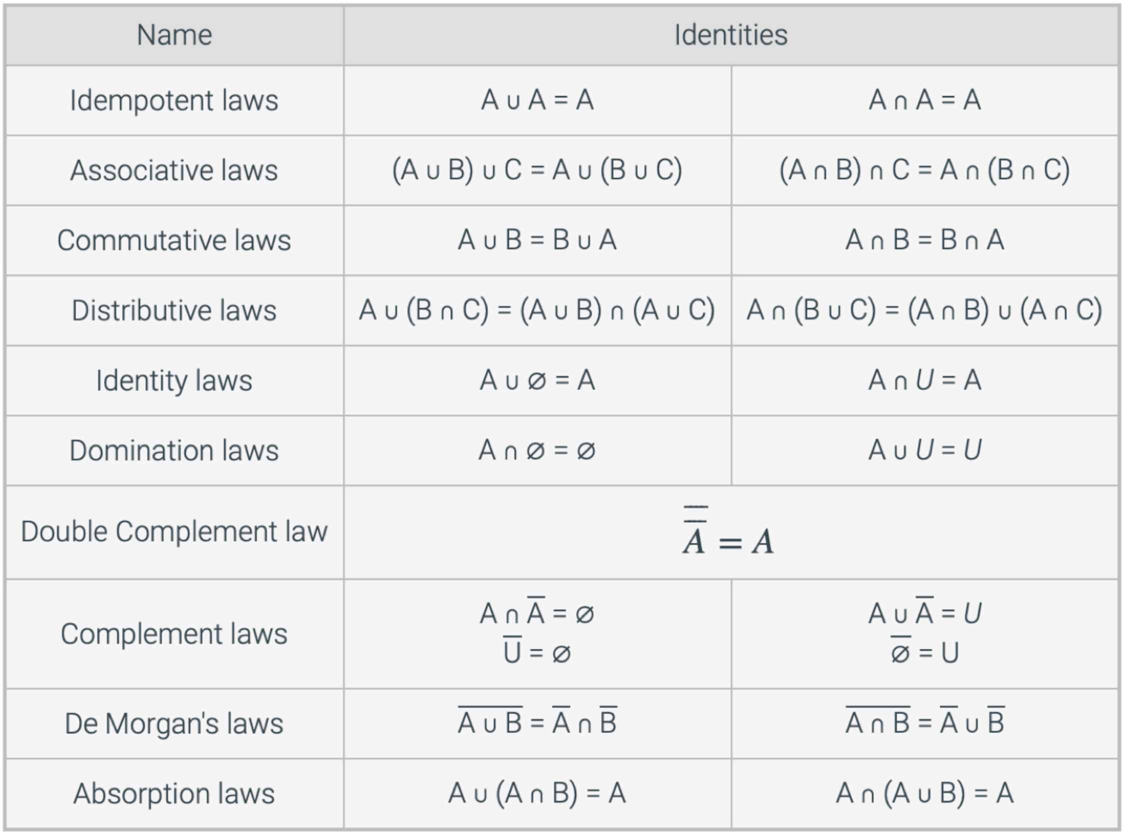
Rule of inference	Name
p q $\therefore p \land q$	Conjunction
$p \rightarrow q$ $q \rightarrow r$ $\therefore p \rightarrow r$	Hypothetical syllogism
$p \lor q$ $\neg p$ $\therefore q$	Disjunctive syllogism
$p \lor q$ $\neg p \lor r$ $\therefore q \lor r$	Resolution

Table 1.13.1: Rules of inference for quantified stateme



Rule of Inference	Name
c is an element (arbitrary or particular) ∀x P(x) ∴ P(c)	Universal instantiation
c is an arbitrary element P(c) ∴ ∀x P(x)	Universal generalization
∃x P(x) ∴ (c is a particular element) ∧ P(c)	Existential instantiation*
c is an element (arbitrary or particular) P(c) ∴ ∃x P(x)	Existential generalization

Table 3.6.1: Set identities.



Name	Identities	
Idempotent laws	A ∪ A = A	A ∩ A = A
Associative laws	(A ∪ B) ∪ C = A ∪ (B ∪ C)	(A ∩ B) ∩ C = A ∩ (B ∩ C)
Commutative laws	A ∪ B = B ∪ A	A ∩ B = B ∩ A
Distributive laws	A ∪ (B ∩ C) = (A ∪ B) ∩ (A ∪ C)	A ∩ (B ∪ C) = (A ∩ B) ∪ (A ∩ C)
Identity laws	A ∪ ∅ = A	A ∩ *U* = A
Domination laws	A ∩ ∅ = ∅	A ∪ *U* = *U*
Double Complement law	$\overline{\overline{A}} = A$	
Complement laws	$A \cap \overline{A} = \varnothing$ $\overline{U} = \varnothing$	$A \cup \overline{A} = U$ $\overline{\varnothing} = U$
De Morgan's laws	$\overline{A \cup B} = \overline{A} \cap \overline{B}$	$\overline{A \cap B} = \overline{A} \cup \overline{B}$
Absorption laws	A ∪ (A ∩ B) = A	A ∩ (A ∪ B) = A

Part I – Theoretical:

- ***You don't need to justify your answers to the questions in this part.***
- *For all questions in this part of the exam (questions 1-11), you should submit a* ***single*** *'.cpp' file. Write your answers as one long comment (`/* … */`).*
  Name this file 'YourNetID_q1to11.cpp'.

Question 1 (8 points)

a. Convert the decimal number $(2375)_{10}$ to its **base-5** representation.

b. Convert the 8-bits two's complement number $(11010011)_{\text{8-bit two's complement}}$ to its decimal representation.

Question 2 (4 points)

Select the propositions that are logically equivalent to $(\neg p \rightarrow \neg q)$.

a. $\neg p \vee q$

b. $\neg p \vee \neg q$

[✓ c. $p \vee \neg q$]

d. $\neg(p \wedge q)$

e. None of the above

Question 3 (5 points)

The domain of the variables x and y consists of all the students in our class. Define the predicates:

I(x): x has an Internet connection

C(x, y): x and y have chatted over the Internet.

Select the logical expression that is equivalent to:

"There is a student in your class who has chatted with everyone in your class over the Internet."

a. $\exists x \forall y\ (x \neq y\ \wedge\ C(x, y))$

b. $\forall x \exists y\ (x \neq y\ \rightarrow\ \neg C(x, y))$

c. $\forall x \exists y\ C(x, y)$

[✓ d. $\exists x \forall y\ C(x, y)$]

e. None of the above

Question 4 (5 points)
Show that at least four of any 22 days must fall on the same day of the week.

A proof by contradiction of the above statement starts by assuming which fact?

a. Suppose that at least three of the 22 days fall on the same day of the week.
b. Suppose that at most four of the 22 days fall on the same day of the week.
[✓ c. Suppose that at most three of the 22 days fall on the same day of the week.]
d. Suppose that exactly four of the 22 days fall on the same day of the week.

Question 5 (5 points)
Select the logical expression that is equivalent to: $\neg\forall y\exists x\exists z\,(P(x,y,z) \wedge \neg Q(x,y))$

a. $\exists y\exists x\,\neg\exists z\,(P(x,y,z) \vee Q(x,y))$
b. $\exists y\forall x\forall z\,(\neg P(x,y,z) \wedge Q(x,y))$
[✓ c. $\exists y\forall x\forall z\,(\neg P(x,y,z) \vee Q(x,y)$]
d. $\exists y\forall x\forall z\,(P(x,y,z) \wedge \neg Q(x,y))$
e. None of the above

Question 6 (4 points)
If A, B, and C are sets such that $(A \cap C) = (B \cap C)$ and $(A \cup C) = (B \cup C)$.
Select the statements that are **true.**

[✓ a. $A = B$]
b. $A \neq B$
c. $\mathrm{A} \subseteq B$
d. $\mathrm{B} \subseteq A$
e. None of the above

Question 7 (10 points)
$A = \{1, 2, 3, 4, \{2\}, \{4\}, \{1, 2,\ 3\}\}$.
For each of the following statements, state if they are true or false (no need to explain your choice).

a. $1 \in A$
b. $2 \subseteq A$
c. $\{1, 2, 4\} \in A$
d. $\{1, 2, 3\} \subseteq A$
e. $\{4\} \in A$
f. $\{4\} \subseteq A$
g. $\{1, 2, 3,\ \{2\}\} \in A$
h. $\{1, 2, 3,\ \{2\}\} \subseteq A$
i. $\emptyset \in A$
j. $\emptyset \subseteq P\,(A)$
k. $|\emptyset| = 1$
l. $|A| = 7$

Question 8 (4 points)

Select the set that is equivalent to $\overline{A} \cap (A \cup B)$.

a. $\emptyset$
b. A
c. $\overline{A} \cup B$
[✓ d. $\overline{A} \cap B$]
e. None of the above

Question 9 (5 points)

Let M be defined to be the set $\{a, b, c, d, f\}$.

Let f be a function: $f: \mathrm{P}(M) \rightarrow P(M)$, defined as follows:

for X ⊆ M, f(X) = M-X.

Select the correct description of the function f.

[✓ a. One-to-one and onto]
b. One-to-one but not onto
c. Not one-to-one but onto
d. Neither one-to-one nor onto

Question 10 (5 points)

The domain and target set of functions f and g are **Z**. The functions are defined as: f(x) = $2x^2 + 3$ and g(x) = $5x + 7$

An explicit formula for the function: f o g(x) will be

[✓ a. $50x^2 + 140x + 101$]
b. $50x^2 + 140X+98$
c. $10x^2 + 10$
d. $10x^2 + 22$
e. None of the above

Question 11 (5 points)

How many numbers must be selected from the set {1, 3, 5, 7, 9, 11, 13, 15} to guarantee that at least one pair of these numbers add up to 16?

Briefly explain your answers.

Part II – Coding:

- *For **each** question in this part (questions 12-13), you should submit a '.cpp' file, containing your code.*
- *Pay special attention to the style of your code. Indent your code correctly, choose meaningful names for your variables, define constants where needed, choose most suitable control statements, etc.*
- *In all questions, you may assume that the user enters inputs as they are asked.*
  For example, if the program expects a positive integer, you may assume that user will enter positive integers.
- *No need to document your code. However, you may add comments if you think they are needed for clarity.*

Question 12 (20 points)

Write a C++ program that reads a positive integer, n, and prints a shape of (2*n -1) lines consisting of asterisks (*) and spaces as follows:

1st line: print 1 asterisk, then print (2*n-2) spaces and then 1 asterisk
2nd line: print 2 asterisks, then (2*n-4) spaces and then 2 asterisks
3rd line: print 3 asterisks, then (2*n-6) spaces and then 3 asterisks
…
…
…
nth line: print n asterisks, then no/zero spaces and print n asterisks
(n+1)th line: print (n-1) asterisks, then 2 spaces and print (n-1) asterisks
(n+2)th line: print (n-2) asterisks, then 4 spaces and print (n-2) asterisks
…
…
…
(2*n-2)th line: print 2 asterisks, then (2*n-4) spaces and then 2 asterisks
(2*n-1)th line: print 1 asterisk, then print (2*n-2) spaces and then 1 asterisk

Your program should interact with the user **exactly** as demonstrated in the following four executions:

Execution example 1:

```
Please enter a positive integer:
5
*        *
**      **
***    ***
****  ****
**********
****  ****
***    ***
**      **
*        *
```

Execution example 2:

```
Please enter a positive integer:
8
*              *
**            **
***          ***
****        ****
*****      *****
******    ******
*******  *******
****************
*******  *******
******    ******
*****      *****
****        ****
***          ***
**            **
*              *
```

Execution example 3:

```
Please enter a positive integer:
3
*    *
**  **
******
**  **
*    *
```

Execution example 4:

```
Please enter a positive integer:
6
*          *
**        **
***      ***
****    ****
*****  *****
************
*****  *****
****    ****
***      ***
**        **
*          *
```

Question 13 (20 points)

A sequence of positive numbers has been given. Each of these positive numbers represents a person's age in years and will be in the range [1, 100]. Suppose we define different age groups as follows:

Age group A: People who are under 21 years old.
Age group B: People who are greater than or equal to 21 years old and less than or equal to 35 years old.
Age group C: People who are greater than or equal to 36 years old and less than or equal to 50 years old.
Age group D: People who are greater than 50 years old.

Write a C++ program that reads from the user a sequence of ages (positive numbers in between 1 and 100) and prints **the following statistics**.

Number of people in the age group A:
Number of people in the age group B:
Number of people in the age group C:
Number of people in the age group D:
Total number of people in the given sequence of ages:
Maximum age among the given sequence of ages:
Minimum age among the given sequence of ages:
Average age of the given sequence of ages:
Age group that has the highest number of people:

Implementation requirement:

a. **The user should enter their numbers, each one in a separate line, and type -1 to indicate the end of the input.**
b. You are not allowed to use C++ syntactic features that were not covered in the Bridge program so far.
c. You are not allowed to use any **cmath** or **math.h** library function for this program. You have to calculate maximum, minimum, and average ages without using any library function.
d. While printing average age, you can just print the raw value of average age. That is, you don't need to format it (like showing two digits after decimal point).

Your program should interact with the user **exactly** the same way, as demonstrated in the following two executions:

Execution example 1:

```
Please enter a sequence of numbers in between 1 and 100, each one in a
separate line. End your sequence by typing -1:
15
59
59
16
32
87
8
93
56
19
26
39
21
42
53
60
20
-1
Number of people in the age group A: 5
Number of people in the age group B: 3
Number of people in the age group C: 2
Number of people in the age group D: 7
Total number of people in the given sequence of ages: 17
Maximum age among the given sequence of ages: 93
Minimum age among the given sequence of ages: 8
Average age of the given sequence of ages: 41.4705882
Age group that has the highest number of people: Age group D
```

Execution example 2:

```
Please enter a sequence of numbers in between 1 and 100, each one in a
separate line. End your sequence by typing -1:
33
67
5
1
100
39
81
34
23
14
17
46
10
29
35
46
25
50
36
25
21
11
6
78
12
63
99
24
23
35
59
41
36
26
15
24
26
84
1
45
34
-1
Number of people in the age group A: 10
Number of people in the age group B: 15
Number of people in the age group C: 8
Number of people in the age group D: 8
Total number of people in the given sequence of ages: 41
Maximum age among the given sequence of ages: 100
Minimum age among the given sequence of ages: 1
Average age of the given sequence of ages: 36.0731707
Age group that has the highest number of people: Age group B
```