# Exam #1

## Thursday, August 12, 2021

- This exam has 12 questions, with 100 points total.

- You have **two hours**.

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

- **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 1-10.
    Write your answer as one long comment (/* … */).
    Name this file 'YourNetID_q1to10.cpp'.
  - One '.cpp' file for question 11, containing your code.
    Name this file 'YourNetID_q11.cpp'.
  - One '.cpp' file for question 12, containing your code.
    Name this file 'YourNetID_q12.cpp'.

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

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

  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: | $p \land \neg p \equiv F$<br>$\neg T \equiv F$ | $p \lor \neg p \equiv T$<br>$\neg F \equiv T$ |
| 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: | $p \to q \equiv \neg p \lor q$ | $p \leftrightarrow q \equiv (p \to q) \land (q \to p)$ |

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

| Rule of inference | Name |
|---|---|
| $p$<br>$p \to q$<br>$\therefore q$ | Modus ponens |
| $\neg q$<br>$p \to q$<br>$\therefore \neg p$ | Modus tollens |
| $p$<br>$\therefore p \lor q$ | Addition |
| $p \land q$<br>$\therefore p$ | Simplification |

| Rule of inference | Name |
|---|---|
| $p$<br>$q$<br>$\therefore p \land q$ | Conjunction |
| $p \to q$<br>$q \to r$<br>$\therefore p \to r$ | Hypothetical syllogism |
| $p \lor q$<br>$\neg p$<br>$\therefore q$ | Disjunctive syllogism |
| $p \lor q$<br>$\neg p \lor r$<br>$\therefore q \lor r$ | Resolution |

Table 1.13.1: Rules of inference for quantified stateme

| Rule of Inference | Name |
|---|---|
| c is an element (arbitrary or particular)<br>$\underline{\forall x\, P(x)}$<br>$\therefore P(c)$ | Universal instantiation |
| c is an arbitrary element<br>$\underline{P(c)\qquad}$<br>$\therefore \forall x\, P(x)$ | Universal generalization |
| $\underline{\exists x\, P(x)}$<br>$\therefore$ (c is a particular element) $\wedge$ $P(c)$ | Existential instantiation* |
| c is an element (arbitrary or particular)<br>$\underline{P(c)\qquad}$<br>$\therefore \exists x\, P(x)$ | Existential generalization |

Table 3.6.1: Set identities.

| Name | Identities | |
|---|---|---|
| Idempotent laws | $A \cup A = A$ | $A \cap A = A$ |
| Associative laws | $(A \cup B) \cup C = A \cup (B \cup C)$ | $(A \cap B) \cap C = A \cap (B \cap C)$ |
| Commutative laws | $A \cup B = B \cup A$ | $A \cap B = B \cap A$ |
| Distributive laws | $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ | $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ |
| Identity laws | $A \cup \varnothing = A$ | $A \cap U = A$ |
| Domination laws | $A \cap \varnothing = \varnothing$ | $A \cup U = U$ |
| Double Complement law | $\overline{\overline{A}} = A$ | |
| Complement laws | $A \cap \overline{A} = \varnothing$<br>$\overline{U} = \varnothing$ | $A \cup \overline{A} = U$<br>$\overline{\varnothing} = U$ |
| De Morgan's laws | $\overline{A \cup B} = \overline{A} \cap \overline{B}$ | $\overline{A \cap B} = \overline{A} \cup \overline{B}$ |
| Absorption laws | $A \cup (A \cap B) = A$ | $A \cap (A \cup B) = A$ |

## ***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 1-10), you should submit a **single** '.cpp' file. Write your answers as one long comment (`/* … */`).*
  *Name this file 'YourNetID_q1to10.cpp'.*

**Question 1 (5 points)**

a. Convert the decimal number $(126)_{10}$ to its **base-4** representation.
b. Convert the 8-bits two's complement number $(11110100)_{\text{8-bit two's complement}}$ to its decimal representation.

**Question 2 (5 points)**

Select the proposition that is logically equivalent to $(\neg p \rightarrow \neg q)$.

a. $p \vee q$
b. $\neg p \vee q$
c. $p \vee \neg q$
d. $\neg p \vee \neg q$
e. None of the above

**Question 3 (5 points)**

The domain for x and y is the set of real numbers. Select the statement that is **false**.

a. $\forall x\ \exists y\ (xy \geq 0)$
b. $\forall x\ \exists y\ (x + y \geq 0)$
c. $\exists x\ \forall y\ (xy \geq 0)$
d. $\exists x\ \forall y\ (x + y \geq 0)$

**Question 4 (5 points)**

The domain of discourse are the students in a class. Define the predicates:
S(x): x studied for the test
A(x): x received an A on the test

Select the logical expression that is equivalent to:
"Everyone who received an A on the test, studied for it."

a. $\forall x\ (A(x) \rightarrow S(x))$
b. $\forall x\ (S(x) \leftrightarrow A(x))$
c. $\forall x\ (S(x) \rightarrow A(x))$
d. $\forall x\ (S(x) \wedge A(x))$

**Question 5 (5 points)**

**Theorem:** A group of 4 kids have a total of 13 chocolate bars. Then at least one of the kids has at least four chocolate bars.

A proof by contradiction of the theorem starts by assuming which fact?

a. All the kids have four or fewer chocolate bars.
b. All the kids have less than four chocolate bars.
c. There is a kid with four or fewer chocolate bars.
d. There is a kid with less than four chocolate bars.

**Question 6 (5 points)**

Select the logical expression that is equivalent to: $\neg\forall x[(\exists y P(x,y)) \rightarrow (\exists y Q(x,y))]$

a. $\exists x(\forall y P(x,y)) \wedge (\forall y Q(x,y))$
b. $\exists x(\exists y P(x,y)) \wedge (\forall y \neg Q(x,y))$
c. $\exists x(\exists y P(x,y)) \wedge (\forall y Q(x,y))$
d. None of the above

**Question 7 (5 points)**

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

a. $\emptyset$
b. $C$
c. $B$
d. $C \cup B$

**Question 8 (10 points)**

$A = \{1, 2, 3, \{1\}, \{2\}, \{3, 4\}\}$.

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. $\{1, 2\} \in A$
d. $\{1, 2\} \subseteq A$
e. $\{3\} \in A$
f. $\{3\} \subseteq A$
g. $\{3, 4\} \in A$
h. $\{3, 4\} \subseteq A$
i. $\emptyset \in A$
j. $\emptyset \subseteq A$

**Question 9 (5 points)**

Let $f$ be a function: $f\colon \mathrm{P}(\{1,2,3\}) \to \{0,1\} \times P(\{2,3\})$, defined as follows:

For every $A \in P(\{1,2,3\})$, $f(A) = \begin{cases} (0, A) & 1 \notin A \\ (1, A - \{1\}) & 1 \in A \end{cases}$

For example, $f(\{1,3\}) = (0, \{3\})$, since $1 \in \{1,3\}$ second case is applied.

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 function $f\colon P(\{1,2,3\}) \to P(\{1,2,3,4\})$ is defined as:

For every $A \in P(\{1,2,3\})$, $f(A) = \{1,2,3,4\} - A$

For example, $f(\{2,3\}) = \{1,4\}$

Find the range of $f$.

# ***Part II – Coding:***

- *For **each** question in this part (questions 11-12), 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 11 (20 points)**

Write a program that reads a positive integer, $n$, and prints a nxn square chess board shape. That is a nxn square with an alternating pattern of squares in two colors.
In this question, instead of two colors, we will use two symbols 'X' and 'O'.

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

**Execution example 1:**

```
Please enter a positive integer:
3
XOX
OXO
XOX
```

**Execution example 2:**

```
Please enter a positive integer:
8
XOXOXOXO
OXOXOXOX
XOXOXOXO
OXOXOXOX
XOXOXOXO
OXOXOXOX
XOXOXOXO
OXOXOXOX
```

**<u>Question 12 (25 points)</u>**

Consider the following definition:

<u>Definition</u>: A positive integer is called a ***more-odd number*** if it has more odd digits than even ones (in the decimal representation of the number).

For example, 310 is a *more-odd number*, while 4123 is not.

Write a program that reads from the user a sequence of positive integers and prints the **largest** *more-odd* number that was entered, or a message saying: "There were no more-odd numbers in the input".

<u>Implementation requirement</u>: The user should enter their numbers, each one in a separate line, and type -1 to indicate the end of the input.

Your program should interact with the user **exactly** the same way, as demonstrated bellow:

```
Please enter a sequence of positive integers, each one in a separate line.
End your sequence by typing -1:
3203
31
21168
5466
758
-1
The largest more-odd number that was entered is 758
```