SEMtech!
Student Engagement & Mentoring in Technology

# Study Guide: Chapter 28 – Logic Components (CompTIA ITF+

## Learning Objectives

By the end of this chapter, you should be able to:

- Understand the function and purpose of logic components in computing.
- Identify basic logic operations (AND, OR, NOT, XOR, etc.).
- Apply logic operations in computing scenarios.
- Recognize how logic is used in programming, circuits, and decision-making processes.

---

## 1. Introduction to Logic Components

- Logic components are fundamental to computing.
- These components evaluate *conditions* and make *decisions* based on *true/false (Boolean)* logic.
- Boolean logic uses values: **True (1)** and **False (0)**.

---

## 2. Basic Logic Gates

These gates form the foundation of both software (if/else logic) and hardware (circuit design).

| Gate Symbol | Function | Truth Table |
|---|---|---|
| AND | True if *both* inputs are true | $1 \wedge 1 = 1$; else 0 |
| OR | True if *at least one* input is true | $0 \vee 1 = 1$; $1 \vee 0 = 1$; $1 \vee 1 = 1$; $0 \vee 0 = 0$ |
| NOT | Inverts the input | $\neg 1 = 0$, $\neg 0 = 1$ |
| XOR | True if *only one* input is true | $1 \oplus 0 = 1$; $0 \oplus 1 = 1$; else 0 |

---

SEMtech!
Student Engagement & Mentoring in Technology

## 3. Logic in Programming

- Used in **conditional statements**, loops, and decision-making.
- Example:

```
python
CopyEdit
if is_user_logged_in and has_permission:
    print("Access granted")
```

- **AND**: Both conditions must be true.
- **OR**: Either condition can be true.
- **NOT**: Reverses a condition.

---

## 4. Real-World Examples

- **Security systems**: If door is closed AND code is correct → unlock door.
- **Login validation**: If username exists AND password is correct → allow access.
- **Error checking**: If NOT valid_input → display error.

---

## 5. Logic Circuits (Introductory)

- Hardware uses physical gates to create logic circuits.
- These are foundational to CPUs and memory operations.
- Logic gates can be combined to build **adders**, **multiplexers**, and **ALUs (Arithmetic Logic Units)**.

---

## Key Terms to Know

- **Boolean logic**
- **Conditional statement**
- **Truth table**
- **Gate**
- **Decision-making**
- **Control flow**
- **Logical operator (AND, OR, NOT, XOR)**

SEMtech!
Student Engagement & Mentoring in Technology

---

## Practice Questions

1. What does the OR gate output when both inputs are false?
2. How is the XOR gate different from the OR gate?
3. What is the output of: NOT (True AND False)?
4. Write a simple program condition using AND and OR.

---

## Tips for the Exam

- Focus on understanding **truth tables**.
- Be able to **simulate simple logic circuits** in your head or on paper.
- Practice **writing logical expressions** in pseudo-code.
- Memorize how different gates behave with all possible inputs.

SEMtech!
Student Engagement & Mentoring in Technology

# Expanded Section: Control Structures — Branching and Looping

## Branching (Decision-Making Logic)

**Definition:**
Branching allows a program to make decisions and follow different paths based on conditions.

**Common Structures:**

- `if` statement
  Executes a block if a condition is true.

```
python
CopyEdit
if user_is_admin:
    print("Access granted")
```

- `if-else` statement
  Chooses between two paths.

```
python
CopyEdit
if password_is_correct:
    print("Welcome!")
else:
    print("Access denied.")
```

- `if-elif-else` (Python) or `switch` (in other languages)
  Selects among multiple conditions.

```
python
CopyEdit
if grade >= 90:
    print("A")
elif grade >= 80:
    print("B")
else:
    print("C or lower")
```

**Key Concept:** Branching uses **Boolean logic** to evaluate conditions. These conditions are typically formed using comparison (`==`, `>`, `<`) and logical (`and`, `or`, `not`) operators.

---

SEMtech!
Student Engagement & Mentoring in Technology

## Looping (Iteration Logic)

**Definition:**
Looping repeats a set of instructions while a condition is true or for a predetermined number of times.

**Types of Loops:**

1. **`while` loop** – Repeats as long as a condition is true.

```
python
CopyEdit
while counter < 5:
    print(counter)
    counter += 1
```

2. **`for` loop** – Repeats a fixed number of times, often used with ranges or collections.

```
python
CopyEdit
for i in range(5):
    print(i)
```

3. **`do-while` loop** (in some languages like C, Java) – Executes at least once before checking the condition.

**Real-World Example:**

- **Login attempts:** Allow up to 3 tries to enter a password correctly using a `while` loop.
- **Menu navigation:** Loop until the user chooses to exit.

---

## Quick Comparison: Branching vs. Looping

| Feature | Branching | Looping |
|---|---|---|
| Purpose | Makes a decision | Repeats a task |
| Structure | if, else if, else, switch | while, for, do-while |
| Logic Used | Boolean conditions | Boolean conditions or counters |
| Example | if user is admin → access | while attempts < 3 → retry login |

SEMtech!
Student Engagement & Mentoring in Technology

## Practice Prompts

1. Write a simple program that prints “Pass” if the score is 70 or more, else prints “Fail”.
2. Write a loop that prints the numbers 1 to 10.
3. Create a login loop that stops after 3 failed attempts using a `while` loop.

## Practice Prompt 1

**Q:** Write a simple program that prints “Pass” if the score is 70 or more, else prints “Fail”.

**Answer (Python Example):**

```
python
CopyEdit
score = int(input("Enter your score: "))

if score >= 70:
    print("Pass")
else:
    print("Fail")
```

---

## Practice Prompt 2

**Q:** Write a loop that prints the numbers 1 to 10.

**Answer (Python Example):**

```
python
CopyEdit
for i in range(1, 11):
    print(i)
```

`range(1, 11)` starts at 1 and goes up to (but not including) 11.

---

SEMtech!
Student Engagement & Mentoring in Technology

## Practice Prompt 3

**Q:** Create a login loop that stops after 3 failed attempts using a `while` loop.

### ✅ Answer (Python Example):

```
python
CopyEdit
correct_password = "secure123"
attempts = 0
max_attempts = 3

while attempts < max_attempts:
    password = input("Enter your password: ")
    if password == correct_password:
        print("Login successful.")
        break
    else:
        attempts += 1
        print("Incorrect password.")

if attempts == max_attempts:
    print("Account locked. Too many failed attempts.")
```

---

These examples are designed to illustrate **basic logic operations** involving **conditions and loops**, in alignment with CompTIA ITF+ objectives.