

Computational Thinking Induction Booklet

Unit 18: Learning Aim B
Understand Mathematical Methods for Functions and Boolean
Operations

BTEC Level 2 – Information & Creative Technology
Pearson

Student Name: _____

Course: BTEC IT Level 2

Unit: Unit 18 – Computational Thinking

Teacher: _____

Submission Date: _____

Developing problem-solving skills through computational thinking

Table of Contents
1. Introduction What This Guide Covers Importance of Functions and Boolean Logic in IT
2. Explaining Computational Uses of Functions (18/2B.P2) What Is a Function? Understanding Linear and Quadratic Functions Real-World Applications in Computing Obtaining the Equation of a Straight Line from a Graph Worked Example: Finding the Gradient and Y-Intercept
3. Explaining Computational Uses of Boolean Logic (18/2B.P3) What Is Boolean Logic? Using AND, OR, and NOT Operators Truth Tables and Logic Gates for Three Operations Practical Examples from Programming and Circuits
4. Representing Domain and Range Using Software (18/2B.M2) Visualising Linear and Quadratic Functions in Excel How Software Supports Function Design in Computing
5. Using Complex Boolean Logic in Series (18/2B.M3) Combining Multiple Boolean Operations Truth Tables for Logic in Series Logic Gate Circuits for Real-World Scenarios
6. Using Functions to Calculate Position and Plot Shapes (18/2B. D2) Functions for Positioning Objects on Screen Linear and Quadratic Applications in Motion Plotting and Drawing Shapes with Functions Examples from Game and App Design

7. Solving Problems Using Logic Gates (18/2B.D3)

Creating a Problem Scenario

Logic Gate Solution with Three or More Operations

Truth Table and Gate Diagram

Explanation of Results

8. Conclusion

What We Learned About Functions and Logic Why

These Skills Matter in Digital Systems

9. Practice Questions & Solutions

10. Glossary of Key Terms

Definitions of Important Computational Concepts

11. Conclusion & Summary

Key Takeaways from the Booklet

The Importance of Computational Thinking in IT Careers

12. References

1.Introduction

What this guide covers

This guide is created for beginner learners and students who are starting to understand computational thinking, functions, and Boolean logic. The main aim of this guide is to explain how problems in computing can be solved using simple mathematical methods and logical thinking. In everyday life, when we use phones, computers, or apps, many processes are happening in the background. These processes are controlled by functions and Boolean logic.

In the first part of this guide, I explain functions. A function is a rule that takes an input and gives an output. This is very important in computing because computers always work with inputs and outputs. For example, when you type something in a search bar, the computer takes your input and gives results as output. This can be shown using a function.

I also explain different types of functions, such as linear functions and quadratic functions. Linear functions produce straight line graphs, and they are used in many simple calculations. Quadratic functions produce curved graphs and are used in more advanced situations. These functions help us understand patterns and relationships between values.

Another important part is learning how to find the equation of a straight line from a graph. This means we can look at a graph and understand how the values are connected. This is useful in computing because many systems use graphs to represent data. For example, companies use graphs to show profit, growth, and performance.

In this guide, I also explain how software tools like Microsoft Excel and Desmos can be used to create graphs. These tools help students to visualise functions easily. Instead of doing everything by hand, software can calculate values and draw graphs quickly. This saves time and helps students understand concepts better. I also explain domain and range. Domain means all possible input values, and range means all possible output values. These are important when working with functions because they show the limits of the system.

In the second part of this guide, I explain Boolean logic. Boolean logic is used in computing to make decisions. It works with two values: true and false. Computers use this system to decide what action to take. For example, when you log into a website, the system checks if your password is correct. If it is true, you can enter. If it is false, access is denied.

I also explain logic gates such as AND, OR, and NOT. These gates are used in both software and hardware. They take inputs and give outputs based on logical conditions. For example, an AND gate will only give true output when both inputs are true. These gates are very important in building circuits and digital systems.

Truth tables are also included in this guide. Truth tables show all possible input combinations and their outputs. This helps students understand how logic gates work step by step. I also give real-life examples like alarm systems, smart homes, and games to show how Boolean logic is used in everyday technology.

Overall, this guide helps students understand how mathematics and logic are connected to computing. It shows how problems can be broken into smaller steps and solved in a structured way. This is called computational thinking, and it is a very important skill in IT.

Important of functions Boolean logic in IT

Functions and Boolean logic are very important in modern information technology. They are used in almost every system, including software, hardware, and networks. Without these concepts, computers would not be able to process data or make decisions.

Functions are used to describe relationships between values. In computing, functions are used in programming to perform calculations and process data. For example, when a program calculates the total price of items in an online store, it is using a function. The inputs are the prices of items, and the output is the total cost. Functions help to organise code and make programs easier to understand.

Functions are also used in graphs and data analysis. Companies use functions to predict future trends. For example, a company may use a function to estimate future sales based on past data. This helps in decision-making and planning.

Boolean logic is used to control decisions in computing systems. It works with true or false conditions. This is very important because computers need to make decisions all the time. For example, in a game, the system checks if the player presses a button. If true, the character jumps. If false, nothing happens. This shows how Boolean logic controls actions.

Boolean logic is also used in login systems. When you enter your username and password, the system checks if both are correct. This is an example of an AND condition. If both inputs are true, access is given. If one is false, access is denied.

In hardware, Boolean logic is used in circuits. Logic gates are used inside microchips to perform operations. These gates control how signals move inside the computer. For example, in a smart home system, sensors can detect motion and turn lights on or off using logic gates.

Boolean logic is also used in safety systems. For example, a security system may only open a door if a card is scanned AND a correct code is entered. This ensures safety and prevents unauthorised access.

Both functions and Boolean logic are important for developing problem-solving skills. They help students think in a logical and structured way. Instead of guessing, students learn to follow steps and find correct solutions. This is very useful in programming, engineering, and data analysis.

These skills are also useful for future careers. Many IT jobs require knowledge of functions and logic. For example, software developers, data analysts, and engineers all use these concepts in their work. Learning these topics helps students prepare for higher education and careers in technology.

In conclusion, functions and Boolean logic are essential parts of computational thinking. They help us understand how computers work and how problems can be solved using maths and logic. By learning these concepts, students can improve their understanding of digital systems and develop important skills for the future.

Identify Linear and Quadratic Functions

In computational thinking and programming, mathematical functions are very important. They help computers perform calculations, control movements, and manage changes in values. For example, when a game character moves, when an animation plays, or when values change over time, functions are used behind the scenes.

Two common types of functions used in computing are linear functions and quadratic functions. These functions help us understand how values are connected and how systems behave.

A linear function shows a straight-line relationship between two variables. It means the value changes at a constant rate.

The formula of a linear function is:

$$y = mx + c$$

Explanation of Each Part

- **y (output / dependent variable)**
This is the result of the function. It depends on the value of x. In computing, this can be something like position, speed, or value on screen.
- **x (input / independent variable)**
This is the value we choose or control. It can change freely. In computing, x can represent time, user input, or movement.
- **m (gradient / slope)**
This shows how fast the value changes. If m is large, the change is fast. If m is small, the change is slow.
In computing, this can control speed of movement or rate of change.
- **c (starting value / intercept)**
This is the value of y when $x = 0$. It shows the starting point of the system.

Example

$$y = 2x + 1$$

$$\text{If } x = 1 \rightarrow y = 3$$

$$\text{If } x = 2 \rightarrow y = 5$$

This shows a straight-line increase.

Computational Use of Linear Functions

Linear functions are used a lot in programming because many systems change at a constant rate.

Example 1: Movement in a Game

If a character moves at a constant speed:

$$x = 5t$$

Where:

t = time

x = position

This means the character moves 5 pixels every second.

Example 2: Temperature Change

$$y = 3x + 10$$

This could represent temperature increasing over time.

Example 3: Salary Calculation

$$y = 10x$$

Where x = hours worked

y = total salary

Linear functions are useful because they are simple and easy to predict.

Quadratic Functions

A quadratic function shows a curved relationship between variables. The graph is not straight; it is a curve called a parabola.

The formula is:

$$y = ax^2 + bx + c$$

Explanation

- x^2 means the value is squared
- This creates a curve instead of a straight line
- The value changes faster as x increases

Example

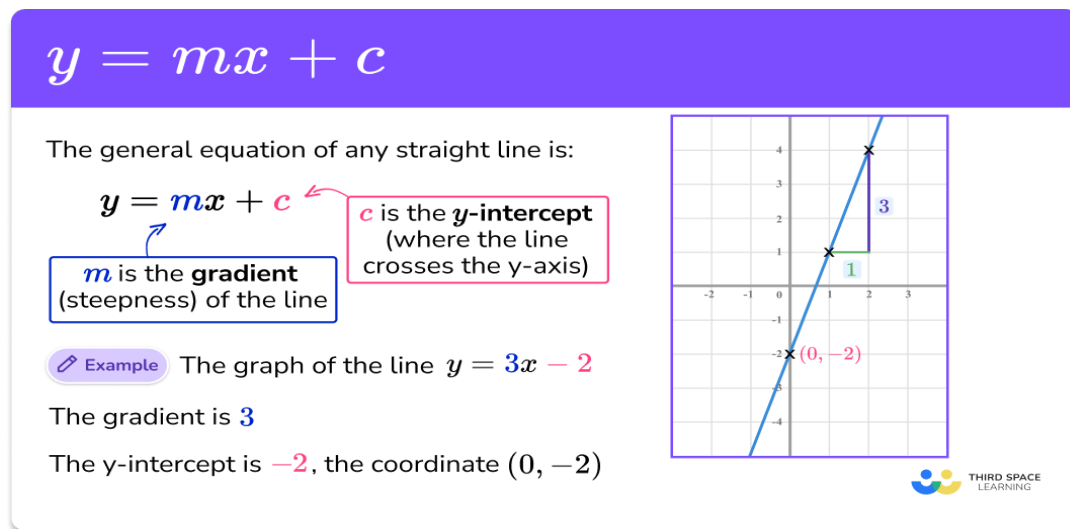
$$y = x^2$$

$$\text{If } x = 1 \rightarrow y = 1$$

$$\text{If } x = 2 \rightarrow y = 4$$

$$\text{If } x = 3 \rightarrow y = 9$$

You can see the value increases faster and faster.



Computational Use of Quadratic Functions

Quadratic functions are used when values do not change at a constant rate

Example 1: Jumping in a Game

When a character jumps, it goes up and then comes down.

This motion can be shown using a quadratic function.

Example 2: Projectile Motion

In games or simulations, objects like balls or bullets follow curved paths.

Example 3: Physics Calculations

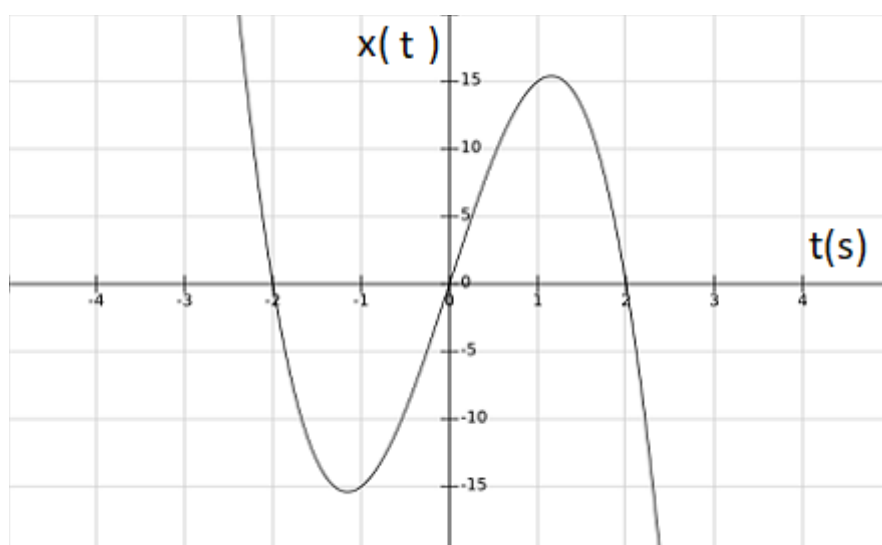
Used in motion, gravity, and acceleration.

Feature	Linear functions	Quadratic function
Shape	Straight line	Curve (parabola)

Formula	$y = mx + c$	$y = ax^2 + bx + c$
Change	Constant	Increasing/Changing
use	Simple system	Complex motion

Linear and quadratic functions are very important in computing. Linear functions are used when values change at a constant rate, such as movement or simple calculations. Quadratic functions are used when values change in a curved way, such as jumping or motion.

Understanding these functions helps students learn how computers control movement, calculate values, and solve problems. This is an important part of computational thinking and is used in many real-life applications like games, simulations, and software systems.



Quadratic Functions

A quadratic function shows a curved relationship between two variables. The graph of a quadratic function is not a straight line. Instead, it forms a U-shaped curve, which is called a parabola.

The general form of a quadratic function is:

$$y = ax^2 + bx + c$$

Explanation of Each Part

- **y (output / dependent variable)**

This is the result of the function. It changes depending on the value of x .

- **x (input / independent variable)**

This is the value we choose or control. The output depends on this value.

- **a (controls shape of graph)**

This value decides how wide or narrow the curve is.

If **a is positive**, the graph opens upwards (U-shape).

If **a is negative**, the graph opens downwards (inverted U).

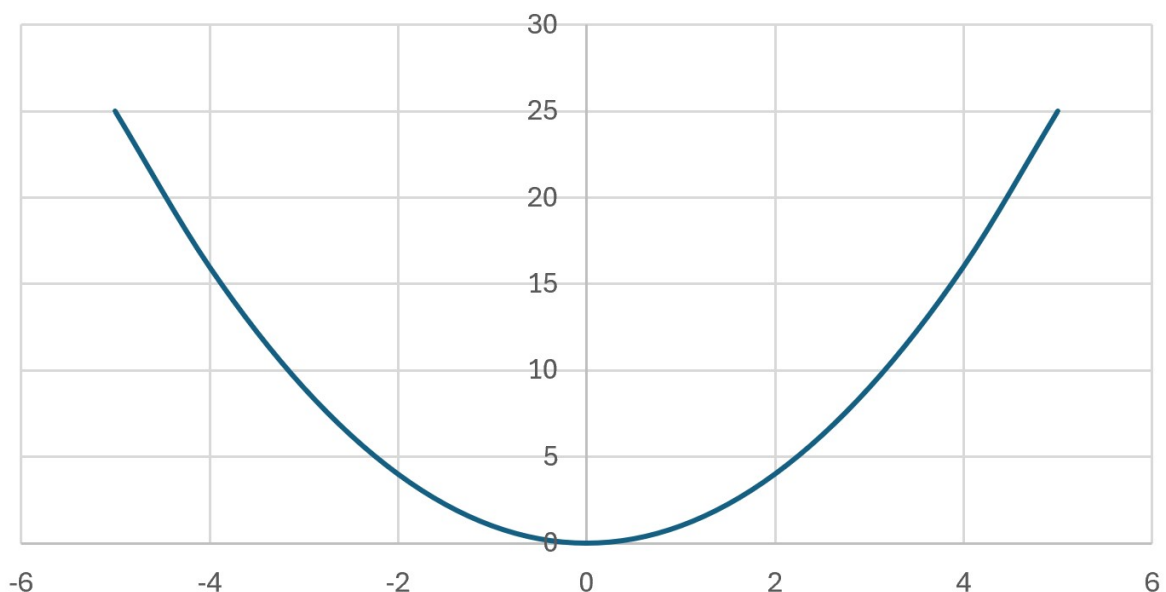
- **b (controls position)**

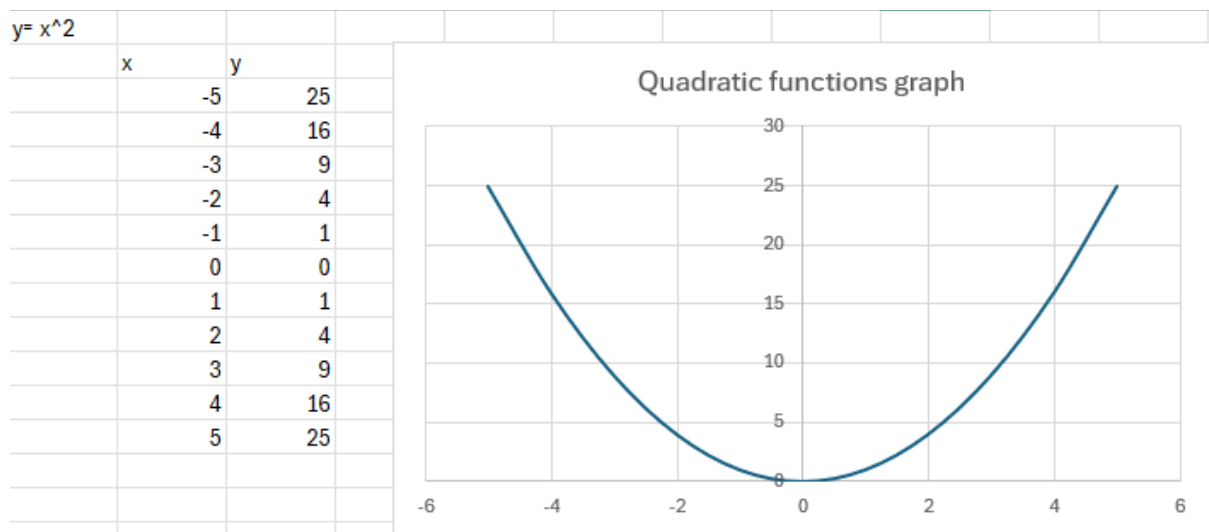
This affects the direction and position of the curve.

- **c (starting value / intercept)**

This is where the graph crosses the y -axis (when $x = 0$).

Quadratic functions graph





Example

$$y = x^2$$

If we put values:

X	Y
-2	4
-1	1
0	0
1	1
2	4

This creates a U-shaped graph

Input and Output Values

- x (input / independent variable)

This is the value we choose. It can be changed freely. In computing, x can represent time, position, or user input.

- y (output / dependent variable)

This is the result of the function. It depends on the value of x . When x changes, y also changes.

Because of the squared value (x^2), the output does not change in a straight line. Instead, it creates a curve called a parabola.

This curve can look like:

- U shape (opening up)
- upside-down U (opening down)

Standard Form

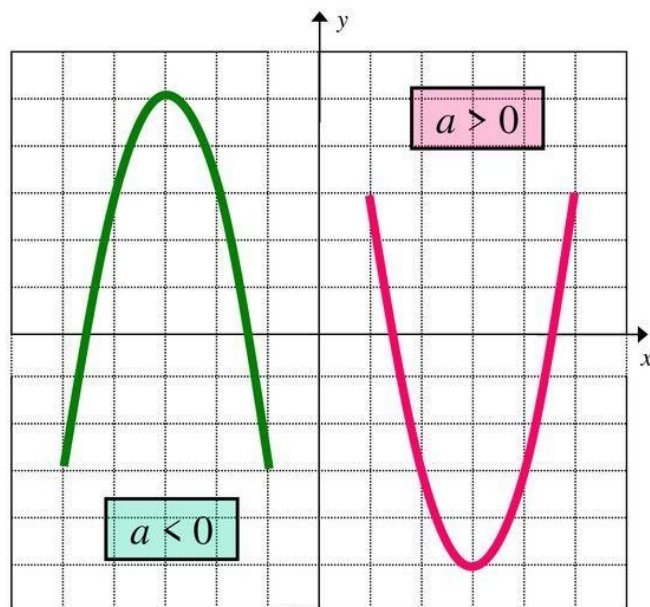
The standard form of a quadratic function is:

$$y = ax^2 + bx + c$$

The parabola will open up when the a value is positive.

The parabola will open down when the a value is negative.

$$a \neq 0$$



Computational Use

Quadratic functions are very useful in computing, especially when values change in a curved way.

Example: Game Jump

When a character jumps in a game

- It moves up
- Slows down
- Stops
- Then comes back down

This motion can be shown using a quadratic function.

Example:

$$y = -5t^2 + 20t$$

Where:

t = time

y = height

At first, the character goes up

Then it slows down

Then it comes down

This creates a parabolic shape

Other Uses

Quadratic functions are also used in:

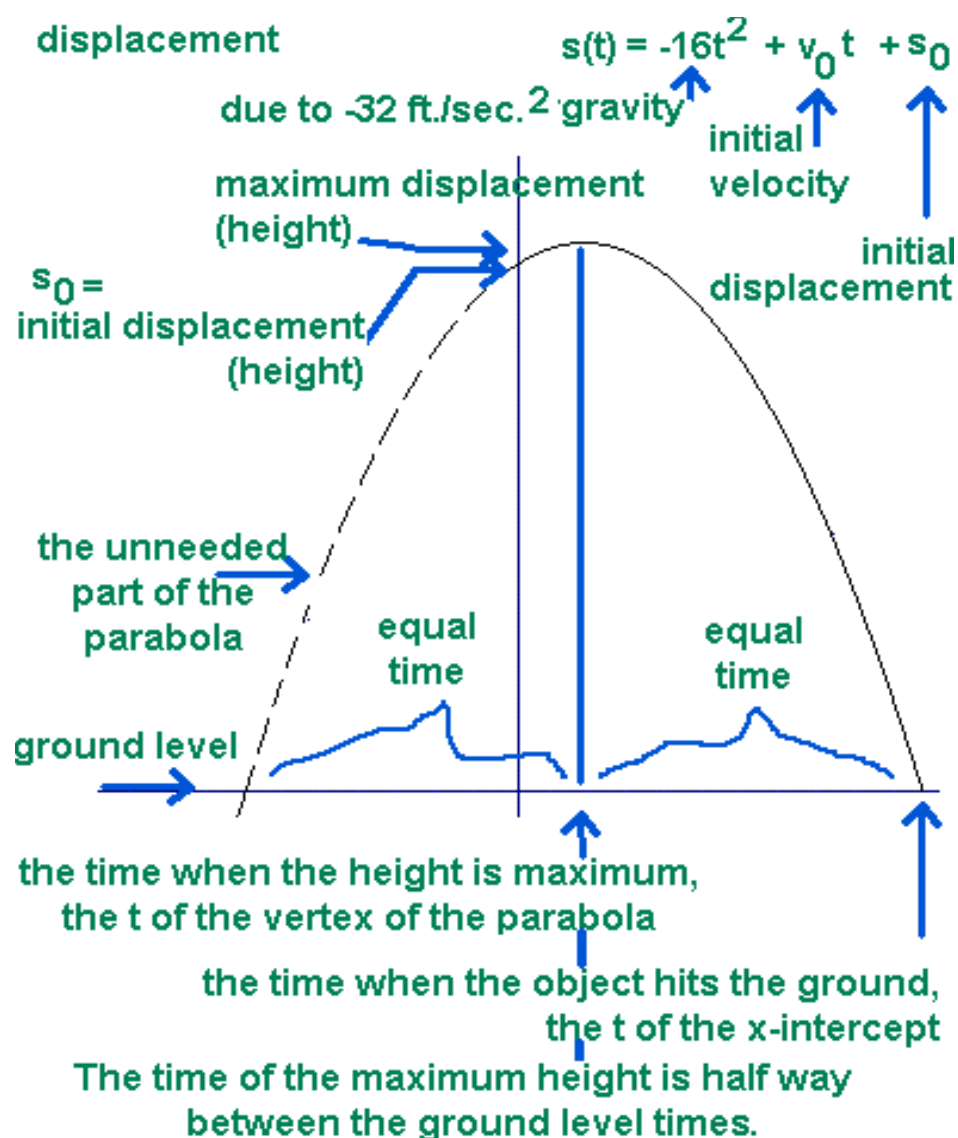
- Game physics (movement and jumping)
- Animations (smooth motion)
- Simulations (real-world models)
- Graphics (curved shapes and paths)

Simple Understanding

Linear function → straight line

Quadratic function → curved line

Quadratic functions are important because they help represent real-life movement and changes that are not constant. They are widely used in games, simulations, and animations. Understanding these functions helps students learn how computers model real-world behavior using mathematics.



Identifying the Computational Use of Functions and Finding the Equation of a Straight Line from a Graph

Computational Use of Functions

In computing, functions are very important because they help show how values change in a system. Functions are used to represent relationships between inputs and outputs. This makes it easier to control actions and predict results.

Functions are used in many areas such as

- Animations
- Games
- Calculating values over time
- Moving objects on screen
- Drawing shapes and graphs

For example, in a game, a function can control how a character moves across the screen. If the movement is constant, a linear function can be used to show the position of the character at different times.

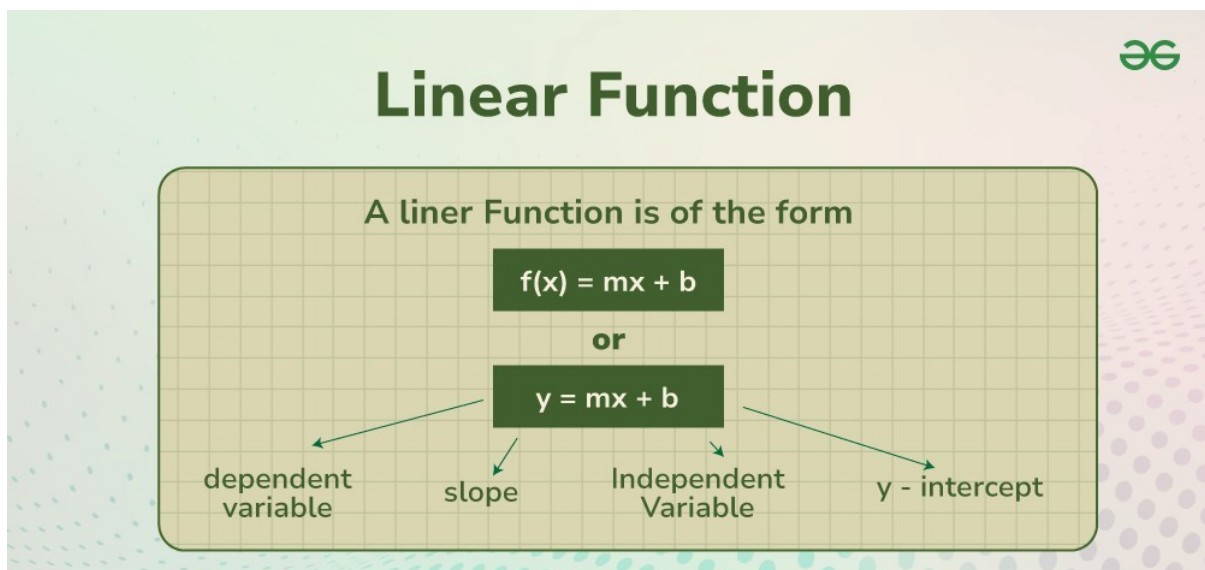
Functions help programmers organise code and make systems easier to understand. Instead of writing everything again and again, functions allow us to reuse logic.

What Is a Linear Function in Computing?

A linear function shows a straight-line relationship between two variables. This means the value changes at a constant rate.

The formula of a linear function is

$$y = mx + b$$



Meaning of Each Part

- x (independent variable / input):
This is the value we control. It can be time, position, or user input.
- y (dependent variable / output):
This is the result. It depends on x .
- m (gradient / slope):
This shows how fast the value changes. A higher value means faster change.
- c (y-intercept):
This is the starting point when $x = 0$.

Example in Computing

In a 2D game, if a character moves at a constant speed:

$$x = 4t$$

Where:

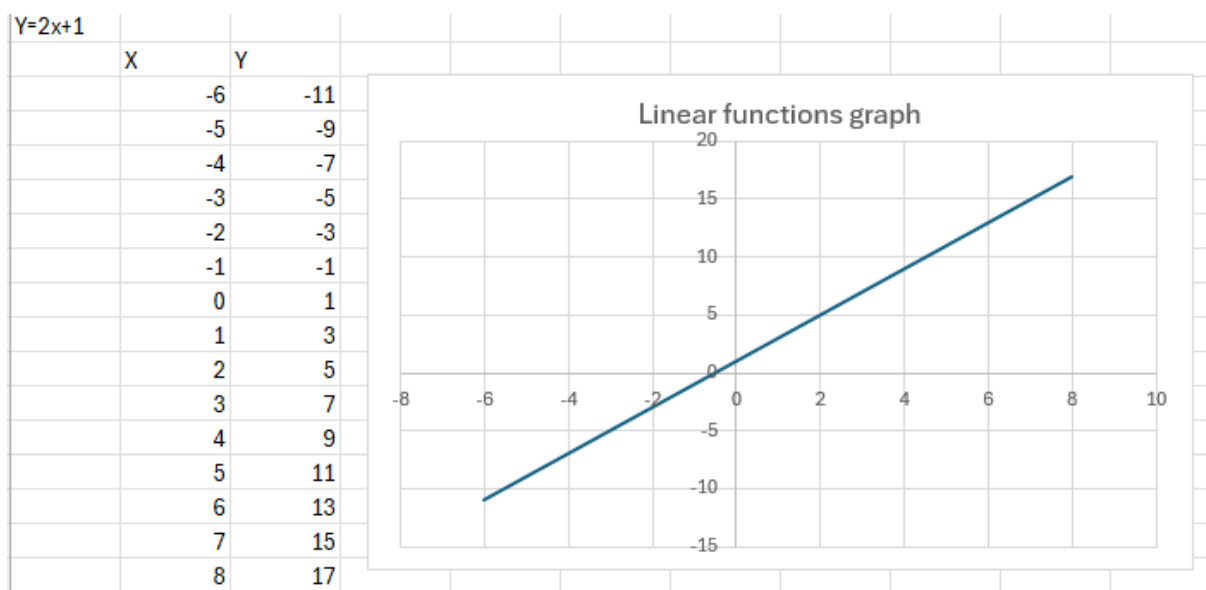
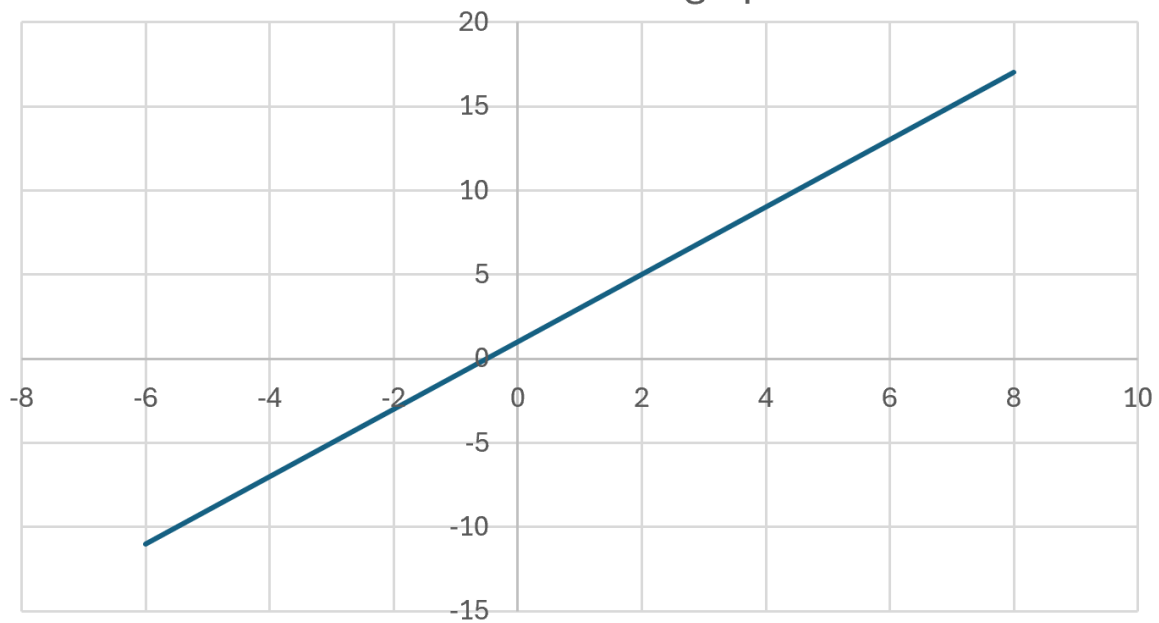
t = time

x = position

This means the character moves 4 units every second

This is a linear function because the movement is constant.

Linear functions graph



Obtaining the Equation of a Straight Line from a Graph

To find the equation of a straight line from a graph, we follow simple steps.

Step 1: Choose Two Points

Pick two clear points from the graph.

Example

(2, 5) and (4, 9)

Step 3: Find the Y-Intercept (c)

Use the formula:

$$y = mx + c$$

Substitute one point

$$5 = 2(2) + c$$

$$5 = 4 + c$$

$$c = 1$$

Step 4: Write Final Equation

$$y = 2x + 1$$

Why This Is Important

Finding the equation helps us

- Predict future values
- Understand patterns
- Control systems in computing
- Represent movement and data

Real-Life Computing Examples

- App animations → smooth movement

- Games → character position
- Data graphs → showing trends
- Business → sales growth

Simple Understanding

Linear function = straight line

Constant change = easy to predict

Overview

Functions are very important in computing because they help control and represent how values change. Linear functions are especially useful because they show constant change, which is easy to calculate and use in programming.

Finding the equation of a straight line from a graph helps us understand relationships between variables and predict results. This is very useful in real-world applications such as games, animations, and data analysis.

Finding the Equation of a Straight Line (Step by Step)

Step 1: Identify Two Points on the Line

The first step is to pick two clear points from the graph. These points should be easy to read, where the line touches exact grid values. This helps to avoid mistakes and gives a more accurate answer.

Example points:

- $(x_1, y_1) = (2, 4)$
- $(x_2, y_2) = (5, 10)$

These two points will be used to find the gradient (slope) and later the full equation.

Step 2: Calculate the Gradient (m)

The gradient tells us how fast the line is going up or down. It shows how much y changes when x increases by 1.

The formula is:

$$m = (y_2 - y_1) \div (x_2 - x_1)$$

Now we calculate:

$$m = (10 - 4) \div (5 - 2)$$

$$m = 6 \div 3$$

$$m = 2$$

This means when x increases by 1, y increases by 2.

So, the line has a constant rate of change.

Step 3: Find the Y-Intercept (c)

Now we find where the line starts. This is called the y-intercept, which is the value of y when x = 0.

We use the formula:

$$y = mx + c$$

Now put one point into the formula:

$$4 = 2(2) + c$$

$$4 = 4 + c$$

$$c = 0$$

This means the line crosses the y-axis at 0, which is the origin.

Step 4: Write the Final Equation

Now we know:

- $m = 2$

- $c = 0$

So the final equation is:

$$y = 2x$$

This shows that y always increases by 2 when x increases by 1.

What This Means in Simple Words

- The line is straight
- The change is constant
- Easy to predict values

Example in Computing

In computing, this type of equation is very useful.

For example:

- A game character moving at constant speed
- Animation moving from left to right
- Data increasing at a steady rate

Example:

If a character moves 2 steps every second:

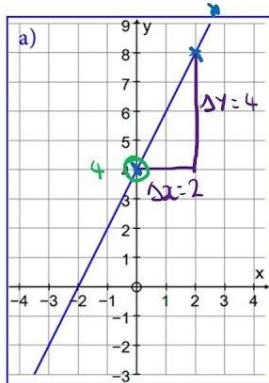
$$\text{Position} = 2 \times \text{time}$$

$$\text{Equation} = y = 2x$$

$$y = mx + c$$

Learning Objective: Derive the gradient of a straight line in the form $y = mx + c$.

Work out the equation of this straight line in the form $y = mx + c$.



Gradient (m)

$$\frac{\Delta y}{\Delta x} = \frac{4}{2} = 2$$

$x=0$
 $y=4$

$y = 2x + 4$

$y = mx + c$ Intercept $x=0$
Gradient = $\frac{\Delta y}{\Delta x}$

mr-mathematics.com

Conclusion

Linear functions are very important in computing because they show simple and constant change. This makes them easy to use in programming and problem solving. Many systems in IT need values that change in a straight and predictable way, and linear functions help to do this.

Learning how to find the equation of a line from a graph is not only useful in maths, but also in real-life computing tasks. It helps us understand how data changes and how to control systems properly. For example, in game development, a character may need to move smoothly across the screen. This movement can be controlled using a linear equation. In the same way, applications that track data like temperature or sales also use linear functions to show changes over time.

Another important benefit is that linear equations help us predict future values. Once we know the equation, we can easily calculate what will happen next. This is very useful in programming, where systems need to react quickly and correctly.

Also, understanding gradient and y-intercept helps programmers control systems better. The gradient shows speed or rate, while the intercept shows the starting point. These are very important in designing animations, simulations, and data systems.

Overall, linear functions make problem solving easier because they break complex problems into simple steps. They help students and developers think in a logical way. This is a key part of computational thinking, which is very important in IT.

In conclusion, learning linear functions improves skills in programming, data analysis, and system design. It helps students move from basic understanding to more advanced problem solving in computing.

Computational Uses of Boolean Operations

What Are Boolean Operations Boolean Logic and Logic Gates?

Boolean logic is an important concept in computing and digital electronics. It is based on the work of mathematician George Boole. Boolean logic uses only two values:

1 = True (ON)

0 = False (OFF)

Computers use these two values to make decisions and perform calculations. In digital circuits, Boolean logic is implemented using logic gates. Logic gates are electronic components that process one or more inputs and produce an output.

The most basic logic gates are AND, OR, and NOT. These gates are used in many computing systems such as processors, memory devices, and control circuits.

AND Gate

An AND gate produces an output of 1 (true) only when both inputs are 1. If one or both inputs are 0, the output will be 0.

This means that both conditions must be true for the output to be true.

Example

If a computer program checks two conditions:

- The user entered the correct password
- The user has permission to access the system

Both conditions must be true for access to be granted.

AND Gate Truth Table

INPUT A	INPUT B	OUTPUT
0	0	0
0	1	0
1	0	0
1	1	1

This table shows that the output becomes 1 only when both inputs are 1.

OR Gate

An OR gate produces an output of 1 when at least one input is 1.

This means the output will be true if either condition is true or both are true.

Example

A computer system might allow access if:

- The user enters the password correctly
- OR
- The user scans a fingerprint.

If either condition is true, the system allows access.

OR Gate Truth Table

INPUT A	INPUT B	OUTPUT
0	0	0
0	1	0
1	0	0
1	1	1

This table shows that the output is 1 if at least one input is 1.

NOT Gate

A NOT gate is different from AND and OR gates because it only has one input.

The NOT gate reverses the input value.

If the input is 1, the output becomes 0.

If the input is 0, the output becomes 1.

This operation is also called inversion.

NOT Gate Truth Table

INPUT	OUTPUT
0	1
1	0

This means the NOT gate always produces the opposite value of the input.

Real-World Computing Examples

Boolean logic and logic gates are used in many real-world computing systems.

1. Computer Processors

The CPU (Central Processing Unit) uses millions of logic gates to perform calculations and process instructions. These gates allow the computer to perform logical decisions during program execution.

2. Login Systems

Login systems use Boolean logic to check multiple conditions.

Example:

Access granted if:

Password is correct AND username exists.

This means both conditions must be true.

3. Security Systems

Security systems use logic gates to detect conditions.

Example:

An alarm may activate if:

Door sensor = open OR window sensor = open.

This means the alarm activates if any sensor detects a problem.

4. Traffic Light Systems

Traffic lights use logic gates to control signals.

Example:

Green light may activate only when:

Timer finished AND pedestrian button not pressed.

5. Smart Home Devices

Smart home systems also use Boolean logic.

Example:

Lights turn on when:

Motion detected AND it is dark.

Conclusion

Boolean logic is a fundamental concept in computing. It allows computers to make decisions using simple true or false values. Logic gates such as AND, OR, and NOT are used to perform logical operations inside digital circuits.

These gates are essential for modern computing systems, including processors, security systems, and automated technologies.

Understanding Boolean logic helps students learn how computers process information and control different operations.

Logic gate diagram

Logic Gate Diagrams

1. AND Gate Diagram

The AND gate gives an output of 1 only when both inputs are 1.

Input A ----\
)---- Output (A AND B)
Input B ----/
 AND

Example

If

$A = 1$ and $B = 1 \rightarrow \text{Output} = 1$

If

$A = 1$ and $B = 0 \rightarrow \text{Output} = 0$

AND Gate Truth Table

Input A	Input B	Output
---------	---------	--------

NOT gate:

Input A	Output
0	1
1	0

AND gate:

0	0	0
0	1	0
1	0	0
1	1	1

OR gate:

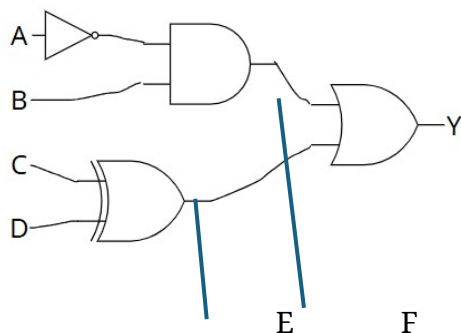
Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	1

XOR gate:

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Practice examples of Boolean circuits are:

$$Y = \bar{A}.B + C \oplus D$$



This logic circuit above shows that the A symbol will go through a NOT gate so the value of A will switch to the opposite value, the B and the A will go through the AND gate and if both units are on then the output of the AND gate will be on, if one of the inputs are off then the AND gate's output will be off. For the XOR gate either C's input or D's input has to be on so the output of the XOR gate will be on however if both C and D's inputs are on or off then the output of the XOR gate will be off. And finally, if either E or F is on and if both E and F are on the output of the OR gate (Y) will be on and if E and F is zero then the output of the OR gate (Y) will be off.

Using Complex Boolean Logic in Series

In computing, we can connect many logic gates together to make more complex systems. This is called Boolean logic in series. Instead of using only one gate, we use multiple gates to control outputs in a better way.

Each logic gate has a different job. When we combine them, they can solve bigger problems and give more accurate results.

How It Works

- **AND gate** → needs both inputs to be 1
- **OR gate** → needs at least one input to be 1
- **NOT gate** → changes 1 to 0 and 0 to 1
- **XOR gate** → gives 1 only when inputs are different

When we connect these gates together, the output of one gate becomes the input of another gate.

This creates a chain of logic operations

Example Explanation

In this circuit:

- First, input A goes into a NOT gate, so it becomes E (NOT A)
- Then, inputs C and D go into an XOR gate, giving output F

- After that, E and B go into an AND gate, giving output G
- Finally, F and G go into an OR gate, which gives the final output Y

So the output depends on multiple steps, not just one condition

Truth Table Explanation

A truth table shows all possible input values and their outputs.

Columns:

- A, B, C, D → inputs
- $E = \text{NOT } A$
- $F = C \text{ XOR } D$
- $G = E \text{ AND } B$
- $Y = F \text{ OR } G$

Simple Understanding

Step by step:

1. Change A using NOT → E
2. Compare C and D using XOR → F
3. Check E and B using AND → G
4. Final output using OR → Y

This means:

- Output can come from two different paths
- Either XOR part OR AND part can make output = 1

Why This Is Important

Using multiple logic gates together helps in:

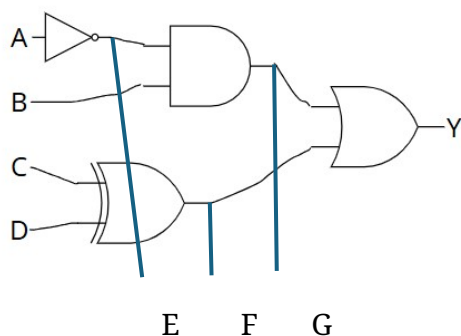
- Complex decision making
- Designing circuits
- Programming conditions
- Creating advanced systems

Real-Life Examples

- Security systems → multiple conditions checked
- Games → different actions based on inputs
- Smart devices → combine sensors and logic
- Computers → processing millions of logic operations

Overview

Complex Boolean logic allows computers to handle bigger problems by combining simple logic gates. Instead of checking only one condition, the system can check many conditions step by step. This makes computing more powerful and flexible



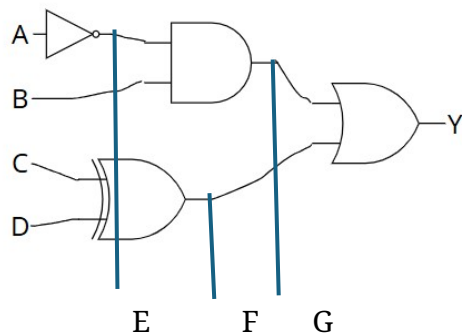
Combining a NOT gate, AND gate, XOR gate and OR gate will look like this as a truth table:

A	B	C	D	E (NOT A)	F (C XOR D)	G (E AND B)	Y (F OR G)
0	0	0	0	1	0	0	0
0	0	0	1	1	1	0	1
0	0	1	0	1	1	0	1
0	0	1	1	1	0	0	0
0	1	0	0	1	0	1	1
0	1	0	1	1	1	1	1
0	1	1	0	1	1	1	1
0	1	1	1	1	0	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	1
1	0	1	0	0	1	0	1
1	0	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	1	0	1	0	1	0	1
1	1	1	0	0	1	0	1
1	1	1	1	0	0	0	0

Solving problems using logic gates

A problem scenario for Boolean I made is: $Y = \bar{A}.B + C \oplus D$, the full meaning of this Boolean logic example is ((NOT A) AND B) OR (C XOR D) = Y. Which contains 4 Boolean logic gates which are AND, NOT, OR and XOR.

The gate diagram for the problem scenario for Boolean ($Y = \bar{A}.B + C \oplus D$) is:



The truth table for this Boolean logic gate diagram is:

A	B	C	D	E (NOT A)	F (C XOR D)	G (E AND B)	Y (F OR G)
0	0	0	0	1	0	0	0
0	0	0	1	1	1	0	1
0	0	1	0	1	1	0	1
0	0	1	1	1	0	0	0
0	1	0	0	1	0	1	1
0	1	0	1	1	1	1	1
0	1	1	0	1	1	1	1
0	1	1	1	1	0	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	1
1	0	1	0	0	1	0	1
1	0	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	1	0	1	0	1	0	1
1	1	1	0	0	1	0	1
1	1	1	1	0	0	0	0

In this truth table, the E column shows the opposite value of A because it uses a NOT gate. This means if A is 1, then E becomes 0, and if A is 0, then

E becomes 1. The F column shows the result of C XOR D. This means the output will be 1 only when one input is 1 and the other is 0. If both inputs are the same, the output will be 0. The G column shows E AND B. Since AND gate needs both inputs to be 1, the output will only be 1 when both E and B are 1. If one or both inputs are 0, then G will be 0. Finally, the Y column shows F OR G. This means the output will be 1 if at least one of F or G is 1. If both inputs are 0, then the output will be 0.

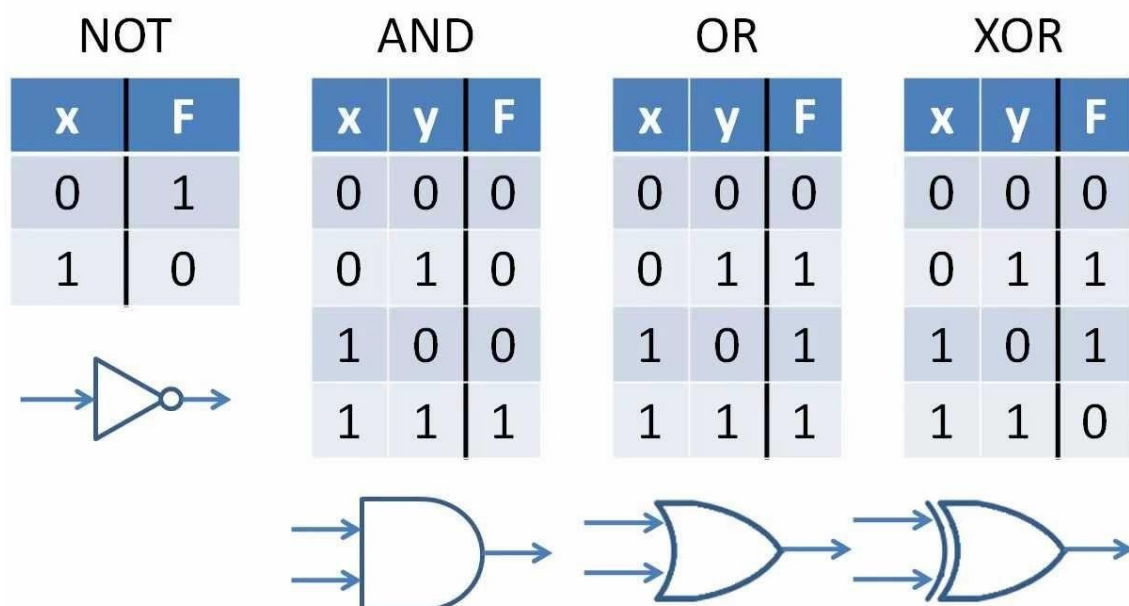
Only one condition needs to be true.

• NAND gate

• NOR gate

• XOR gate

• XNOR gate



Additional Logic Gates

1. NAND Gate (NOT + AND)

A NAND gate is the opposite of an AND gate. It first performs the AND operation and then reverses the result using NOT.

This means the output will be 0 only when both inputs are 1. In all other cases, the output will be 1.

NAND Gate Diagram

Input A ----\)----o---- Output
 / Input B ----/
 NAND

NAND Gate Truth Table

A	B	OUTPUT
0	0	1
0	1	1
1	0	1
1	1	0

Example in Computing

NAND gates are very important in computer hardware. Many processors and memory systems are built mainly using NAND gates because they are simple and efficient.

2. NOR Gate (NOT + OR)

A NOR gate is the opposite of an OR gate. It performs the OR operation and then reverses the output.

The output will be 1 only when both inputs are 0.

NOR Gate Diagram

Input A ----\)----o---- Output
Input B ----/ NOR

NOR Gate Truth Table

A	B	OUTPUT
0	0	1
0	1	0
1	0	0
1	1	0

Example in Real Systems

NOR gates can be used in alarm systems. The alarm may stay off only if no sensors detect a problem.

If any sensor sends a signal, the system reacts.

3. XOR Gate (Exclusive OR)

An XOR gate gives an output of 1 only when the inputs are different.

If both inputs are the same (both 0 or both 1), the output is 0.

XOR Gate Diagram

Input A ----\
))---- Output
Input B ----/
 XOR

(The double curved line shows it is an XOR gate.)

XOR Gate Truth Table

A	B	OUTPUT
0	0	0
0	1	1
1	0	1
1	1	0

Example in Computing

XOR gates are used in error detection systems and encryption algorithms. They are also used in computer processors to compare data values.

4. XNOR Gate (Exclusive NOR)

An XNOR gate is the opposite of an XOR gate.

It produces an output of 1 when both inputs are the same.

If the inputs are different, the output will be 0.

XNOR Gate Diagram

Input A ----\
))----o---- Output
Input B ----/
 XNOR

XNOR Gate Truth Table

A	B	OUTPUT
0	0	1
0	1	0
1	0	0
1	1	1

Example in Real Systems

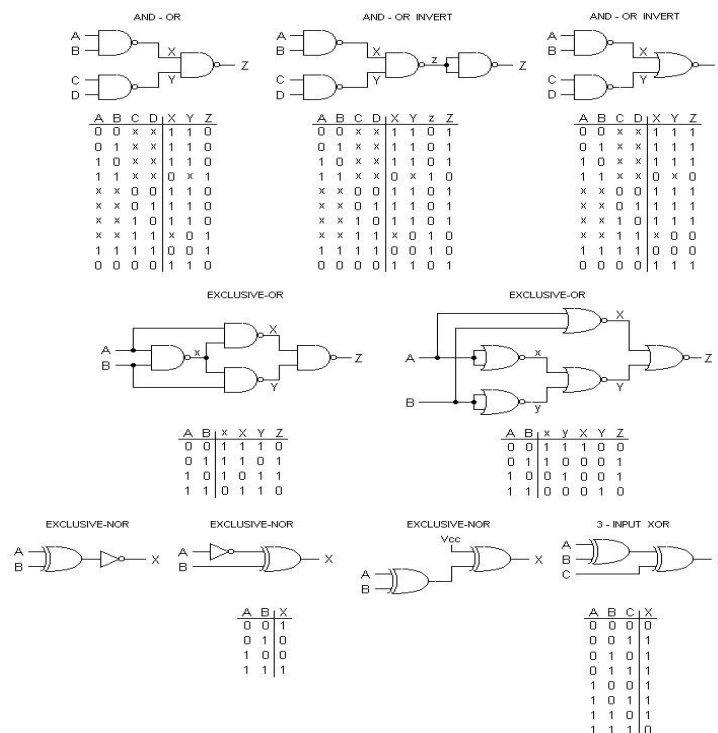
XNOR gates are often used in digital comparison circuits. For example, they can check if two binary numbers are the same.

Summary of Logic Gates

Boolean Algebra and Logic Circuits

Key Types of Logic Gates

- AND: Outputs 1 only if *all* inputs are 1.
- OR: Outputs 1 if *at least one* input is 1.
- NOT (Inverter): Reverses the input (1 becomes 0, 0 becomes 1).
- NAND: AND followed by NOT. Outputs 0 only if all inputs are 1.
- NOR: OR followed by NOT. Outputs 1 only if all inputs are 0.
- XOR (Exclusive OR): Outputs 1 if inputs are different.
- XNOR (Exclusive NOR): Outputs 1 if inputs are the same



- **Universal Gates:** NAND and NOR are called "universal gates" because any other logic gate can be built using only combinations of these two types.
- **Truth Tables:** A standard way to represent a gate's logic by listing every possible input combination and its resulting output.
- **Physical Hardware:** In modern electronics, these gates are typically made of transistors (specifically MOSFETs) etched onto silicon.

Computational Uses of Boolean Operations

Boolean logic is used in many areas of computing and digital systems. It helps computers make decisions by checking conditions that are either true or false. Boolean operations are very important in programming, hardware, games, and security systems.

Programming

In programming, Boolean logic is used to check if certain conditions are true before the program performs an action. This helps programs work correctly and avoid errors.

For example, a system may only allow access if all conditions are correct

Example

```
if age > 18 and has_ticket:
```

`allow_entry = True`

In this example

The person must be older than 18

The person must also have a ticket

Both conditions must be true for access to be allowed.

Boolean logic helps programs make safe and correct decisions.

Search Engines

Search engines also use Boolean logic to improve search results.

For example

“Cats AND dogs” → results contain both words

“Cats OR dogs” → results contain one or both words

“Cats NOT dogs” → results contain cats but not dogs

This helps users find information more easily and quickly.

Hardware and Circuit Design

Boolean logic is very important in computer hardware and electronic circuits.

Logic gates such as

AND

OR

NOT

are built into computer processors and circuits.

These gates control how electrical signals move inside a computer. Millions of Boolean operations happen every second inside CPUs and digital devices.

Without Boolean logic, computers would not be able to process data.

Access Control Systems

Security systems use Boolean logic to decide if access should be allowed.

For example:

A door opens only if the correct PIN is entered AND the access card is scanned.

Both conditions must be true.

This makes systems more secure because multiple checks are used before access is given.

Game Development

Boolean logic is also used in games to control gameplay and actions.

For example,

A player jumps if the jump button is pressed

A shield activates if the player presses a key OR power becomes active

A game ends if health becomes 0

Boolean operations help game developers control events and create smooth gameplay.

Logic Gate Diagrams and Truth Tables

AND Gate

Rule

The output is 1 only when both inputs are 1.

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

Example Use

In a game, a door opens only if:

The player has the key

AND

The door is unlocked

OR Gate

Rule

The output is 1 if one or both inputs are 1

A	B	A and B
0	0	0
0	1	1
1	0	1
1	1	1

Example Use

A shield activates if

The player presses the shield button

OR

Automatic protection turns on

NOT Gate

Rule

The output is the opposite of the input.

A	Not A
0	1
1	0

Example Use

If it is NOT raining, the sprinkler system starts.

Combined Logic Example

Scenario:

A fan turns on if

The room is hot (A)

AND

The window is NOT open (B)

Logic expression:

Fan = A AND (NOT B)

A (hot)	B (window open)	Not B	A AND (NOT B)
0	0	1	0
0	1	0	0
1	0	1	1
1	1	0	0

Conclusion

In this section, I learned about Boolean logic and how it is used in computing systems. Boolean operations such as AND, OR, and NOT help computers make decisions and control different actions.

I also learned that Boolean logic is used in many real-world systems such as programming, security systems, search engines, and game development. Truth tables and logic gates help show how inputs and outputs work together in digital systems.

Another important thing I understood is that computers use Boolean logic to process information and automate tasks. Without Boolean operations, modern computer systems would not work properly.

Overall, learning Boolean logic helped me understand how computers think and make decisions. It is an important part of computational thinking and is widely used in software development, digital electronics, and IT systems.

4. Representing Domain and Range Using Software

Understanding Domain and Range

In computational thinking, domain and range are important because they help us understand the values that can be used in calculations and functions.

- The **domain** means the input values that we can enter into a function.
- The **range** means the output values that we get from the function.

Understanding domain and range is useful in programming because it helps developers know what inputs are allowed and what outputs can be expected. This also helps systems avoid errors and work correctly.

For example, in games or animations, domain and range can help control the position and movement of objects on the screen. By understanding these values, programmers can predict how a system will behave.

Using Excel to Plot and Analyse Functions

I used Microsoft Excel to create a set of x-values ranging from -10 to 10 to find my dataset. For my linear and quadratic outcomes, I then applied these formulas to create y-values under a linear function and quadratic condition:

- **Linear Function:** Linear $y = 2x + 3$
- **Quadratic Function:** $y = x^2 - 4$

Here is what the spreadsheet looked like

X	Linear ($y = 2x + 3$)	Quadratic ($y = x^2 - 4$)
-10	-17	96
-9	-15	77
-8	-13	60
-7	-11	45
-6	-9	32
-5	-7	21
-4	-5	12
-3	-3	5
-2	-1	0
-1	1	-3
0	3	-4
1	5	-3
2	7	0
3	9	5
4	11	12
5	13	21
6	15	32
7	17	45
8	19	60
9	21	77
10	23	96

All the calculated values were plotted in Microsoft Excel by using **Insert → Chart → Scatter Plot**. The graphs were created successfully from both equations. The scatter graph helped me place all the values on the (x, y) axis as points and then connect them together to clearly see the shape of each function.

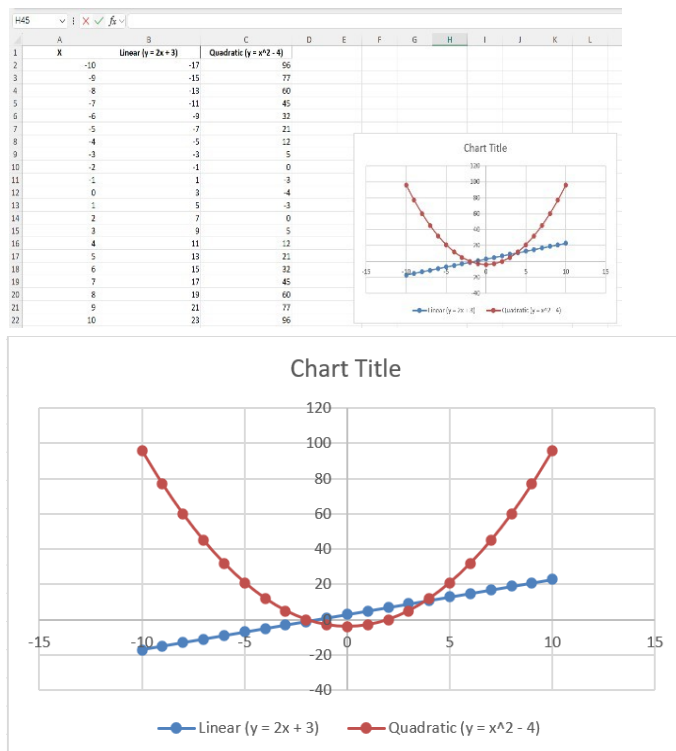
For the first equation, **$y = 2x + 3$** , the graph created a straight line. This showed that the values were increasing at a constant rate and the gradient stayed the same. The points were equally spaced, which made the graph linear.

For the second equation, **$y = x^2 - 4$** , the graph created a curved shape called a parabola. This showed that the y-values increased more quickly as the x-values moved further away from zero. The curve helped me understand how quadratic functions behave differently from linear functions.

Excel also gave me useful tools to change the graph design. I was able to adjust the maximum and minimum axis values, add labels to points, and

clearly identify where the graph crossed the axes. This made the graphs easier to understand and analyse.

These visual graphs helped me understand how the domain affects the range in different functions. It also showed me how functions are useful in computing and programming, especially in areas such as animation movement, collision detection in games, and user interface positioning.



Defining Domain and Range

Linear Function $y = 2x + 3$

- Domain: All real numbers (-10 to 10 in this case)
- Range: All real numbers (increases or decreases forever)
- Graph Behaviour: Straight line, consistent slope

Quadratic Function $y = x^2 - 4$

- Domain: All real numbers (-10 to 10 in this case)
- Range: Minimum -4 and then increases (minimum value occurs at vertex: $x = 0$)
- Graph Behaviour: U-shaped parabola

Excel provided a concrete sense of how changes in the domain of the input

values would lead to changes in the range of output values over time, an understanding learned best by zooming in on particular sections of a few of the output graphs.

[Link for Excel worksheet](#)

Shahzaib Asim (2026). [Unit 18 representing quadratic and linear functions for assignment 2.xlsx](#). this is my second excel sheet first one got deleted

Shahzaib Asim (2026). [unit 18 assignment 2.xlsx](#).

Real-World Relevance in Computing

In computing, these representations are essential for

Game development: Using quadratic curves to calculate jumps, trajectories, and object location.

Therefore, by using Excel or graphing software such as Desmos, I was able to test different results and possibilities before creating the actual code. This helps save time and reduces mistakes in the program. It also makes digital systems more accurate, reliable, and easier to manage. By checking the graph first, programmers can understand how the function will behave before using it in software or games.

Summary

Overall, using spreadsheet software to identify and display domain and range helped me improve both my maths and computing knowledge. It allowed me to understand how functions work and how different values affect the final result. Visual graphs made it easier to see patterns, shapes, and changes in the data.

This type of learning is very useful in computational thinking because it helps developers test ideas, solve problems, and create more reliable systems. These skills are important in many areas of IT such as software development, game design, robotics, simulations, and automation systems.

5. Using Complex Boolean Logic in Series

Combining Multiple Boolean Operations

In computing, Boolean logic is not always made with only one logic gate. Many digital systems use multiple logic operations together in series to make more advanced decisions. This means the output from one logic gate becomes the input for another logic gate. These combinations help computers and smart systems work more accurately and safely.

Complex Boolean logic is very important in real-world technology because many systems need to check several conditions before giving a final output. For example, a smart security system may need to check a password, fingerprint, and security code before allowing access. This cannot be done using only one simple logic gate.

Different Boolean operations such as AND, OR, NOT, XOR, NAND, and NOR can be combined together to create more advanced circuits. These circuits are used in automation systems, alarm systems, robots, traffic lights, gaming systems, and computer processors.

For example:

- An **AND gate** checks if all conditions are true.
- An **OR gate** checks if at least one condition is true.
- A **NOT gate** reverses the input.
- An **XOR gate** checks if only one input is true.

When these gates are connected together, they can create powerful logical systems that make decisions automatically.

Truth Tables for Logic in Series

Truth tables are used to show all possible combinations of inputs and outputs in a Boolean expression. They help programmers and engineers understand how the logic circuit will behave before building the real system.

Example 1: (A AND B) OR C

This logic means:

- A and B must both be true,
- OR C can also make the final output true.

Truth Table

A	B	C	A AND B	OUTPUT
0	0	0	0	0
0	1	0	0	0
1	0	0	0	0

1	1	0	1	1
0	0	1	0	1
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Explanation

This logic can be used in a security system. A person can enter if:

- They have the correct key card AND fingerprint,
- OR if the manager presses the override button.

Example 2: NOT (A OR B)

This expression first checks A OR B and then reverses the result using a NOT gate.

Truth Table

A	B	A OR B	NOT A OR B
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Explanation

This logic can be used in alarm systems. The alarm stays active only when both sensors are OFF.

Example 3: (A OR B) AND (NOT C)

This logic combines OR, NOT, and AND gates together.

Truth Table

A	B	C	A OR B	NOT C	OUTPUT
0	0	0	0	1	0
0	1	0	1	1	1
1	0	0	1	1	1
1	1	0	1	1	1
0	0	1	0	0	0
0	1	1	1	0	0
1	0	1	1	0	0
1	1	1	1	0	0

Explanation

This system can be used in a smart fan or air conditioning system:

- The fan turns on if Sensor A or Sensor B detects heat,
- But the fan stays OFF if the manual stop button is pressed.

Logic Gate Circuits for Real-World Scenarios

Logic gate circuits are used in many real-life digital systems. Computers, phones, smart homes, robots, and gaming devices all use logic circuits to make decisions.

Security Systems

Security systems often use AND and OR gates together. For example:

- A door opens only if the password is correct AND the fingerprint matches.
- A manager override can also open the system using an OR gate.

Smart Home Systems

Smart homes use Boolean logic for automatic control.
For example:

- Lights turn on if motion is detected OR it becomes dark.
- Heating systems turn off if the room is warm.

Traffic Light Systems

Traffic lights use multiple logic gates to control timing and safety conditions. Sensors on the road send signals to the system, and the logic gates decide when lights should change.

Gaming Systems

Games use Boolean logic for player controls and actions.
For example:

- A character jumps only if the jump button is pressed AND the player is on the ground.
- A special power activates if one button OR another button is pressed.

Industrial Machines

Factories use logic circuits to control machines safely.
For example:

- A machine starts only if all safety conditions are true.
- Emergency stop buttons use NOT gates to stop dangerous actions immediately.

Conclusion

Using complex Boolean logic in series helps computers and digital systems make more advanced decisions. By combining multiple logic gates together, systems can check many conditions before giving an output. Truth tables help show all possible outcomes clearly, while logic gate circuits help control real-world technology safely and efficiently.

These Boolean operations are very important in computing because they are used in security systems, smart devices, games, robots, and automated machines. Learning how to combine Boolean logic improves computational thinking and helps students understand how digital systems work step by step.

6. Using Functions to Calculate Position and Plot Shapes

Functions for Positioning Objects on Screen

In computing and game development, functions are very important for controlling the position of objects on the screen. Functions help programmers decide where an object should move, how fast it should move, and what direction it should go. This is useful in games, animations, apps, simulations, and graphic design.

Most digital systems use coordinates to place objects on a screen. The horizontal position is usually called the x-axis, and the vertical position is called the y-axis. Functions are used to calculate these positions automatically.

For example, if a character moves across the screen at a constant speed, a linear function can be used.

Example:

$$x = 5t$$

Where

- x = position of the object
- t = time
- 5 = movement speed

This means the object moves 5 pixels every second. As the time increases, the position also increases in a straight line.

Functions make movement smoother and more realistic because the computer can calculate the new position continuously.

Linear and Quadratic Applications in Motion

Linear Functions in Motion

Linear functions create straight-line movement. They are often used when an object moves at the same speed without changing direction.

Linear Function Formula

$$y = mx + c$$

Where:

- m = gradient or speed
- c = starting position

Real Example

A car in a racing game moves forward 10 pixels every second.

$$y = 10x + 5$$

This means:

- The car starts at position 5
- Every second, it moves 10 more pixels

Uses of Linear Functions

- Character movement
- Side-scrolling games
- Menu animations
- Object tracking
- Progress bars

Linear functions are useful because they are simple and predictable.

Quadratic Functions in Motion

Quadratic functions create curved movement. They are mostly used when movement changes over time, especially with gravity and jumping effects.

Quadratic Function Formula

$$y = ax^2 + bx + c$$

Quadratic functions create a curve called a parabola.

Example

A character jumps in a game.

$$y = -5x^2 + 20x$$

This means:

- The character moves upward first
- Then slows down
- Then falls back down because of gravity

This curved movement looks more realistic than a straight line.

Uses of Quadratic Functions

- Jumping mechanics
- Ball movement
- Physics simulations
- Animation effects
- Projectile motion

Quadratic functions are important in modern games because they make movement feel natural.

Plotting and Drawing Shapes with Functions

Functions are also used to draw shapes and graphics in computing. Computers use mathematical coordinates to create circles, lines, curves, and patterns.

For example

- Linear functions can draw straight lines
- Quadratic functions can create curves
- Multiple functions together can create complex shapes

Programs such as

- Microsoft Excel
- GeoGebra
- Desmos
- Scratch
- Python graphics libraries

can plot graphs and shapes using functions.

Example of Plotting a Straight Line

Function:

$$y = 2x + 1$$

Table of Values

X	Y
0	1
1	3
2	5
3	7

These points create a straight line on the graph.

Example of Plotting a Quadratic Curve

Function:

$$y = x^2$$

Table of Values

X	Y
-2	4
-1	1
0	0
1	1
2	4

These points create a U-shaped curve called a parabola.

Examples from Game and App Design

Functions are used in many real-world applications in computing.

1. Video Games

Games use functions for:

- Character movement
- Jumping and gravity
- Enemy movement
- Collision detection
- Shooting systems

For example:

- A football game uses quadratic functions for ball movement.
- A racing game uses linear functions for car speed.

2. Mobile Apps

Apps use functions for:

- Screen animations
- Sliding menus
- Interactive buttons
- Motion effects

For example:

- A menu moving smoothly from left to right uses a linear function.
- Bounce effects in apps often use quadratic functions.

3. Animation and Graphics

Animations use functions to control:

- Object size
- Rotation
- Position
- Timing

Graphic designers and animators use functions to create smooth transitions and visual effects.

4. Robotics

Robots use functions to:

- Move accurately
- Calculate directions
- Avoid obstacles
- Control arm movement

Functions help robots calculate exact positions and distances.

Conclusion

Functions are very important in computing because they help calculate positions, control movement, and draw shapes on screens. Linear functions are useful for straight and constant movement, while quadratic functions are useful for curved movement and realistic physics.

These functions are used in games, mobile apps, robotics, simulations, and animations. By understanding how functions work, programmers can create smoother, more accurate, and more interactive digital systems. This also improves computational thinking and problem-solving skills in IT and software development.

7. Solving Problems Using Logic Gates

Creating a Problem Scenario

Logic gates are very important in computing because they help digital systems make decisions automatically. In real life, many systems must check different conditions before giving an output. Logic gates allow computers and electronic systems to process these conditions step by step.

For this task, I created a smart security system scenario using multiple logic gates together.

Scenario: Smart Security Door System

A company office has a smart security door. The door should only open under the following conditions:

- The employee must enter the correct password (A)
- The employee must scan their ID card (B)
- The emergency lockdown switch must be OFF (C)

The door will open only if:

- Password AND ID card are correct
- AND the lockdown system is NOT active

This system is useful because it improves security and prevents

unauthorized access during emergencies.

Logic Gate Solution with Three or More Operations

The Boolean expression for this system is:

$$(A \text{ AND } B) \text{ AND } (\text{NOT } C)$$

This expression uses:

- AND gate
- NOT gate
- Another AND gate

This means the system uses three logic operations together in series.

Step-by-Step Logic Process

Step 1: A AND B

The system first checks:

- Is the password correct?
- Is the ID card valid?

Both must be true.

Step 2: NOT C

The system checks if the emergency lockdown is OFF.

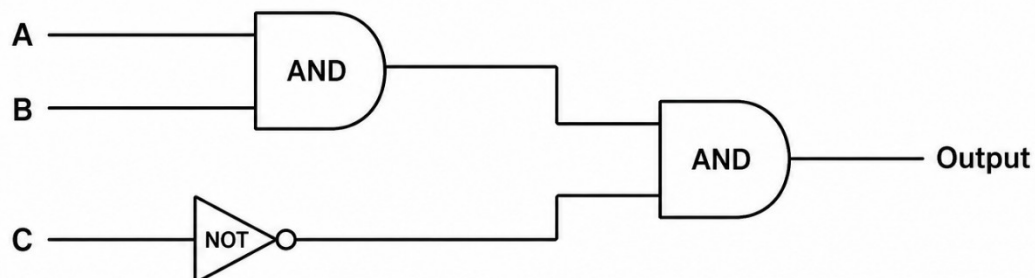
- If $C = 1 \rightarrow$ lockdown active
- NOT C changes it to 0
- Door stays locked

Step 3: Final AND Gate

The outputs are combined:

- (A AND B)
- AND (NOT C)

The door opens only if all conditions are true.



Boolean Expression:

Output = (A AND B) AND (NOT C)

Simplified:

Output = $A \cdot B \cdot C'$

Explanation:

The output is 1 only when
A = 1, B = 1, and C = 0.

A	B	C	NOT C	A AND B	Output
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	0
0	1	1	0	0	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	1	0

Explanation of Diagram

1. Inputs A and B go into the first AND gate.
2. Input C goes through a NOT gate.
3. The outputs from both gates enter the final AND gate.
4. The final output decides whether the door opens or stays locked.

Explanation of Results

The truth table shows that the final output is only equal to 1 in one

situation:

- Password correct = 1
- ID card valid = 1
- Emergency lockdown OFF = 0

This means the system only opens the door when all security conditions are correct.

If:

- the password is wrong,
- the ID card is missing,
- or the emergency lockdown is active,

then the output becomes 0 and the door stays locked.

This type of logic is very useful in real-world computing systems because it improves security, safety, and automation.

Real-World Applications

This type of Boolean logic can be used in:

- Smart door security systems
- Bank vault access systems
- Employee attendance systems
- Smart home automation
- Alarm systems
- Industrial safety systems

Large companies and organizations use similar logic systems to protect important information and equipment.

Conclusion

Logic gates help computers and digital systems solve problems automatically by checking different conditions. In this example, I used multiple Boolean operations together to create a smart security system.

The system used:

- AND gates
- NOT gates
- Multiple operations in series

The truth table and logic gate diagram clearly showed how the system works step by step. This task helped me understand how Boolean logic is used in real-world technology such as security systems, automation, and smart devices.

8. Conclusion

What We Learned About Functions and Logic

In this assignment, I learned about functions and Boolean logic and how they are used in computing and digital systems. These topics are very important in computational thinking because they help computers solve problems, make decisions, and control different operations.

I learned that functions are used to show relationships between values. Linear functions create straight-line movement, while quadratic functions create curved movement. These functions are very useful in games, animations, simulations, and app design because they help control motion, speed, position, and object behaviour on the screen.

I also learned how to find the equation of a straight line by calculating the gradient and y-intercept from a graph. This helped me understand how mathematical functions can be used in real computing situations. Using software such as Excel and graphing tools also helped me

understand domain and range more clearly because I could see the graphs visually.

Another important part of this assignment was Boolean logic. I learned how logic gates such as AND, OR, NOT, NAND, NOR, XOR, and XNOR work together to make decisions inside digital systems. Truth tables helped me understand all possible combinations of inputs and outputs.

I also created more complex Boolean expressions using multiple logic gates in series. This showed how computers process different conditions before giving a final result. Real-life examples such as security systems, smart homes, alarms, robots, and gaming systems helped me understand how Boolean logic is used every day in technology.

Why These Skills Matter in Digital Systems

Functions and Boolean logic are very important because modern digital systems depend on them. Almost every computer system, mobile app, game, website, or smart device uses mathematical functions and logic operations.

Functions are important because they help systems:

- calculate movement,
- position objects,
- predict changes,
- create graphs,
- and control animations.

Without functions, digital systems would not be able to calculate data correctly or create smooth movement in games and applications.

Boolean logic is also important because it allows computers to make decisions. Logic gates help systems check conditions and respond correctly. For example:

- A phone unlocks only if the password is correct.
- A smart light turns on when motion is detected.
- A game character jumps when the correct button is pressed.
- An alarm activates when danger is detected.

These systems work because Boolean operations process true and false conditions very quickly.

Learning these skills also improved my computational thinking and problem-solving ability. I now understand how complex systems can be broken down into smaller logical steps. This is an important skill for careers in:

- software development,
- cybersecurity,
- game development,
- robotics,
- engineering,
- and IT support.

Overall, this assignment helped me understand how mathematics and logic are connected to computing. It showed me how digital systems work behind the scenes and how programmers use functions and Boolean logic to create reliable and intelligent technology.

9. Practice Questions & Solutions

This section includes different practice questions about functions, Boolean logic, logic gates, graphs, and computational thinking. These questions helped me improve my understanding of how mathematical

functions and Boolean operations are used in computing systems.

Section 1 – Linear Functions

Question 1

Write the equation of a straight line with:

- Gradient = 3
- Y-intercept = 2

Solution

The formula for a linear equation is:

$$y = mx + c$$

Where:

- m = gradient
- c = y-intercept

Substitute the values:

$$y = 3x + 2$$

Final Answer

$$y = 3x + 2$$

Question 2

Find the gradient of the line using these points:

- (2, 5)
- (6, 13)

Solution

Formula:

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

Substitute values:

$$m = \frac{13-5}{6-2} m = \frac{8}{4} m = 2$$

Final Answer

Gradient = 2

Question 3

Find the y-intercept for the equation:

$$y = 4x + 7$$

Solution

In the equation:

$$y = mx + c$$

The y-intercept is the value of c.

Final Answer

Y-intercept = 7

Section 2 – Quadratic Functions

Question 4

Write the type of graph created by a quadratic function.

Solution

Quadratic functions create a curved graph called a parabola.

Final Answer

A parabola

Question 5

Find the value of y when:

$$y = x^2 + 2x + 1$$

and

$$x = 3$$

Solution

Substitute $x = 3$

$$y = 3^2 + 2(3) + 1 \quad y = 9 + 6 + 1 \quad y = 16$$

Final Answer

$$y = 16$$

Question 6

Give one real-world use of quadratic functions in computing.

Solution

Quadratic functions are used in video games to create jumping movement and gravity effects.

Final Answer

Game character jumping physics

Section 3 – Boolean Logic

Question 7

What does an AND gate do?

Solution

An AND gate only gives an output of 1 when both inputs are 1.

Final Answer

Both inputs must be true for the output to be true.

Question 8

Complete the truth table for an OR gate.

A	B	Output
0	0	?
0	1	?
1	0	?
1	1	?

Solution

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

Question 9

What does a NOT gate do?

Solution

A NOT gate reverses the input value.

Final Answer

- 1 becomes 0
- 0 becomes 1

Question 10

What is the output of XOR when both inputs are 1?

Solution

XOR only gives 1 when the inputs are different.

Final Answer

Output = 0

Section 4 – Complex Boolean Operations

Question 11

Solve this Boolean expression:

$$(A \text{ AND } B) \text{ OR } C$$

when:

- $A = 1$
- $B = 1$
- $C = 0$

Solution

Step 1:

$$A \text{ AND } B = 1 \wedge 1 = 1$$

Step 2:

$$1 \vee 0 = 1$$

Final Answer

Output = 1

Question 12

Find the output:

$$(A \vee B) \wedge (\neg C)$$

when:

- $A = 0$
- $B = 1$
- $C = 1$

Solution

Step 1:

$$A \vee B = 0 \vee 1 = 1$$

Step 2:

$$\neg C = \neg 1 = 0$$

Step 3:

$$1 \wedge 0 = 0$$

Final Answer

Output = 0

Section 5 – Domain and Range

Question 13

What is the domain of a function?

Solution

The domain is all possible input values of a function.

Final Answer

The set of input values

Question 14

What is the range of a function?

Solution

The range is all possible output values of a function.

Final Answer

The set of output values

Section 6 – Real-World Applications

Question 15

Give one example of Boolean logic in real life.

Solution

A smart door system opens only if the correct password AND fingerprint are entered.

Final Answer

Security access systems

Question 16

Give one example of functions used in games.

Solution

Functions are used to move characters and objects around the screen.

Final Answer

Character movement and animations

Section 7 – Extended Questions

Question 17

Explain why Boolean logic is important in computing.

Solution

Boolean logic is important because it helps computers make decisions using true and false conditions. Digital systems use Boolean operations to control programs, circuits, alarms, games, and security systems.

Question 18

Explain the difference between linear and quadratic functions.

Solution

Linear functions create straight-line graphs and change at a constant rate. Quadratic functions create curved graphs called parabolas and change at different rates.

Final Challenge Questions

Question 19

Convert 25 into binary.

Solution

25 ÷ 2 method:

- $25 \div 2 = 12$ remainder 1
- $12 \div 2 = 6$ remainder 0
- $6 \div 2 = 3$ remainder 0
- $3 \div 2 = 1$ remainder 1
- $1 \div 2 = 0$ remainder 1

Read from bottom to top:

Final Answer

$$25_{10} = 11001_2$$

Question 20

Convert binary 101101 into decimal.

Solution

$$1(2^5) + 0(2^4) + 1(2^3) + 1(2^2) + 0(2^1) + 1(2^0) = 32 + 0 + 8 + 4 + 0 + 1 = 45$$

Final Answer

$$101101_2 = 45_{10}$$

Summary

These practice questions helped me improve my understanding of:

- Linear and quadratic functions
- Boolean logic
- Logic gates
- Truth tables
- Domain and range
- Real-world computing applications

The questions also helped me understand how mathematical functions and Boolean operations are connected to programming, digital systems, games, and automation.

Term	Definition
Algorithm	A set of step-by-step instructions used to solve a problem or complete a task in computing.

Animation	A process in computing where images or objects are moved to create the illusion of motion.
AND Gate	A logic gate that only gives an output of 1 when both inputs are 1.
Binary	A number system that only uses 0 and 1. Computers use binary to process and store data.
Boolean Logic	A type of logic used in computing that works with true/false or 1/0 values.
Circuit	A connected path used in electronic systems to carry signals and data.
Computational Thinking	A problem-solving method that uses logical steps and computing concepts to find solutions.
Term	Definition
Coordinate	A set of values used to show the position of an object on a graph or screen.
Data Representation	The way information is stored and shown inside a computer system.
Decimal	A base-10 number system that uses digits from 0 to 9.
Domain	The set of all possible input values in a function.
Equation	A mathematical statement showing that two expressions are equal.
Function	A mathematical rule that connects input values to output values.

Game Physics	The use of mathematical calculations in games to create realistic movement and motion.
Gradient	The steepness or rate of change of a straight line. It is also called slope.
Graph	A visual chart used to show relationships between values or functions.
Hardware	The physical parts of a computer system such as keyboard, monitor, and processor.
Hexadecimal	A base-16 number system using numbers 0–9 and letters A–F.
Input	Data or instructions entered into a computer system.
Integer	A whole number that can be positive, negative, or zero.
Term	Definition
Linear Function	A function that creates a straight-line graph and changes at a constant rate.
Logic Gate	An electronic component that performs Boolean operations such as AND, OR, and NOT.
Microprocessor	A small electronic chip that performs calculations and controls computer operations.
Motion Control	The use of calculations and functions to control movement in games or software.
NAND Gate	A logic gate that gives the opposite result of an AND gate.

Natural Numbers	Counting numbers such as 1, 2, 3, 4, and so on.
NOR Gate	A logic gate that gives the opposite result of an OR gate.
NOT Gate	A logic gate that reverses the input value.
OR Gate	A logic gate that gives an output of 1 if at least one input is 1.
Output	The final result produced by a computer system or function.
Parabola	A curved graph shape created by a quadratic function.
Plotting	Placing points on a graph to show data or mathematical functions.
Term	Definition
Programming	Writing instructions and code for computers to follow.
Quadratic Function	A function that creates a curved graph and usually contains x^2 .
Range	The set of all possible output values in a function.
Rational Numbers	Numbers that can be written as fractions such as $\frac{1}{2}$ or $\frac{3}{4}$.
Scatter Graph	A graph that displays points to show relationships between values.
Simulation	A computer-generated model of a real-world system or process.

Smart System	A digital system that can make decisions automatically using sensors and logic.
Software	Programs and applications used on a computer.
Truth Table	A table showing all possible inputs and outputs of a Boolean expression.
Two's Complement	A method used in binary to represent negative numbers.
Variable	A symbol or value that can change in a function or equation.
XOR Gate	A logic gate that gives an output of 1 only when the inputs are different.
XNOR Gate	A logic gate that gives an output of 1 when both inputs are the same.
Term	Definition
Y-Intercept	The point where a graph crosses the y-axis.

11. Conclusion & Summary

Key Takeaways from the Booklet

In this booklet, I learned many important topics related to computational thinking, functions, calculations, and Boolean logic. These topics helped me understand how computers solve problems and how digital systems work in real life.

One of the main things I learned was how calculations are used in computing. Computers use calculations for processing data, solving problems, creating programs, and controlling systems. I also learned

about different number systems such as binary, decimal, and hexadecimal. These number systems are very important because computers store and process all information using binary values.

Another important topic in this booklet was power notation. I learned how powers of two are used in computing, especially for memory sizes and binary calculations. I also practiced converting numbers between decimal, binary, and hexadecimal systems. These conversions helped me understand how computers represent data internally.

I also learned about rounding errors and how computers sometimes cannot store decimal numbers exactly. This can create small calculation mistakes in programs and systems. Understanding rounding errors is important because it helps programmers create more accurate software and calculations.

The booklet also explained functions and how they are used in computing. I learned about:

- Linear functions
- Quadratic functions
- Domain and range
- Plotting graphs
- Finding equations from graphs

Functions are useful because they help control movement, positioning, and animations in software and games. Linear functions are used for straight movement, while quadratic functions are useful for curved motion such as jumping in games.

Another important part of this booklet was Boolean logic and logic gates. I learned how logic gates such as:

- AND
- OR

- NOT
- NAND
- NOR
- XOR
- XNOR

are used in digital systems to make decisions. Truth tables helped me understand how inputs and outputs work in Boolean expressions.

I also learned how complex Boolean operations can be combined together in series to create more advanced systems. These systems are used in:

- Security systems
- Smart homes
- Robots
- Game controls
- Automation systems

By completing this booklet, I improved my understanding of computational thinking and problem-solving skills.

The Importance of Computational Thinking in IT Careers

Computational thinking is one of the most important skills in the IT industry. It helps people solve problems in a logical and organised way. Instead of solving a large problem all at once, computational thinking breaks problems into smaller steps that are easier to manage.

These skills are important in many IT careers such as:

- Software development

- Cybersecurity
- Game development
- Data analysis
- Robotics
- Web development
- Artificial intelligence
- Network engineering

Programmers and IT professionals use computational thinking every day to design systems, create algorithms, and solve technical problems.

Functions and Boolean logic are also very important in modern technology. Almost every digital device uses these concepts. Mobile phones, computers, gaming systems, websites, smart devices, and robots all depend on mathematical functions and Boolean operations to work correctly.

For example:

- Games use functions for movement and physics.
- Security systems use Boolean logic for access control.
- Smart homes use logic gates to automate lights and sensors.
- Apps use functions to create animations and interactive designs.

Learning these skills also helps students prepare for future studies and careers in technology. It improves logical thinking, problem-solving, creativity, and programming ability.

Overall, this booklet helped me understand how computing systems use mathematics and logic together to create reliable and intelligent digital technology. These concepts are very useful in the modern world and are important for anyone interested in IT and computing careers.

12. References

<https://youtu.be/k4juL-gT1X0?si=Y7j01pEIN5kGPE-R>

<https://youtu.be/JQBRzsPhw2w?si=6aEgAmHtLAHAMy->

https://en.wikipedia.org/wiki/Logic_gate

<https://www.bbc.co.uk/bitesize/guides/zkklkw6f/revision/2>

<https://www.geeksforgeeks.org/digital-logic/logic-gates/>

[https://en.wikipedia.org/wiki/Function_\(mathematics\)](https://en.wikipedia.org/wiki/Function_(mathematics))

<https://www.bbc.co.uk/bitesize/guides/zqh49j6/revision/5>

[https://en.wikipedia.org/wiki/Function_\(computer_programming\)](https://en.wikipedia.org/wiki/Function_(computer_programming))

function

/ˈfʌŋ(k)ʃn/

A **function** is a relationship, rule, or purpose that dictates how inputs map to outputs or how something operates. The concept spans multiple fields:  Wikipedia +1

1. Mathematics

In math, a function is a rule that assigns exactly one output to each input.  Wikipedia +1

- **Notation:** Expressed as $f(x)$, where f is the function and x is the input.
- **Example:** $f(x) = 2x + 1$. If the input $x = 3$, the output is $f(3) = 2(3) + 1 = 7$.
- **Core Rule:** Every valid input must have only one output.  YouTube · TabletClass Math +4

2. Computer Science & Programming

In programming, a function is a reusable block of code designed to perform a specific action or calculation.  BBC

- It takes optional inputs (called arguments or parameters), runs its defined instructions, and typically returns an output.
- **Benefits:** Breaks complex tasks into smaller pieces, prevents code repetition, and makes programs easier to read.  BBC +4

3. General Use & Biology

- **Purpose:** The natural or intended purpose of a person, object, or system. For example, the *function* of the heart is to pump blood.
- **Event:** An official ceremony, social gathering, or formal event.  Cambridge Dictionary +2

AI responses may include mistakes. [Learn more](#)





A Boolean logic gate is a fundamental physical or theoretical device that performs logical operations on one or more binary inputs (0 or 1, representing False or True) to produce a single output based on Boolean algebra. These gates are the building blocks of digital circuits, forming complex computational systems. [Wikipedia +2](#)

Key Aspects of Boolean Logic Gates:

- **Fundamental Types:** The three basic gates are AND, OR, and NOT.
- **Functions:**
 - **AND Gate:** Outputs 1 if both inputs are 1.
 - **OR Gate:** Outputs 1 if at least one input is 1.
 - **NOT Gate (Inverter):** Reverses the input; 1 becomes 0, 0 becomes 1.
 - **NAND/NOR/XOR:** Combinations of basic gates creating specific outputs (e.g., NAND is NOT + AND).
- **Truth Tables:** These are used to map every possible input combination to its corresponding output.
- **Representation:** Logic gates are represented by standard symbols in circuit diagrams and Boolean algebra expressions (e.g., $Q = A \wedge B$). [Basic Electronics Tutorials +5](#)

These gates are built from transistors acting as switches, which are essential for processing binary information in computers. [YouTube - Spanning Tree +1](#)