

UP Board Class 10 Computer - 836 - 2025 Question Paper with Solutions

Time Allowed :3 Hours	Maximum Marks :70	Total Questions :14
-----------------------	-------------------	---------------------

General Instructions

Read the following instructions very carefully and strictly follow them:

1. All questions are compulsory.
2. This question paper has two parts – Part-A and Part-B.
3. Part-A consists of 20 multiple choice questions of one mark each that have to be answered on OMR Answer Sheet by darkening completely the correct circle with blue or black ballpoint pen.
4. For Part-A read the question-wise instructions and mark your answer on the given OMR Answer Sheet only. Do not cut, use eraser or whitener on the OMR Answer Sheet after answering.
5. Marks of the questions are mentioned against them.
6. Part-B consists of descriptive type questions of 50 marks.
7. All the questions of Part-B are to be attempted all at a time.
8. Start from the first question and go up to the last question. Do not waste your time on the question you cannot solve.

Part - A

1. Which data type is used to store decimal numbers ?

- (A) int
- (B) float
- (C) char
- (D) byte

Correct Answer: (B) float

Solution:

Step 1: Concept of Data Types

Data types in programming define what kind of values a variable can hold. For example, integers, floating-point numbers, characters, etc. The type of value a variable holds influences how

it is stored in memory.

Step 2: Explanation of the Options

Now, let's examine the given options:

- (A) **int**: The **int** data type is used to store integer values, which are whole numbers without decimal points. For example, 5, 100, or -23.
- (B) **float**: The **float** data type is used for storing decimal numbers (floating-point numbers) such as 3.14, -0.5, or 2.718.
- (C) **char**: The **char** data type is used for storing a single character, such as 'a', 'Z', or '9'.
- (D) **byte** is not a primitive data type in C, although it is used in languages like Java.

Thus, the correct data type to store decimal numbers is **float**.

Step 3: Final Answer

The correct answer is option (B), which is **float**.

💡 Quick Tip

For decimal values, use **float** or **double** in C. Remember, **double** is used for higher precision decimal values than **float**.

2. In C Programming %d format specifier is used for which data type ?

- (A) float
- (B) char
- (C) int
- (D) double

Correct Answer: (C) int

Solution:

Step 1: Understanding Format Specifiers

Format specifiers in C are used to specify the data type of variables being read or printed using functions like **printf()** and **scanf()**. Each data type has a specific format specifier.

Step 2: Explanation of Format Specifiers

Let's review the format specifiers for different data types:

- For **int** (integer), the format specifier is **%d** or **%i**.
- For **float** (single-precision floating-point number), the format specifier is **%f**.
- For **char** (character), the format specifier is **%c**.

- For `double` (double-precision floating-point), the format specifier is `%lf`.

As per the question, `%d` is used for `int` data type, representing signed decimal integers.

Step 3: Final Answer

The `%d` format specifier is used for the `int` data type in C programming.

💡 Quick Tip

Keep a cheat sheet for format specifiers (`%d`, `%f`, `%c`, `%s`, `%lf`, `%p`) to make it easier to recall during exams or coding challenges.

3. Which keyword defines a constant in C?

- (A) `var`
- (B) `constant`
- (C) `const`
- (D) `let`

Correct Answer: (C) `const`

Solution:

Step 1: Understanding Constants in C

In C programming, constants are values that cannot be modified after their initial assignment. Constants are critical for preventing accidental changes to critical values in the program. C provides specific ways to declare constants, one of which involves using the `'const'` keyword.

Step 2: Evaluating Each Option

Let's analyze the provided options:

- **(A) `var`:** The keyword `'var'` is used for variable declaration in some languages like JavaScript, but it is not a valid keyword for declaring constants in C.
- **(B) `constant`:** While this word describes what a constant is, it is not a valid keyword in C for defining constants.
- **(C) `const`:** This is the correct keyword in C. By using the `'const'` keyword, you can define a constant. For example: `'const int MAX_SIZE = 100;'` *This means that 'MAX_SIZE' cannot be modified.*
- **(D) `let`:** `'let'` is used in modern JavaScript for variable declaration but is not a keyword in C.

Another way to define constants in C is by using the ‘define’ preprocessor directive (e.g., ‘define PI 3.14159’), but the ‘const’ keyword is the standard way for declaring constants in C code.

Step 3: Conclusion

The keyword ‘const’ is used in C to define constants, ensuring that the value cannot be altered after its initialization. Thus, the correct answer is:

const

Quick Tip

Remember the two common methods for defining constants in C: the ‘const’ keyword for type-checked constants and the ‘define’ preprocessor directive for simple symbolic constants. Both have their uses, but ‘const’ provides additional type safety.

4. Which operator is used for arithmetic addition in C?

- (A) *
- (B) /
- (C) -
- (D) +

Correct Answer: (D) +

Solution:

Step 1: Overview of Arithmetic Operators in C

In C programming, arithmetic operators are used to perform mathematical operations on numbers. These operators include addition, subtraction, multiplication, division, and modulus.

Step 2: Detailed Explanation of Each Option

Let’s review the options:

- **(A) * (Multiplication):** This operator is used for multiplication of two operands. For example, ‘5 * 3’ results in ‘15’.
- **(B) / (Division):** This operator is used to divide the first operand by the second. For example, ‘6 / 3’ results in ‘2’.
- **(C) - (Subtraction):** This operator subtracts the second operand from the first. For example, ‘5 - 3’ results in ‘2’.
- **(D) + (Addition):** The ‘+’ operator is used to add two operands. For example, ‘5 + 3’ results in ‘8’. This is the correct operator for arithmetic addition in C.

Step 3: Conclusion

The correct operator for performing arithmetic addition is the '+' operator. Thus, the correct answer is:

+

💡 Quick Tip

When using division with integers in C, be cautious as it performs integer division, which truncates any decimal portion. For example, '5 / 2' results in '2', not '2.5'. To ensure a floating-point result, at least one operand must be a floating-point number.

5. Which library file defines the printf and scanf functions ?

- (A) math.h
- (B) string.h
- (C) stdio.h
- (D) stdlib.h

Correct Answer: (C) stdio.h

Solution:

Step 1: Understanding the Concept:

In C, a library file (or header file) contains the declarations of standard functions and macros that can be included in a program using the `#include` directive. This allows programmers to use pre-written functions without having to write them from scratch.

Step 2: Detailed Explanation:

Let's look at the purpose of each header file listed:

- **(A) math.h:** Declares mathematical functions like `sqrt()` (square root), `pow()` (power), `sin()` (sine), etc.
- **(B) string.h:** Declares functions for string manipulation, such as `strcpy()` (string copy), `strlen()` (string length), and `strcmp()` (string compare).
- **(C) stdio.h:** Stands for **Standard Input/Output**. It declares functions for input and output operations, most notably `printf()` (formatted output to console) and `scanf()` (formatted input from console), as well as file operations like `fopen()` and `fclose()`.
- **(D) stdlib.h:** Stands for **Standard Library**. It declares functions for general utilities, including memory allocation (`malloc()`, `free()`), random number generation (`rand()`),

and string conversions (`atoi()`).

The functions `printf` and `scanf` are fundamental for standard input and output operations.

Step 3: Final Answer:

The declarations for `printf` and `scanf` are located in the `stdio.h` library file.

💡 Quick Tip

Almost every C program you write that interacts with the user via the console will start with `#include <stdio.h>`. Remembering that "stdio" stands for "Standard Input/Output" is an easy way to associate it with functions like `printf` and `scanf`.

6. What is the syntax used to initialize a variable in C ?

- (A) `int a == 5;`
- (B) `int a = 5;`
- (C) `int a := 5;`
- (D) `int a -> 5;`

Correct Answer: (B) `int a = 5;`

Solution:

Step 1: Understanding the Concept:

Variable initialization is the process of assigning an initial value to a variable at the time it is declared. The syntax in C involves specifying the data type, the variable name, the assignment operator, and the value.

Step 2: Detailed Explanation:

Let's analyze the syntax of each option:

- **(A) `int a == 5;`** The `==` operator is the **equality comparison operator**, not the assignment operator. This syntax is incorrect for initialization and would be used inside a conditional statement like `if (a == 5)`.
- **(B) `int a = 5;`** This is the correct syntax. It declares a variable named `a` of type `int` and initializes it with the value 5 using the **assignment operator** `=`.
- **(C) `int a := 5;`** The `:=` operator is used for declaration and initialization in other languages like Go or Pascal. It is not valid C syntax.

- **(D) `int a -> 5;`**: The `->` operator is the **arrow operator** (structure pointer operator), used to access members of a struct or union through a pointer. It is completely incorrect for variable initialization.

Step 3: Final Answer:

The correct syntax to declare and initialize a variable in C is `int a = 5;`.

💡 Quick Tip

A very common bug for beginners is confusing the assignment operator `=` with the equality comparison operator `==`. Always remember: use a single equals sign (`=`) for assigning a value and a double equals sign (`==`) for comparing two values.

7. What do you call a loop that never ends ?

- (A) for loop
- (B) while loop
- (C) infinite loop
- (D) do-while loop

Correct Answer: (C) infinite loop

Solution:

Step 1: Understanding the Concept:

A loop is a control flow statement that allows code to be executed repeatedly. Normally, a loop has a termination condition that eventually becomes false, causing the loop to end. A loop that lacks a proper termination condition, or whose condition never becomes false, will run forever.

Step 2: Detailed Explanation:

- **(A) for loop**, **(B) while loop**, and **(D) do-while loop** are types of loop structures available in C. Any of these can be written in a way that causes them to run forever, but the names themselves refer to the structure, not the behavior.
- **(C) infinite loop** is the specific term used to describe the behavior of a loop that executes indefinitely. This happens when the loop's controlling condition always evaluates to true.

Examples of infinite loops:

```
// Using a while loop
while (1) {
    // This code runs forever
}
```

```
// Using a for loop
for (;;) {
    // This code runs forever
}
```

Step 3: Final Answer:

A loop that never ends is called an **infinite loop**.

💡 Quick Tip

While often a bug, infinite loops are sometimes created intentionally, especially in embedded systems or server applications where the program is meant to run continuously, waiting for events or processing data. For example, the main event loop in a GUI application is a form of an infinite loop.

8. What is the purpose of the main function in C ?

- (A) To start the program
- (B) To store data
- (C) To detect errors
- (D) To manage files

Correct Answer: (A) To start the program

Solution:

Step 1: Understanding the Concept:

Every C program must have a `main()` function. It is a special function that serves as the entry point for the program's execution. When you run a compiled C program, the operating system starts the execution from the first line of code inside the `main()` function.

Step 2: Detailed Explanation:

Let's analyze the given options:

- **(A) To start the program:** This is the correct purpose. The `main()` function is where the program execution begins. All other functions are called either directly or indirectly from `main()`.

- **(B) To store data:** Data is stored in variables, which can be defined inside `main()` or any other function, but the primary purpose of the function itself is not data storage.
- **(C) To detect errors:** Error detection is a part of programming logic and can be implemented in any function using conditional statements or specific error-handling routines. It is not the unique purpose of `main()`.
- **(D) To manage files:** File management is done using functions from the `stdio.h` library (like `fopen()`, `fclose()`). While these functions can be called from `main()`, file management is not its fundamental role.

Step 3: Final Answer:

The fundamental purpose of the `main` function in a C program is to serve as the starting point for program execution.

💡 Quick Tip

Remember the two standard signatures for the `main` function: `int main(void)` for programs that don't take command-line arguments, and `int main(int argc, char *argv[])` for programs that do. The `int` return type is used to return an exit status to the operating system, where 0 typically indicates successful execution.

9. Which operator is used to denote a pointer ?

- (A) `&`
- (B) `*`
- (C) `%`
- (D) `@`

Correct Answer: (B) `*`

Solution:

Step 1: Understanding the Concept:

Pointers are special variables in C that store memory addresses as their values. They are a fundamental part of the language and are denoted and manipulated using specific operators.

Step 2: Detailed Explanation:

The asterisk symbol `*` has a dual role in the context of pointers:

1. **Declaration:** When used in a variable declaration, it denotes that the variable is a pointer. For example, `int *ptr;` declares `ptr` as a pointer to an integer.

2. **Dereferencing:** When used with an already declared pointer variable in an expression, it acts as the dereference operator (or indirection operator). It accesses the value stored at the memory address the pointer is pointing to. For example, `value = *ptr;`.

Let's review the other operators:

- **(A) & (Ampersand):** This is the "address-of" operator. It returns the memory address of a variable (e.g., `ptr = &var;`).
- **(C) % (Percent):** This is the modulo operator, used for finding the remainder of a division.
- **(D) @ (At sign):** This symbol is not a standard operator in C.

The question asks which operator is used to *denote* a pointer, which directly refers to its use in declaration.

Step 3: Final Answer:

The asterisk `*` operator is used to declare a pointer variable.

 Quick Tip

A simple way to remember the pointer operators: `&` means "address of" and `*` means "value at address". For example, if `int x = 10;` and `int *p = &x;`, then `p` holds the address of `x`, and `*p` holds the value at that address, which is 10.

10. Which library file defines mathematical functions (like `sqrt`) ?

- (A) `math.h`
- (B) `stdio.h`
- (C) `stdlib.h`
- (D) `string.h`

Correct Answer: (A) `math.h`

Solution:

Step 1: Understanding the Concept:

The C standard library is organized into different header files, each containing declarations for a group of related functions. To use a specific function, the corresponding header file must be

included in the source code.

Step 2: Detailed Explanation:

Let's examine the purpose of each header file:

- **(A) `math.h`:** This header file contains declarations for common mathematical functions. This includes trigonometric functions (`sin`, `cos`), logarithmic functions (`log`), power functions (`pow`), and the square root function (`sqrt`).
- **(B) `stdio.h`:** Declares standard input/output functions like `printf` and `scanf`.
- **(C) `stdlib.h`:** Declares general utility functions for memory management, string conversion, etc.
- **(D) `string.h`:** Declares functions for string manipulation like `strcpy` and `strlen`.

The question specifically asks about mathematical functions like `sqrt`.

Step 3: Final Answer:

The `math.h` library file contains the declarations for mathematical functions such as `sqrt()`.

💡 Quick Tip

When using functions from `math.h` and compiling with GCC or Clang on a Linux system, you often need to explicitly link the math library. This is done by adding the `-lm` flag at the end of the compilation command (e.g., `gcc my_program.c -o my_program -lm`).

11. Which of the following is a common application of AI ?

- (A) Data entry
- (B) Email-writing
- (C) Image recognition
- (D) Hardware installation

Correct Answer: (C) Image recognition

Solution:

Step 1: Understanding the Concept:

Artificial Intelligence (AI) involves creating systems that can perform tasks that typically require human intelligence. This includes learning, reasoning, problem-solving, perception, and

language understanding. Common applications leverage these capabilities.

Step 2: Detailed Explanation:

Let's evaluate the options:

- **(A) Data entry:** This is typically a manual, repetitive task that does not require complex reasoning or learning, though AI (specifically OCR - Optical Character Recognition) can assist in automating parts of it. However, it's not considered a core AI application in itself.
- **(B) Email-writing:** While modern AI can assist with email writing (e.g., predictive text, grammar correction, generating drafts), the act itself is a human task. AI's role is assistive.
- **(C) Image recognition:** This is a classic and widespread application of AI, specifically in the fields of computer vision and machine learning. AI systems are trained on vast datasets of images to identify and classify objects, faces, scenes, and text within them. Examples include facial recognition on smartphones, self-driving car sensors, and medical image analysis.
- **(D) Hardware installation:** This is a physical, manual task that requires mechanical skill and knowledge of physical components. It does not fall under the domain of AI.

Among the choices, image recognition is the most prominent and defining application of modern AI.

Step 3: Final Answer:

Image recognition is a common and fundamental application of Artificial Intelligence.

💡 Quick Tip

When identifying AI applications, look for tasks that involve pattern recognition, prediction, learning from data, or understanding unstructured data like images, text, and speech. Tasks that are purely manual, physical, or based on simple, fixed rules are less likely to be core AI applications.

12. What does NLP stand for in the context of AI ?

- (A) National Language Processing
- (B) Natural Language Processing
- (C) Neural Language Processing
- (D) Native Language Processing

Correct Answer: (B) Natural Language Processing

Solution:

Step 1: Understanding the Concept:

NLP is a major subfield of Artificial Intelligence (AI) that focuses on the interaction between computers and humans using natural language. The goal is to enable computers to read, understand, interpret, and generate human languages in a way that is valuable.

Step 2: Detailed Explanation:

The acronym NLP stands for **Natural Language Processing**.

- **Natural:** Refers to human languages (like English, Spanish, Hindi) as they have evolved naturally, as opposed to constructed languages like programming languages or mathematical notations.
- **Language:** Refers to the system of communication used by humans.
- **Processing:** Refers to the computational analysis and manipulation of language data.

The other options are incorrect interpretations. While "Neural Language Processing" might sound plausible because neural networks are heavily used in modern NLP, the established term for the entire field is Natural Language Processing.

Step 3: Final Answer:

In the context of AI, NLP stands for Natural Language Processing.

 Quick Tip

Associate NLP with real-world applications you use every day: virtual assistants (Siri, Alexa), machine translation (Google Translate), spam filters in your email, and sentiment analysis on social media. This makes the term easier to remember.

13. Which AI technique is used for decision-making based on past data ?

- (A) Machine learning
- (B) Database management
- (C) Network analysis
- (D) Manual programming

Correct Answer: (A) Machine learning

Solution:

Step 1: Understanding the Concept:

The question describes a process where a system learns from historical information (past data) to make predictions or decisions about new, unseen data. This is the core definition of a specific AI technique.

Step 2: Detailed Explanation:

Let's analyze the options:

- **(A) Machine learning:** This is a subset of AI that involves creating algorithms and statistical models that enable computer systems to learn and improve from experience (data) without being explicitly programmed. The system identifies patterns in the past data and uses these patterns to make decisions. This perfectly matches the question's description.
- **(B) Database management:** This involves organizing, storing, and retrieving data from a database. While it provides the data, it is not the technique used for learning or decision-making from that data.
- **(C) Network analysis:** This field deals with studying the relationships and structures of networks (e.g., social networks, computer networks). While it can use AI techniques, it is a separate domain.
- **(D) Manual programming:** This is the traditional approach where a programmer writes explicit, step-by-step rules for the computer to follow. It is the opposite of learning from data.

Step 3: Final Answer:

Machine learning is the AI technique used for decision-making based on past data.

💡 Quick Tip

The key phrase to look for is "learning from data" or "based on past data/experience." Whenever a question mentions a system that improves its performance by analyzing historical information, the answer is almost always related to machine learning.

14. What is a neural network inspired by ?

- (A) Human brain
- (B) Internet
- (C) Solar system

(D) Nervous system of insects

Correct Answer: (A) Human brain

Solution:

Step 1: Understanding the Concept:

Artificial Neural Networks (ANNs), often just called neural networks, are a cornerstone of modern AI and deep learning. Their fundamental structure and functioning are modeled after a biological system.

Step 2: Detailed Explanation:

- **(A) Human brain:** This is the correct answer. Neural networks are computational models inspired by the structure of the human brain. The brain consists of billions of interconnected neurons that transmit signals. An ANN consists of interconnected nodes (artificial neurons) organized in layers. Each connection has a weight that is adjusted during the learning process, similar to how synaptic strengths are believed to change in a biological brain.
- **(B) Internet:** The internet is a network of computers, but its structure and principles of data packet routing are different from the processing model of a neural network.
- **(C) Solar system:** The solar system is a gravitational system of celestial bodies, which has no structural resemblance to a neural network.
- **(D) Nervous system of insects:** While this is a biological neural network, the primary and most commonly cited inspiration for ANNs is the more complex and powerful human brain. The human brain serves as the high-level conceptual model.

Step 3: Final Answer:

Neural networks are computing systems inspired by the biological neural networks that constitute the human brain.

💡 Quick Tip

The word "neural" in "neural network" is a direct reference to the "neurons" in the brain. Making this simple word association is the easiest way to remember the inspiration behind this AI concept.

15. Which of the following is a type of AI ?

- (A) Narrow AI
- (B) General AI
- (C) Super intelligent AI
- (D) All of these

Correct Answer: (D) All of these

Solution:

Step 1: Understanding the Concept:

Artificial Intelligence is often categorized into different types based on its capabilities and level of intelligence, particularly in comparison to human intelligence.

Step 2: Detailed Explanation:

The main theoretical and practical types of AI are:

- **(A) Narrow AI (ANI - Artificial Narrow Intelligence):** Also known as Weak AI. This type of AI is designed and trained for a specific task. It operates within a limited context and cannot perform tasks beyond its designated field. All currently existing AI, from chess-playing programs to Siri and self-driving cars, is Narrow AI.
- **(B) General AI (AGI - Artificial General Intelligence):** Also known as Strong AI. This is a theoretical form of AI where a machine would have an intelligence equal to humans. It would have a self-aware consciousness and the ability to solve problems, learn, and plan for the future. AGI does not yet exist.
- **(C) Super intelligent AI (ASI - Artificial Superintelligence):** This is a hypothetical AI that would surpass human intelligence and ability across virtually every field. It would be more creative, have better social skills, and possess a general wisdom far beyond that of any human.

Since Narrow AI, General AI, and Super intelligent AI are all recognized classifications within the field of artificial intelligence, all the given options are correct.

Step 3: Final Answer:

All the listed options—Narrow AI, General AI, and Super intelligent AI—are types of AI. Therefore, the correct answer is "All of these".

💡 Quick Tip

Remember the progression: **Narrow** (what we have today, task-specific) → **General** (the goal, human-level intelligence) → **Superintelligent** (the future possibility, smarter than humans).

16. What is the full form of UAV ?

- (A) Unmanned Aerial Vehicle
- (B) Unmanned Artificial Vehicle
- (C) United Aerial Vehicle
- (D) Universal Aerial Vehicle

Correct Answer: (A) Unmanned Aerial Vehicle

Solution:

Step 1: Understanding the Concept:

UAV is a widely used acronym in aviation, military, and technology to describe a specific type of aircraft. The acronym itself breaks down into its constituent parts, defining the object.

Step 2: Detailed Explanation:

Let's break down the correct full form:

- **U - Unmanned:** This means the aircraft operates without a human pilot on board. It can be controlled remotely by a pilot on the ground or fly autonomously based on a pre-programmed flight plan or AI.
- **A - Aerial:** This signifies that the vehicle operates in the air; it is an aircraft.
- **V - Vehicle:** This refers to it being a machine used for transporting payload (like cameras or sensors) or for other purposes.

Combining these, UAV stands for **Unmanned Aerial Vehicle**. The term "drone" is a more common, popular synonym for a UAV. The other options use incorrect words like "Artificial," "United," or "Universal."

Step 3: Final Answer:

The full form of UAV is Unmanned Aerial Vehicle.

 Quick Tip

To easily remember the full form of UAV, think about what a drone is: it's a **Vehicle** that flies in the **Air (Aerial)** and has no pilot inside (**Unmanned**).

17. Which component is essential for a drone to fly ?

- (A) Propellers
- (B) Landing gear
- (C) Camera
- (D) LED lights

Correct Answer: (A) Propellers

Solution:

Step 1: Understanding the Concept:

For a multirotor drone (the most common type) to fly, it needs a mechanism to generate lift, which is the upward force that counteracts gravity. This is accomplished by a key component of its propulsion system.

Step 2: Detailed Explanation:

Let's analyze the role of each component:

- **(A) Propellers:** These are the rotating blades connected to the motors. As they spin, they push air downwards, which, according to Newton's third law of motion, generates an equal and opposite upward force called thrust or lift. This lift is what allows the drone to take off and fly. Without propellers, there is no lift, and therefore no flight. This component is absolutely essential.
- **(B) Landing gear:** This is used for safe takeoffs and landings. While important for protecting the drone and its payload, a drone could technically fly without it (though landing would be difficult). It is not essential for the act of flying itself.
- **(C) Camera:** A camera is a payload. It is what the drone carries to perform a task (like photography or surveillance), but it is not required for the drone to achieve flight.
- **(D) LED lights:** These are used for orientation (helping the pilot see which way the drone is facing) and visibility, especially at night. They are not essential for generating lift.

Step 3: Final Answer:

Propellers are the essential component required to generate the lift necessary for a drone to fly.

💡 Quick Tip

When answering questions about essential components, think about the most fundamental principles. For flight, the most fundamental principle is generating lift to overcome gravity. In a drone, propellers do this job.

18. What is the purpose of a gimbal on a drone ?

- (A) To provide stability for the camera
- (B) To increase flight speed
- (C) To reduce weight
- (D) To enhance battery life

Correct Answer: (A) To provide stability for the camera

Solution:

Step 1: Understanding the Concept:

A gimbal is a mechanical device that uses motors and intelligent sensors (like gyroscopes and accelerometers) to keep an object, typically a camera, level and steady, isolating it from the motion of the vehicle it's mounted on.

Step 2: Detailed Explanation:

Let's evaluate the purpose described in each option:

- **(A) To provide stability for the camera:** This is the primary function of a gimbal on a drone. Drones are subject to vibrations from motors, wind, and movement (tilting, rolling). The gimbal actively counteracts these movements to keep the camera pointed in a steady direction, resulting in smooth, professional-looking video and sharp photos.
- **(B) To increase flight speed:** A gimbal has no effect on the drone's propulsion system or aerodynamics; it does not increase flight speed. In fact, it adds weight, which can slightly reduce speed and flight time.
- **(C) To reduce weight:** A gimbal is an additional component with its own motors and structure, so it adds weight to the drone, it does not reduce it.
- **(D) To enhance battery life:** Since a gimbal consumes power to operate its motors, it slightly reduces the overall battery life, rather than enhancing it.

Step 3: Final Answer:

The purpose of a gimbal on a drone is to provide stability for the camera, ensuring smooth and

stable footage.

 Quick Tip

Think of a gimbal as a "stabilization platform." It's the technology that separates shaky, amateur footage from the smooth, cinematic shots you see in professional drone videos.

19. Which technology is commonly used for drone navigation ?

- (A) GPS
- (B) Bluetooth
- (C) Infrared
- (D) Wi-Fi

Correct Answer: (A) GPS

Solution:

Step 1: Understanding the Concept:

Drone navigation refers to the drone's ability to determine its position, orientation, and velocity to move along a desired path or hold a specific position. For outdoor operations over significant distances, a global positioning technology is required.

Step 2: Detailed Explanation:

Let's analyze the technologies listed:

- **(A) GPS (Global Positioning System):** This is a satellite-based radionavigation system. A GPS receiver on the drone can determine its precise geographical location (latitude, longitude, and altitude) anywhere on Earth. This is fundamental for many autonomous features, including position hold, return-to-home, and waypoint navigation. It is the most common technology for outdoor navigation.
- **(B) Bluetooth:** This is a short-range wireless technology used for connecting devices over a few meters. It is not suitable for navigation over the distances drones typically fly.
- **(C) Infrared:** Infrared sensors can be used on drones for specific tasks like thermal imaging or obstacle avoidance at close range, but they do not provide geographical location data for navigation.
- **(D) Wi-Fi:** This is used for wireless communication, often to transmit the video feed from the drone's camera to the controller or for short-range flight control. Its range is

limited, and it does not provide absolute positioning like GPS.

Step 3: Final Answer:

GPS is the technology most commonly used for outdoor drone navigation.

💡 Quick Tip

While GPS is key for outdoor navigation, drones also use an IMU (Inertial Measurement Unit), which includes accelerometers and gyroscopes, for short-term stability and orientation. For indoor flight where GPS is unavailable, drones may use vision-based systems (vSLAM) or LiDAR for navigation.

20. Which of the following is a common use of drones in agriculture ?

- (A) Crop monitoring
- (B) Livestock tracking
- (C) Soil analysis
- (D) All of these

Correct Answer: (D) All of these

Solution:

Step 1: Understanding the Concept:

Drones have become a powerful tool in modern agriculture, a field often referred to as "precision agriculture." They provide farmers with aerial data that can be used to make more informed decisions, increase efficiency, and improve crop yields.

Step 2: Detailed Explanation:

Let's examine each of the listed applications:

- **(A) Crop monitoring:** This is one of the most common uses. Drones equipped with multispectral or RGB cameras can fly over large fields to assess crop health, identify areas with pests or disease, monitor irrigation levels, and estimate yields. This allows farmers to apply treatments only where needed.
- **(B) Livestock tracking:** In large ranches, drones can be used to quickly locate and monitor livestock. Drones with thermal cameras can even be used to find animals at night or check for health issues by observing heat signatures.
- **(C) Soil analysis:** Drones can assist in soil analysis by creating 3D maps of the terrain to plan planting patterns and manage water runoff. They can also carry sensors to gather

data about soil properties or identify areas where manual soil samples should be taken.

Since all three are valid and common applications of drones in agriculture, the correct option encompasses all of them.

Step 3: Final Answer:

All the given options—Crop monitoring, Livestock tracking, and Soil analysis—are common uses of drones in agriculture.

💡 Quick Tip

Other key applications of drones in agriculture include crop spraying (applying pesticides or fertilizers with high precision) and planting (using specialized drones to shoot seed pods into the soil). The overarching theme is using aerial data to farm more precisely and efficiently.

Part - B

1. What is a pointer in 'C'?

Solution:

Step 1: Understanding the Concept:

A pointer is a special type of variable in the C programming language that is designed to store the memory address of another variable. Instead of holding a direct value (like an integer or a character), it holds the location where a value is stored in the computer's memory.

Step 2: Key Features and Syntax:

The use of pointers involves three main operations:

1. **Declaration:** A pointer is declared by specifying the data type of the variable it will point to, followed by an asterisk (*).

Syntax: `data_type *pointer_name;`

Example: `int *ptr;` declares a pointer named `ptr` that can store the address of an integer variable.

2. **Initialization (Address-of operator &):** To get the memory address of a variable, the address-of operator (&) is used. This address is then assigned to the pointer.

Example:

```
int var = 10;
int *ptr;
ptr = &var; // ptr now holds the address of var
```

3. **Dereferencing (Indirection operator *)**: To access the value stored at the memory address held by a pointer, the dereference or indirection operator (*) is used.

Example:

```
int value = *ptr; // value will be 10
```

Step 3: Final Answer:

In summary, a pointer in C is a variable that stores the memory address of another variable, allowing for indirect manipulation of data, dynamic memory allocation, and efficient handling of arrays and structures.

💡 Quick Tip

Remember the two key pointer operators: **&** (ampersand) means "address of," and ***** (asterisk) means "value at the address." Mastering this distinction is fundamental to understanding pointers.

2. What is a loop in 'C'?

Solution:

Step 1: Understanding the Concept:

A loop is a control flow statement in C that allows a block of code to be executed repeatedly as long as a specified condition remains true. Loops are used to automate repetitive tasks, making the code more concise and efficient.

Step 2: Types of Loops in C:

C programming provides three types of loops:

1. while loop:

- An entry-controlled loop.
- The condition is checked **before** executing the loop body.
- If the condition is false initially, the loop body will not execute even once.

- **Syntax:** `while (condition) { // code to execute }`

2. do...while loop:

- An exit-controlled loop.
- The condition is checked **after** executing the loop body.
- The loop body is guaranteed to execute at least once.
- **Syntax:** `do { // code to execute } while (condition);`

3. for loop:

- A concise loop structure, also entry-controlled.
- It combines initialization, condition, and increment/decrement in a single line.
- Ideal for situations where the number of iterations is known beforehand.
- **Syntax:** `for (initialization; condition; update) { // code to execute }`

Step 3: Final Answer:

A loop in C is a fundamental programming construct used to repeat a section of code until a specific condition is met. The three types are **while**, **do...while**, and **for** loops.

💡 Quick Tip

Use a **for** loop when you know the number of iterations. Use a **while** loop when the number of iterations is unknown and depends on a condition. Use a **do...while** loop when you need the code to execute at least once.

3. How do you declare a function in 'C'?

Solution:

Step 1: Understanding the Concept:

A function declaration, also known as a function prototype, is a statement that informs the compiler about a function's name, its return type, and the types and number of its parameters. It is essential to declare a function before it is called, especially if the function's definition appears after its call in the source code.

Step 2: Syntax of Function Declaration:

The general syntax for a function declaration in C is:

```
return_type function_name(parameter_type1 parameter_name1,  
parameter_type2 parameter_name2, ...);
```

- **return_type:** The data type of the value that the function returns. If the function does not return any value, the return type is **void**.
- **function_name:** The name of the function.
- **parameter list:** The data types of the arguments passed to the function. Parameter names are optional in the declaration but are good for documentation.

A semicolon at the end is mandatory.

Step 3: Example:

Here is an example of a declaration for a function named **addNumbers** that takes two integers as input and returns an integer sum:

```
int addNumbers(int a, int b);
```

Or, without parameter names:

```
int addNumbers(int, int);
```

This declaration would typically be placed at the top of the C file or in a header file.

💡 Quick Tip

Remember the key difference: a function **declaration** (prototype) ends with a semicolon and tells the compiler *what* the function is. A function **definition** has a code block `{...}` and tells the compiler *how* the function works.

4. How do you define a multi-dimensional array in 'C'?

Solution:

Step 1: Understanding the Concept:

A multi-dimensional array can be thought of as an "array of arrays." The most common type is a two-dimensional (2D) array, which is used to represent a matrix or a table with rows and columns. C allows for arrays with more than two dimensions as well.

Step 2: Syntax for Defining a Multi-dimensional Array:

You define a multi-dimensional array by specifying the data type, a name, and the size of each dimension in square brackets.

General Syntax: `data_type array_name[size1][size2]...[sizeN];`

Example (2D Array):

To define a 2D array of integers named `matrix` with 3 rows and 4 columns, you would write:

```
int matrix[3][4];
```

Step 3: Initialization:

You can initialize a multi-dimensional array at the time of definition. The values for each "inner" array are enclosed in their own curly braces.

Example:

```
int matrix[3][4] = {
    {1, 2, 3, 4},    // Row 0
    {5, 6, 7, 8},    // Row 1
    {9, 10, 11, 12} // Row 2
};
```

This creates a table-like structure in memory, where you can access an element using indices for each dimension, e.g., `matrix[1][2]` would access the value 7.

 Quick Tip

In C, multi-dimensional arrays are stored in "row-major order." This means that all elements of the first row are stored contiguously in memory, followed by all elements of the second row, and so on. Understanding this is crucial for advanced pointer arithmetic with arrays.

5. How do you write an if-else statement in 'C'?

Solution:

Step 1: Understanding the Concept:

The `if-else` statement is a conditional control structure used to execute a block of code if a certain condition is true, and a different block of code if the condition is false. It allows a program to make decisions and follow different paths based on runtime conditions.

Step 2: Syntax:

The basic syntax for an `if-else` statement is as follows:

```
if (condition) {
    // Block of code to be executed if condition is true
} else {
    // Block of code to be executed if condition is false
}
```

- The **condition** is an expression that evaluates to either true (any non-zero value) or false (0).
- The **else** block is optional. You can have a simple **if** statement without it.

Step 3: Example:

Here is a program that checks if a number is even or odd:

```
#include <stdio.h>

int main() {
    int number = 10;

    if (number % 2 == 0) {
        printf("%d is an even number.\n", number);
    } else {
        printf("%d is an odd number.\n", number);
    }

    return 0;
}
```

For more complex conditions, you can use an **if-else if-else** ladder.

💡 Quick Tip

It is a very good practice to always use curly braces `{}` for the **if** and **else** blocks, even if there is only a single statement. This prevents common errors, especially when adding more lines of code later (this is known as the "dangling else" problem).

6. What is a preprocessor directive in 'C'?

Solution:

Step 1: Understanding the Concept:

A preprocessor directive is an instruction for the C preprocessor. The preprocessor is a program that runs **before** the actual compiler. It scans the source code for lines beginning with a hash symbol (`#`) and modifies the code according to these directives. The output of the preprocessor is a modified source file that is then fed to the compiler.

Step 2: Common Preprocessor Directives:

Some of the most frequently used preprocessor directives include:

1. **#include:** This directive is used to include the contents of another file (typically a header file) into the current source file.

Example: `#include <stdio.h>` includes the standard input/output library.

2. **#define:** This directive is used to create macros. A macro is a fragment of code that has been given a name. Whenever the name is used, it is replaced by the contents of the macro. It is commonly used for defining constants.

Example: `#define PI 3.14159`

3. **Conditional Compilation (#ifdef, #ifndef, #if, #endif, #else, #elif):** These directives allow you to include or exclude parts of the code from compilation based on certain conditions. This is useful for writing code that can be compiled for different platforms or with different features.

Example:

```
#ifdef DEBUG
    printf("Debugging information\n");
#endif
```

Step 3: Final Answer:

A preprocessor directive in C is a command starting with `#` that modifies the source code before compilation. Key directives include `#include` for file inclusion, `#define` for creating macros and constants, and directives for conditional compilation.

💡 Quick Tip

Remember the difference between `#include <filename>` and `#include "filename"`. Angle brackets `<>` are used for system header files and search in standard system directories. Double quotes `" "` are used for your own header files and search first in the current directory and then in the standard directories.

7. What is the difference between structure & union in 'C'?

Solution:

Step 1: Understanding the Concept:

Both `struct` (structure) and `union` are user-defined data types in C that allow you to group different types of data together under a single name. However, they differ fundamentally in how they store their members in memory.

Step 2: Key Differences:

The main differences are summarized in the table below:

—1—X—X—

Feature Structure (struct) Union (union)

Memory Allocation The total memory allocated is the **sum** of the sizes of all its members. The total memory allocated is the size of its **largest** member.

Data Storage Each member has its own unique memory location. All members share the **same** memory location.

Value Access You can store and access values in **all members** at any time. You can only store and access a value in **one member** at a time.

Example Use Case Storing related data, like an employee record (name, id, salary). Storing data that is mutually exclusive, where only one type of value is needed at any given moment.

Keyword struct union

Step 3: Code Example:

Consider the following declarations:

```
struct ExampleStruct {
    int i;    // 4 bytes
    char c;   // 1 byte
    float f;  // 4 bytes
}; // Total size ~= 4 + 1 + 4 = 9 bytes (plus padding)
```

```
union ExampleUnion {
    int i;    // 4 bytes
    char c;   // 1 byte
    float f;  // 4 bytes
}; // Total size = size of float = 4 bytes
```

In `ExampleStruct`, you can use `i`, `c`, and `f` simultaneously. In `ExampleUnion`, if you store a value in `i`, it will corrupt the data if you try to read it as `c` or `f`.

💡 Quick Tip

Use a **union** when you need to save memory and you know that the members will not be used at the same time. Use a **struct** in almost all other cases where you need to group related data.

8. How many types of AI are there? Write only names.

Solution:

Step 1: Understanding the Categorization:

Artificial Intelligence (AI) can be classified into different types based on two main criteria: their capabilities (how they compare to human intelligence) and their functionality (how they

operate). The question asks for the names of the types. The most common classification is based on capabilities.

Step 2: Types of AI by Capability:

Based on their level of intelligence and capability, there are three main types of AI:

1. **Artificial Narrow Intelligence (ANI)** or **Weak AI**
2. **Artificial General Intelligence (AGI)** or **Strong AI**
3. **Artificial Superintelligence (ASI)**

Step 3: Brief Explanation:

- **ANI:** AI systems designed for a specific task (e.g., facial recognition, virtual assistants like Siri). All AI that exists today is Narrow AI.
- **AGI:** A theoretical form of AI where a machine would have an intelligence equal to humans, capable of understanding, learning, and applying its knowledge to a wide range of tasks.
- **ASI:** A hypothetical AI that would surpass human intelligence and capabilities in virtually every field.

The question only asks for the names, so the list in Step 2 is the direct answer.

💡 Quick Tip

A simple way to remember the types: **Narrow** is task-specific (today's AI), **General** is human-level (the future goal), and **Superintelligence** is beyond-human (a hypothetical future).

9. What is machine learning in AI?

Solution:

Step 1: Understanding the Concept:

Machine Learning (ML) is a subfield of Artificial Intelligence (AI). It is not AI itself, but rather a set of techniques that allows us to implement AI. The core idea of ML is to create systems that

can learn from data, identify patterns, and make decisions with minimal human intervention.

Step 2: Core Principle:

Instead of being explicitly programmed with a set of rules to perform a task, a machine learning algorithm is "trained" on a large dataset. During training, the algorithm learns the underlying patterns and relationships within the data. Once trained, this model can be used to make predictions or decisions on new, unseen data.

Step 3: Types of Machine Learning:

There are three primary categories of machine learning:

1. **Supervised Learning:** The algorithm learns from a labeled dataset (where each data point is tagged with the correct output). It's used for tasks like classification (e.g., spam detection) and regression (e.g., predicting house prices).
2. **Unsupervised Learning:** The algorithm works with an unlabeled dataset and tries to find patterns and structures on its own. It's used for tasks like clustering (e.g., customer segmentation) and dimensionality reduction.
3. **Reinforcement Learning:** The algorithm learns to make decisions by performing actions in an environment and receiving rewards or penalties. It is commonly used in robotics, game playing (e.g., AlphaGo), and navigation.

Step 4: Final Answer:

Machine learning is a method of achieving AI by training algorithms on data, allowing them to learn and make predictions or decisions without being explicitly programmed for the task.

💡 Quick Tip

Think of it this way: Traditional programming creates results from rules and data. Machine Learning creates rules from data and results. It's about learning the rules automatically.

10. What is a drone?

Solution:

Step 1: Definition:

A drone is an unpiloted aircraft or spacecraft. More formally, it is known as an **Unmanned Aerial Vehicle (UAV)**. Drones are flying robots that can be remotely controlled by a human operator or can fly autonomously through software-controlled flight plans in their embedded

systems, working in conjunction with onboard sensors and a Global Positioning System (GPS).

Step 2: Key Components:

While designs vary, most common multi-rotor drones consist of:

- **Frame:** The main body of the drone.
- **Motors and Propellers:** The propulsion system that generates lift.
- **Flight Controller:** The "brain" of the drone, containing sensors like gyroscopes and accelerometers to maintain stability.
- **Battery:** The power source.
- **Receiver:** To receive signals from the remote controller.
- **Payload:** The equipment carried by the drone for its specific task, such as a camera, sensors, or a delivery package.

Step 3: Common Applications:

Drones were initially developed for military purposes but are now widely used in various civilian roles, including:

- Aerial photography and videography.
- Package delivery and logistics.
- Agriculture (crop monitoring, spraying).
- Infrastructure inspection (bridges, power lines).
- Search and rescue operations.
- Mapping and surveying.

Step 4: Final Answer:

A drone is an unmanned aerial vehicle (UAV) that can be remotely or autonomously controlled to perform a variety of tasks, ranging from photography to industrial inspections.

💡 Quick Tip

While "drone" is the popular term, using its formal name, **Unmanned Aerial Vehicle (UAV)**, or the broader term **Unmanned Aircraft System (UAS)** (which includes the ground station), can be more precise in technical contexts.

11. (a) Write the difference between while & do..while loops in 'C'.

Solution:

Step 1: Understanding the Concept:

Both **while** and **do...while** are iterative control structures in C used to repeat a block of code. The fundamental difference lies in **when** they check their loop condition, which affects whether the loop body is guaranteed to execute at least once.

Step 2: Key Differences:

The following table highlights the main distinctions:

Feature	while loop	do...while loop
Control Type	Entry-Controlled Loop	Exit-Controlled Loop
Condition Check	The condition is checked before the loop body is executed. The condition is checked after the loop body is executed.	
Guaranteed Execution	The loop body may not execute at all if the condition is false initially. The loop body is guaranteed to execute at least once.	
Syntax	<code>while (condition) { ... } do { ... } while (condition);</code>	
Semicolon	No semicolon after the <code>while(condition)</code> . A semicolon is required after the <code>while(condition)</code> .	

Step 3: Example:

```
// while loop example
int i = 5;
while (i < 5) {
    // This block will NOT execute
    printf("In while loop\n");
}

// do...while loop example
int j = 5;
do {
    // This block WILL execute once
    printf("In do-while loop\n");
} while (j < 5);
```

💡 Quick Tip

Use a **do...while** loop when you need to perform an action first and then check if it needs to be repeated. A classic example is a menu-driven program where you display the menu options at least once. For all other conditional looping, a **while** loop is generally preferred.

11. (b) What is file handling in 'C'? Write any two file handling functions in 'C'.

Solution:

Step 1: Understanding the Concept:

File handling in C refers to the set of functions and techniques used to store data permanently in a file on a storage device (like a hard disk). When a C program terminates, the data stored in its variables is lost. File handling provides a way to create, read, write, and update files so that data can persist even after the program has finished running. All file handling operations are performed using functions from the `<stdio.h>` header file.

Step 2: Two File Handling Functions:

Here are two of the most fundamental file handling functions in C:

1. `fopen()`

- **Purpose:** To open a file or create a new one if it doesn't exist. Before any other operation can be performed on a file, it must be opened.
- **Syntax:** `FILE *fopen(const char *filename, const char *mode);`
- **Explanation:** It takes the file name and a mode (like "r" for read, "w" for write, "a" for append) as arguments. It returns a pointer to a `FILE` structure if the file is opened successfully, otherwise it returns `NULL`.
- **Example:** `FILE *fptr = fopen("data.txt", "w");`

2. `fclose()`

- **Purpose:** To close a file that was opened using `fopen()`.
- **Syntax:** `int fclose(FILE *stream);`
- **Explanation:** It takes the file pointer as its argument. Closing a file saves the data, flushes any buffered information to the disk, and releases the resources used by the file stream. It returns 0 on success and EOF (End-Of-File) on failure.
- **Example:** `fclose(fptr);`

💡 Quick Tip

It is crucial to always check if `fopen()` returned `NULL`. If it did, it means the file could not be opened, and attempting any further operations on the null pointer will cause your program to crash. Also, always close every file you open with `fclose()` to prevent data loss and resource leaks.

11. (c) What are arrays? How are arrays declared?

Solution:

Step 1: What are Arrays?

An array is a user-defined data structure in C that can store a fixed-size, sequential collection of elements of the **same data type**. The elements are stored in contiguous memory locations, meaning they are placed one after another in memory. This allows for efficient access to elements using an index.

Key Characteristics of an Array:

- **Homogeneous Data Type:** All elements in an array must be of the same type (e.g., all integers, or all characters).
- **Fixed Size:** The size of an array is determined at compile time and cannot be changed during program execution.
- **Contiguous Memory:** Elements are stored next to each other in memory.
- **Indexed Access:** Elements are accessed using an integer index, which starts from 0. For an array of size `N`, the valid indices are 0 to `N-1`.

Step 2: How are Arrays Declared?

An array is declared by specifying the data type of its elements, a name for the array, and its size in square brackets.

Syntax:

```
data_type array_name[array_size];
```

Explanation:

- `data_type`: Specifies the type of elements the array will hold (e.g., `int`, `float`, `char`).
- `array_name`: The name of the array variable.

- **array_size**: A positive integer constant that specifies the number of elements in the array.

Example:

```
int scores[10];
```

This declaration creates an array named **scores** that can hold 10 integer elements. The elements can be accessed as **scores[0]**, **scores[1]**, ..., up to **scores[9]**.

 Quick Tip

A common mistake is the "off-by-one" error. Always remember that for an array of size N, the last element is at index N-1, not N. Accessing an index outside the valid range (e.g., **scores[10]** in the example above) leads to undefined behavior.

11. (d) Write all types of logical operators in 'C'.

Solution:

Step 1: Understanding Logical Operators:

Logical operators in C are used to perform logical operations on boolean expressions. They are typically used in conditional statements (like **if**, **while**) to combine multiple conditions. The result of a logical operation is either true (represented by 1) or false (represented by 0).

Step 2: Types of Logical Operators:

C has three logical operators:

1. && (Logical AND)

- **Purpose:** Returns true (1) if and only if **both** of its operands are true (non-zero). Otherwise, it returns false (0).
- **Example:** `(x > 5) && (y < 10)` is true only if **x** is greater than 5 AND **y** is less than 10.

2. —— (Logical OR)

- **Purpose:** Returns true (1) if **at least one** of its operands is true (non-zero). It returns false (0) only if both are false.
- **Example:** `(ch == 'a') || (ch == 'e')` is true if the character **ch** is 'a' OR 'e'.

3. ! (Logical NOT)

- **Purpose:** It is a unary operator that reverses the logical state of its operand. If a condition is true, Logical NOT makes it false. If it is false, it makes it true.
- **Example:** `!(x == 0)` is true if `x` is not equal to 0.

💡 Quick Tip

Be careful not to confuse logical operators (`&&`, `||`) with bitwise operators (`&`, `|`). Logical operators evaluate the truth value of entire expressions, while bitwise operators perform operations on the individual bits of their integer operands. Using one in place of the other is a common bug.

11. (e) What is the role of the `break` statement in a `switch` case?

Solution:

Step 1: Understanding the `switch` Statement:

The `switch` statement is a multi-way conditional control statement. It evaluates an expression and compares the result to several `case` labels. When a match is found, the block of code associated with that `case` is executed.

Step 2: The Role of the `break` Statement:

The primary role of the `break` statement within a `switch` block is to **terminate the execution** of the `switch` statement immediately. When a `break` statement is encountered inside a `case`, the program control jumps to the statement immediately following the closing curly brace of the `switch` block.

Step 3: Behavior without `break` (Fall-through):

If a `case` block does not end with a `break` statement, execution will "fall through" to the next `case` block and continue executing its statements. This fall-through behavior continues until a `break` is encountered or the end of the `switch` block is reached.

Example:

```
int day = 2;
switch (day) {
    case 1:
        printf("Monday\n");
        break; // Exits switch after printing
    case 2:
        printf("Tuesday\n");
        // No break here!
```

```

    case 3:
        printf("Wednesday\n");
        break;
    default:
        printf("Another day\n");
        break;
}

```

Output of the above code:

Tuesday

Wednesday

Because there is no **break** in **case 2**, the execution falls through and also executes the code in **case 3**.

💡 Quick Tip

While fall-through is sometimes used intentionally to handle multiple cases with the same code block, it is a frequent source of bugs. As a rule of thumb, always end each **case** block (and the **default** block) with a **break** statement unless you have a specific reason for the fall-through and have commented it clearly.

12. (a) Which type of e-commerce refers to the transactions between consumers & businesses ?

Solution:

Step 1: Understanding E-commerce Models:

E-commerce (electronic commerce) is categorized into several models based on the types of participants involved in the transaction. The main participants are Businesses (B), Consumers (C), and Governments (G).

Step 2: Identifying the Correct Model:

The question asks for the model representing transactions **between consumers and businesses**. This describes a scenario where an individual consumer offers products or services to a business.

This model is known as **Consumer-to-Business (C2B)**.

Step 3: Examples of C2B:

- A freelance graphic designer selling a logo design to a company through a platform like Upwork or Fiverr.

- A blogger earning revenue from a company by placing its ads on their blog (affiliate marketing).
- A photographer selling their photos to a stock photo company like Getty Images or Shutterstock.

In all these cases, the consumer initiates the transaction by providing value to the business.

💡 Quick Tip

Remember the e-commerce models by the direction of the transaction: B2C (Business sells to Consumer, like Amazon), B2B (Business sells to Business, like a manufacturer selling to a retailer), C2C (Consumer sells to Consumer, like eBay), and C2B (Consumer sells to Business).

12. (b) What is an online payment gateway?

Solution:

Step 1: Definition:

An online payment gateway is a technology service used by e-commerce businesses and merchants to accept debit or credit card payments from customers. It acts as a secure intermediary or a "digital cash register" that connects the merchant's website, the customer, and the financial institutions (like banks) involved in the transaction.

Step 2: How it Works:

The process generally involves these steps:

1. A customer places an order on a website and enters their payment card details.
2. The payment gateway securely encrypts this sensitive information and sends it from the browser to the merchant's web server.
3. The gateway then forwards the transaction information to the payment processor of the merchant's acquiring bank.
4. The payment processor routes the transaction to the card association (e.g., Visa, Mastercard), which then forwards it to the customer's issuing bank.

5. The issuing bank checks for sufficient funds and fraud, then sends an approval or decline response back through the chain.
6. The gateway receives this response and relays it to the merchant's website, which then informs the customer whether their payment was successful.

This entire process typically takes only a few seconds.

Step 3: Examples:

Popular examples of payment gateways include PayPal, Stripe, Authorize.Net, and Square.

💡 Quick Tip

Think of a payment gateway as the digital equivalent of a physical point-of-sale (POS) terminal in a retail store. Its primary job is to securely capture payment information and manage the communication between all the financial parties to approve or decline the transaction.

12. (c) Write the main components of a drone.

Solution:

Step 1: Understanding the System:

A drone, or Unmanned Aerial Vehicle (UAV), is a complex system composed of several key components that work together to enable flight, control, and data collection.

Step 2: Main Components:

The main components of a typical multi-rotor drone are:

1. **Frame:** The main structure or chassis of the drone that holds all other components together. It needs to be lightweight yet strong.
2. **Motors:** Drones use brushless motors that spin the propellers. The number of motors usually corresponds to the number of arms (e.g., a quadcopter has four motors).
3. **Propellers:** These are the rotating blades that, when spun by the motors, generate the thrust (lift) required for the drone to fly.
4. **Flight Controller (FC):** This is the "brain" of the drone. It's a circuit board containing a processor, sensors (like a gyroscope and accelerometer), and software that interprets

signals from the remote controller and adjusts motor speeds to keep the drone stable.

5. **Electronic Speed Controllers (ESCs):** Each motor is connected to an ESC. The ESC takes the signal from the flight controller and delivers the precise amount of power required to spin the motor at the desired speed.
6. **Battery:** This is the power source for the drone, typically a rechargeable Lithium Polymer (LiPo) battery.
7. **Transmitter and Receiver:** The transmitter is the remote controller held by the pilot. It sends signals to the receiver on the drone to control its movement.
8. **Payload:** This refers to any equipment the drone carries to perform its mission, such as a camera, gimbal, sensors, or a delivery package.

💡 Quick Tip

To easily remember the core flight components, think of the chain of command: The **Flight Controller** (brain) tells the **ESCs** (muscles) how much power to send from the **Battery** (heart) to the **Motors**, which spin the **Propellers** to create lift.

12. (d) What is machine learning in the context of AI?

Solution:

Step 1: Defining the Relationship:

Machine Learning (ML) is a core **subfield** of Artificial Intelligence (AI). It is not AI itself, but rather a powerful set of methods and techniques used to **achieve** AI. While AI is the broader concept of creating machines that can simulate human intelligence, Machine Learning is the practical application that focuses on algorithms that allow systems to learn from data.

Step 2: The Core Principle of Machine Learning:

The fundamental idea of Machine Learning is to enable a system to learn and improve from experience without being explicitly programmed. Instead of a developer writing a set of hard-coded rules for every possible scenario, they develop an ML algorithm that is "trained" on a large amount of data.

During this training process, the algorithm identifies patterns, learns relationships, and builds a statistical model. This model can then be used to make predictions, classifications, or decisions about new, unseen data.

Step 3: Example:

- **AI Goal:** Create a system that can identify spam emails.
- **Traditional Programming Approach:** Write thousands of rules like "If the email contains 'free money', mark as spam." This is brittle and hard to maintain.
- **Machine Learning Approach:** Train an ML algorithm with thousands of emails that have already been labeled as "spam" or "not spam." The algorithm learns the features (words, sender patterns, etc.) that are characteristic of spam. It can then accurately classify new, incoming emails.

In this context, machine learning is the tool used to build the intelligent spam-filtering feature, which is a part of the broader AI system.

Quick Tip

A simple analogy: AI is the destination (a "smart" machine), while Machine Learning is one of the most important vehicles (a method of learning from data) to get there. Not all AI involves machine learning, but most modern, practical AI applications heavily rely on it.

12. (e) What is Natural Language Processing (NLP) in AI?

Solution:

Step 1: Definition:

Natural Language Processing (NLP) is a specialized field of Artificial Intelligence (AI) that focuses on enabling computers to understand, interpret, manipulate, and generate human language (like English, Hindi, etc.). The "natural" in NLP distinguishes human language from "constructed" languages like computer programming languages.

Step 2: The Goal of NLP:

The ultimate objective of NLP is to bridge the communication gap between humans and computers. It aims to give computers the ability to read, decipher, and make sense of human languages in a valuable way. This is a complex task because human language is often ambiguous, full of context, and has nuanced grammatical rules.

Step 3: Key Tasks in NLP:

NLP involves several sub-tasks, including:

- **Speech Recognition:** Converting spoken language into text.
- **Part-of-Speech Tagging:** Identifying words as nouns, verbs, adjectives, etc.
- **Sentiment Analysis:** Determining the emotional tone (positive, negative, neutral) behind a piece of text.
- **Machine Translation:** Translating text from one language to another (e.g., Google Translate).
- **Named Entity Recognition (NER):** Identifying names of people, organizations, locations, etc., in text.
- **Natural Language Generation (NLG):** Generating human-like text from structured data.

Step 4: Applications:

Common applications of NLP in AI include virtual assistants (Siri, Alexa), chatbots, email spam filters, and auto-correct features.

💡 Quick Tip

Think of NLP as the part of AI that deals with "language." Anytime an AI system is reading, listening, talking, or writing in a human language, it is using Natural Language Processing.

13. (a) How many types of functions are in 'C' ? Explain each type with suitable example.

Solution:

Step 1: Understanding Functions in C:

A function is a self-contained block of code that performs a specific task. Using functions helps in modularizing a program, making it more organized, reusable, and easier to debug. In C, functions can be broadly classified into two main types.

Step 2: Types of Functions:

The two types of functions in C are:

1. Library Functions (or Built-in Functions)

- **Explanation:** These are pre-defined functions that are part of the C standard library. Their declarations are available in header files (e.g., `<stdio.h>`, `<math.h>`, `<string.h>`), which we must include in our program to use them. The actual code for these functions is already compiled and linked to our program by the compiler. We don't need to know their internal implementation; we just need to know how to call them.
- **Example:** To print output to the console, we use the `printf()` function. Its declaration is in the `<stdio.h>` header file.

```
#include <stdio.h>
#include <math.h>

int main() {
    // printf() is a library function
    printf("Hello, World!\n");

    // sqrt() is another library function
    double result = sqrt(16.0);
    printf("Square root of 16 is %.1f\n", result);

    return 0;
}
```

2. User-Defined Functions

- **Explanation:** These are functions created by the programmer to perform specific tasks as per the program's requirements. The programmer defines the function's name, return type, parameters, and the code inside it. This allows for breaking down a large, complex problem into smaller, manageable parts.
- **Example:** We can create a function to add two numbers.

```
#include <stdio.h>

// User-defined function declaration (prototype)
int addNumbers(int a, int b);

int main() {
    int num1 = 10, num2 = 20;

    // Calling the user-defined function
    int sum = addNumbers(num1, num2);

    printf("The sum is: %d\n", sum);
    return 0;
}
```

```
// User-defined function definition
int addNumbers(int a, int b) {
    int result = a + b;
    return result; // Return the sum
}
```

💡 Quick Tip

Remember that every C program has at least one user-defined function: the `main()` function. It's the entry point of the program, and even though it's special, it's still defined by the user.

13. (b) What is intelligent agent ? Explain three applications of AI.

Solution:

Part 1: What is an Intelligent Agent?

Step 1: Definition:

An intelligent agent (IA) is an autonomous entity that perceives its environment through sensors and acts upon that environment through actuators to achieve specific goals. An agent is considered "intelligent" because it acts rationally, meaning it selects the action that is expected to maximize its performance measure, given its perception and any built-in knowledge.

Step 2: Key Properties:

- **Autonomy:** It can operate without direct human intervention.
- **Reactivity:** It perceives and responds to changes in its environment in a timely fashion.
- **Pro-activeness:** It takes initiative and exhibits goal-directed behavior.
- **Social Ability:** It can interact or communicate with other agents (including humans).

A simple example is a thermostat. It senses the room temperature (sensor) and turns the heater on or off (actuator) to maintain a target temperature (goal). A more complex example is a self-driving car, which uses cameras and LiDAR (sensors) to perceive the road and controls steering and acceleration (actuators) to reach a destination safely.

Part 2: Three Applications of AI

Step 1: Application 1 - Virtual Assistants

- **Description:** Virtual assistants like Apple's Siri, Google Assistant, and Amazon's Alexa are AI-powered agents that can understand and respond to human voice commands. They use Natural Language Processing (NLP) to interpret requests and can perform a wide range of tasks, such as setting alarms, playing music, answering questions by searching the internet, and controlling smart home devices.

Step 2: Application 2 - E-commerce Recommendation Engines

- **Description:** Companies like Amazon and Netflix use sophisticated AI and machine learning algorithms to analyze a user's past behavior (e.g., viewing history, purchase history, items browsed). Based on this data, the recommendation engine predicts what other products or movies the user might like and presents them with personalized suggestions. This enhances user experience and drives sales.

Step 3: Application 3 - Autonomous Vehicles (Self-Driving Cars)

- **Description:** Self-driving cars (e.g., from companies like Tesla, Waymo) represent a cutting-edge application of AI. They use a combination of computer vision, sensor fusion (from cameras, radar, LiDAR), and machine learning to perceive their surroundings, identify obstacles, interpret traffic signs, and navigate roads safely without human intervention. The AI system is responsible for making critical real-time decisions about steering, braking, and acceleration.

💡 Quick Tip

The concept of an "agent" is central to modern AI. When thinking about AI applications, try to identify the agent, its sensors, its actuators, and its goal within the environment it operates in.

13. (c) What is e-commerce ? How many types of e-commerce are there? Explain each type in brief.

Solution:

Part 1: What is E-commerce?

Step 1: Definition:

E-commerce, short for electronic commerce, is the buying and selling of goods or services using the internet, and the transfer of money and data to execute these transactions. It encompasses a wide variety of data, systems, and tools for online buyers and sellers, including online shopping and online payment processing. E-commerce has fundamentally changed how businesses

operate and how consumers shop.

Part 2: Types of E-commerce

Step 1: Main Types of E-commerce Models:

While many classifications exist, the four primary types of e-commerce are based on the relationship between the participants in the transaction:

1. Business-to-Consumer (B2C):

- **Explanation:** This is the most common form of e-commerce. It involves businesses selling products or services directly to individual consumers. It's the online equivalent of a traditional retail store.
- **Example:** Purchasing a book from Amazon, buying clothes from a brand's website, or subscribing to Netflix.

2. Business-to-Business (B2B):

- **Explanation:** This model involves transactions between two businesses. One business sells its products or services to another business. These transactions often involve bulk orders, raw materials, or professional services.
- **Example:** A car manufacturer buying steel from a steel company, a firm purchasing office software from Microsoft, or a business buying office furniture online.

3. Consumer-to-Consumer (C2C):

- **Explanation:** This model facilitates transactions between two individual consumers. It typically involves a third-party platform that connects buyers and sellers.
- **Example:** An individual selling their used smartphone to another person on platforms like eBay, OLX, or Facebook Marketplace.

4. Consumer-to-Business (C2B):

- **Explanation:** In this model, the roles are reversed from B2C. Individual consumers create value and sell their products or services to businesses.
- **Example:** A freelance writer providing services to a company via Upwork, a photographer selling stock images to a business, or a social media influencer being paid by a brand to promote a product.

💡 Quick Tip

Other emerging e-commerce models include G2C (Government-to-Citizen), B2G (Business-to-Government), and C2G (Citizen-to-Government), which involve government bodies as one of the transaction parties.

14. What is cyber crime ? How many types of cyber crime are there ? What are the major causes of cyber crime ? Explain with example.

Solution:

Part 1: What is Cyber Crime?

Cyber crime is defined as any criminal activity that involves a computer, a networked device, or a network. While some cyber crimes target computers or devices directly (e.g., with viruses), others use them as tools to commit offenses (e.g., phishing or identity theft). The primary goal is often financial gain, but it can also be driven by other motives like espionage, disruption, or personal revenge.

Part 2: Types of Cyber Crime

Cyber crime is a broad category. Some common types include:

1. **Phishing:** Deceiving victims into revealing sensitive information (like passwords, credit card numbers) by masquerading as a trustworthy entity in an electronic communication, typically an email.
2. **Malware Attacks:** Using malicious software (malware) like viruses, worms, trojans, spyware, and ransomware to disrupt computer operations, gain unauthorized access, or damage a system.
3. **Ransomware:** A type of malware that encrypts a victim's files, making them inaccessible. The attacker then demands a ransom (payment) to restore access.
4. **Identity Theft:** Illegally obtaining and using someone else's personal data (e.g., Social Security number, bank account details) for fraudulent purposes.
5. **Denial-of-Service (DoS) / Distributed Denial-of-Service (DDoS) Attacks:** Flooding a server or network with overwhelming traffic to make it unavailable to its intended users.
6. **Cyberstalking:** Using the internet and other electronic means to stalk or harass an individual, group, or organization.

Part 3: Major Causes of Cyber Crime

The motivations behind cyber crime are diverse:

- **Financial Gain:** This is the most common cause. Criminals engage in activities like online fraud, stealing financial data, and ransomware attacks to make money.
- **Espionage:** Governments or competing corporations may sponsor cyber attacks to steal confidential data, trade secrets, or intellectual property.
- **Activism (Hacktivism):** Hackers may target systems to promote a political or social cause, often by defacing websites or leaking sensitive information.
- **Revenge or Disruption:** Disgruntled employees or individuals may launch attacks to harm a company's reputation or disrupt its operations.
- **Lack of Awareness and Security:** Weak passwords, unpatched software, and lack of user awareness create easy opportunities for criminals to exploit.

Part 4: Example

A clear example of cyber crime is a **phishing attack**.

- **Scenario:** A user receives an email that appears to be from their bank. The email warns of a "security issue" with their account and provides a link to "verify their identity."
- **The Crime:** The link leads to a fake website that looks identical to the real bank's site. When the user enters their username and password on this fake site, the criminals capture the credentials.
- **The Result:** The attackers now have access to the victim's real bank account and can transfer money or commit further fraud. This crime is motivated by financial gain and preys on the user's trust and lack of security awareness.

💡 Quick Tip

The best defense against many common cyber crimes is a combination of technical measures (like antivirus software and firewalls) and user education. Be skeptical of unsolicited emails, use strong, unique passwords, and enable two-factor authentication whenever possible.

14. (OR-1) What is sorting in 'C' ? Write a program in 'C' for Bubble sort & Selection sort.

Solution:

Part 1: What is Sorting?

Sorting is the process of arranging a list or collection of elements in a specific order. The most common orders are numerical order (ascending or descending) and lexicographical (alphabetical) order. Sorting is a fundamental operation in computer science as it often makes other operations, such as searching for an element, significantly more efficient.

Part 2: Program for Bubble Sort

Bubble Sort is a simple sorting algorithm that repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order. The pass through the list is repeated until the list is sorted.

```
#include <stdio.h>

// Function to perform Bubble Sort
void bubbleSort(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n - 1; i++) {
        // Last i elements are already in place
        for (j = 0; j < n - i - 1; j++) {
            // Swap if the element found is greater than the next element
            if (arr[j] > arr[j + 1]) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

// Function to print an array
void printArray(int arr[], int size) {
    int i;
    for (i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    int arr[] = {64, 34, 25, 12, 22, 11, 90};
    int n = sizeof(arr) / sizeof(arr[0]);
```

```

printf("Unsorted array: \n");
printArray(arr, n);

bubbleSort(arr, n);

printf("Sorted array (Bubble Sort): \n");
printArray(arr, n);
return 0;
}

```

Part 3: Program for Selection Sort

Selection Sort is an in-place comparison sorting algorithm. It divides the input list into two parts: a sorted sublist which is built up from left to right at the front of the list, and a sublist of the remaining unsorted elements. Initially, the sorted sublist is empty and the unsorted sublist is the entire input list. The algorithm proceeds by finding the smallest element in the unsorted sublist, exchanging it with the leftmost unsorted element, and moving the sublist boundaries one element to the right.

```

#include <stdio.h>

// Function to perform Selection Sort
void selectionSort(int arr[], int n) {
    int i, j, min_idx, temp;
    // One by one move boundary of unsorted subarray
    for (i = 0; i < n - 1; i++) {
        // Find the minimum element in unsorted array
        min_idx = i;
        for (j = i + 1; j < n; j++) {
            if (arr[j] < arr[min_idx]) {
                min_idx = j;
            }
        }
        // Swap the found minimum element with the first element
        temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}

// Function to print an array (can reuse from above)
void printArray(int arr[], int size) {
    int i;
    for (i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

```

```

int main() {
    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("Unsorted array: \n");
    printArray(arr, n);

    selectionSort(arr, n);

    printf("Sorted array (Selection Sort): \n");
    printArray(arr, n);
    return 0;
}

```

💡 Quick Tip

Bubble Sort and Selection Sort are simple to understand but are inefficient for large datasets, both having an average time complexity of $O(n^2)$. For practical applications, more efficient algorithms like Merge Sort or Quick Sort (with average complexity $O(n \log n)$) are used.

14. (OR-2) What is drone technology ? Explain the advantages of drone technology in today's scenario. Write three major applications of drone technology.

Solution:

Part 1: What is Drone Technology?

Drone technology refers to the science and engineering behind the design, manufacturing, and operation of drones, which are more formally known as Unmanned Aerial Vehicles (UAVs). This technology encompasses the entire system required for a drone to function, including the aircraft itself (with its frame, motors, and propellers), the flight control systems (with onboard computers, sensors, and GPS), the ground-based control station, and the system of communication that connects them. Modern drone technology heavily integrates advancements in robotics, artificial intelligence, and data processing.

Part 2: Advantages of Drone Technology

In today's scenario, drone technology offers several significant advantages:

1. **Enhanced Safety:** Drones can be deployed in hazardous or inaccessible environments, eliminating the risk to human life. This is crucial for tasks like inspecting damaged infrastructure (bridges, power lines), monitoring disaster areas, or performing military reconnaissance.

2. **Cost and Time Efficiency:** Drones can perform many tasks much faster and at a lower cost than traditional methods. For example, surveying a large area of land with a drone takes a fraction of the time and cost of using a manned aircraft or a ground-based team.
3. **High-Quality Data Collection:** Equipped with high-resolution cameras, thermal sensors, or LiDAR, drones can capture detailed aerial data and imagery. This data provides valuable insights for industries like agriculture, construction, and environmental monitoring, leading to better decision-making.
4. **Increased Access and Reach:** Drones can reach remote or difficult-to-access locations easily, making them ideal for delivering medical supplies to rural areas, monitoring wildlife in dense forests, or search and rescue missions.

Part 3: Three Major Applications of Drone Technology

1. Agriculture (Precision Farming):

- **Description:** Drones equipped with multispectral sensors are used to monitor crop health, identify areas affected by pests or drought, and analyze soil conditions. This allows farmers to apply water, fertilizers, and pesticides with high precision, only where needed. This practice increases crop yields, reduces costs, and minimizes environmental impact.

2. Logistics and Delivery:

- **Description:** Companies like Amazon (Prime Air) and Zipline are pioneering the use of drones for last-mile package delivery. Drones can deliver small packages, medical supplies, and food quickly and efficiently, especially in congested urban areas or remote rural locations where traditional delivery methods are slow or difficult.

3. Filmmaking and Media:

- **Description:** Drones have revolutionized aerial cinematography. They provide a low-cost and flexible way to capture stunning, dynamic aerial shots that were previously only possible with expensive helicopters or cranes. This has become standard in movies, documentaries, real estate marketing, and event coverage.

💡 Quick Tip

When discussing drone applications, always think about tasks that are "dull, dirty, or dangerous" for humans. These are often the areas where drone technology provides the most significant benefits by improving safety, efficiency, and data quality.