UP Board Class 12 Computer - 341 - 2025 Question Paper with Solutions

Time Allowed :3 Hours	**Maximum Marks :**70	**Total questions :**35

General Instructions

Instruction:

i) *All* questions are compulsory. Marks allotted to each question are given in the margin.

ii) In numerical questions, give all the steps of calculation.

iii) Give relevant answers to the questions.

iv) Give chemical equations, wherever necessary.

Q1. (a) Who invented OOP?

i) Andrea Ferro

ii) Adele Goldberg

iii) Alan Kay

iv) Dennis Ritchie

Correct Answer: (iii) Alan Kay

Solution:

Step 1: Recall the origin of Object-Oriented Programming (OOP).

OOP was conceptualized in the 1960s and 1970s to handle complex software systems. The main idea was to structure programs using objects, classes, and inheritance.

Step 2: Contribution of Alan Kay.

Alan Kay, a computer scientist, is credited with inventing OOP through the development of the programming language Smalltalk. He emphasized objects as the central element of programming.

Step 3: Elimination of other options.

Andrea Ferro and Adele Goldberg contributed to computing but are not credited with inventing OOP. Dennis Ritchie created C, which is a structured programming language, not OOP.

Final Answer:

Alan Kay

Quick Tip

Remember: Alan Kay + Smalltalk = Birth of OOP.

Q1. (b) Whose abbreviation is JVM?

i) Java Version Machine

ii) Java Virtual Machine

iii) Java Verified Module

iv) None of these

Correct Answer: (ii) Java Virtual Machine

Solution:

Step 1: Expand the abbreviation.

JVM stands for Java Virtual Machine.

Step 2: Purpose of JVM.

It provides a runtime environment to execute Java bytecode. It makes Java platform-independent, as the same bytecode can run on different systems using the JVM.

Final Answer:

Java Virtual Machine

Quick Tip

JVM = The engine that runs Java programs everywhere.

Q1. (c) Which component is used to compile, debug and execute the Java program?

i) JRE

ii) JIT

iii) JDK

iv) JVM

Correct Answer: (iii) JDK

Solution:

Step 1: Role of JDK.

The Java Development Kit (JDK) is a complete software development kit required for compiling, debugging, and running Java programs.

Step 2: Difference from JRE and JVM.

- JRE (Java Runtime Environment) is only for running programs, not compiling.

- JVM (Java Virtual Machine) executes the bytecode, but cannot compile.

- JIT (Just-In-Time compiler) is a part of JVM to speed execution.

Thus, the correct development tool is JDK.

Final Answer:

$$\boxed{\text{JDK}}$$

Quick Tip

Use JDK for developing Java programs; JRE/JVM are only for running them.

Q1. (d) A simple mechanical arm is an example of which generation of robots?

i) First generation

ii) Second generation

iii) Third generation

iv) Fourth generation

Correct Answer: (i) First generation

Solution:

Step 1: Robot generation classification.

- First generation: Simple mechanical arms, programmed to perform repetitive tasks.

- Second generation: Equipped with sensors for feedback.

- Third generation: Advanced robots with adaptive learning.

- Fourth generation: Intelligent robots with AI.

Step 2: Identify the category.

A simple mechanical arm falls under the first generation, as it lacks sensors and advanced intelligence.

Final Answer:

$$\boxed{\text{First generation}}$$

Quick Tip

Mechanical arms used in industries (like car manufacturing) are first-generation robots.

Q1. (e) Where is drone used in construction industry?

i) Inventory management

ii) Volumetric measurement

iii) Structural integrity maintenance

iv) All of these

Correct Answer: (iv) All of these

Solution:

Step 1: Uses of drones in construction.

- Drones can monitor and manage inventory effectively.

- They are used in volumetric measurement for land surveying and construction material estimation.

- They help in structural integrity checks by capturing images and detecting cracks.

Step 2: Conclusion.

Since drones are applied in all of the above areas, the answer is "All of these."

Final Answer:

$$\boxed{\text{All of these}}$$

Quick Tip

Drones are multipurpose tools in construction: inventory, measurement, and safety checks.

Q2. (a) What is OOP?

Solution:

Step 1: Understanding OOP.

Object-Oriented Programming (OOP) is a programming paradigm based on the concept of ”objects”, which are instances of classes. It organizes code into manageable sections called objects, where each object is capable of holding data (attributes) and behaviors (methods).

Step 2: Key Concepts of OOP.

OOP emphasizes the following key concepts: - **Encapsulation:** Bundling data with methods that operate on the data, restricting direct access to some of the object's components. - **Inheritance:** A mechanism to define new classes based on existing ones, facilitating code reuse. - **Polymorphism:** The ability to treat objects of different classes in a similar way. - **Abstraction:** Hiding implementation details and showing only the necessary features of the object.

Step 3: Why OOP is important.

OOP promotes code reusability, scalability, and maintainability, which are crucial for modern software development.

Final Answer:

Object-Oriented Programming

Quick Tip

OOP structures code around objects, leading to better code organization and reuse.

Q2. (b) What is JDI?

Solution:

Step 1: Definition of JDI.

The Java Debug Interface (JDI) is part of the Java Platform Debugger Architecture (JPDA). It provides the necessary functionality for debugging Java applications, allowing developers to control the execution flow of a Java program, inspect variables, set breakpoints, and step through code during runtime.

Step 2: How JDI Works.

JDI interacts with the Java Virtual Machine (JVM) to access the runtime data of a Java application. It allows debuggers to interact with JVMs that are running Java programs, making it possible to perform operations such as halting program execution, checking variable values, and more.

Step 3: JDI's Role in Debugging.

JDI provides the interface for developers to programmatically debug their Java applications, which is essential for finding and fixing bugs in a timely manner.

Final Answer:

Java Debug Interface

Quick Tip

JDI is crucial for interacting with JVM and performing runtime debugging.

Q2. (c) What is the purpose of JIT compiler?

Solution:

Step 1: Purpose of JIT Compilation.

JIT stands for Just-In-Time compilation, and it is a part of the JVM. The JIT compiler improves the performance of Java applications by compiling bytecode into native machine code during runtime, rather than at the time of compilation. This process optimizes performance because the program is compiled only when needed, which speeds up execution.

Step 2: How JIT Works.

When a Java program is run, the JVM interprets the bytecode. The JIT compiler identifies "hot spots" (frequently used code) and compiles those into native code. As the program continues, the JIT compiler further optimizes the performance of the application by compiling more frequently used code into machine code.

Step 3: Benefits of JIT.

JIT compilation improves performance by minimizing the overhead associated with interpreting bytecode and directly executing machine code. This makes Java programs run faster, particularly for long-running applications.

Final Answer:

$$\boxed{\text{Just-In-Time Compilation}}$$

Quick Tip

JIT optimizes Java performance by compiling frequently used code into machine code at runtime.

Q2. (d) Name three challenges in robotics.

Solution:

Step 1: Major Challenges in Robotics.

The challenges in robotics can be broadly classified into three categories: - **Cost:** Developing and maintaining robots can be very expensive due to the advanced technology and high precision required. - **Safety:** Ensuring that robots operate safely, especially when interacting with humans, is a major concern. Accidents can lead to injuries or even fatalities. - **Complexity:** Designing robots that can perform complex, dynamic tasks autonomously is a major challenge. Robots often struggle with tasks that require adaptability or understanding of their environment.

Step 2: Conclusion.

These challenges must be addressed to make robots more effective and accessible across industries.

Final Answer:

$$\boxed{\text{Cost, Safety, Complexity}}$$

Quick Tip

Robotics challenges stem from technological, safety, and financial barriers.

Q2. (e) Who made drone technology?

Solution:

Step 1: Origin of Drone Technology.

Drone technology (or Unmanned Aerial Vehicles - UAVs) was developed by several inventors and organizations over time. Military applications were among the earliest uses of UAVs, and significant contributions came from engineers in both the military and aerospace sectors.

Step 2: Pioneers in UAV Development.

- The early development of UAVs can be traced back to the 1900s, with innovations from companies and military research. - In the 1990s, civilian drones emerged for various applications such as surveillance, photography, and mapping.

Step 3: Conclusion.

Drone technology is a product of multiple inventors and is still evolving with contributions from various fields, including defense, agriculture, and technology.

Final Answer:

Multiple inventors

Quick Tip

Drones are the result of contributions from multiple engineers and organizations across time.

Q3. (a) Why was OOP needed?

Solution: Step 1: Understanding OOP.

Object-Oriented Programming (OOP) was introduced to manage complexity in software development. The paradigm focuses on organizing code into objects that can hold data and methods, making the software more modular, reusable, and easier to maintain.

Step 2: Reasons for OOP.

OOP was needed for several reasons: 1. Encapsulation: Allows hiding of the internal state of an object and only exposing necessary functionality. 2. Inheritance: Allows for code reuse by creating new classes based on existing ones. 3. Polymorphism: Enables methods to have different meanings based on the object calling them. 4. Abstraction: Helps in reducing complexity by hiding implementation details and exposing only relevant functionalities.

OOP provides modularity, code reuse, and easier maintenance.

Quick Tip

OOP improves maintainability and reduces code redundancy.

Q3(b). What is core Java? Describe it in brief.

Solution: Core Java refers to the fundamental features of Java programming, which include the basic libraries and APIs that are essential for building any Java application. It consists of the core components of Java, such as: 1. Object-Oriented concepts (classes, objects, inheritance) 2. Java Standard Libraries (Java.io, Java.net, etc.) 3. Java Syntax and basic data structures Core Java is typically used to build desktop and small-scale applications.

Core Java provides the essential building blocks of Java programming.

Quick Tip

Core Java is the foundation of Java programming and essential for every Java developer.

Q3(c). What is the difference between core Java and advanced Java?

Solution: Core Java is the basic version of Java, used for simple applications like desktop applications. Advanced Java, on the other hand, extends Core Java and includes additional APIs and libraries used for developing complex applications like enterprise-level systems, web services, and databases.

Key Differences: 1. Core Java focuses on foundational concepts like classes and methods, whereas advanced Java includes frameworks like Servlets and JSP for building web applications. 2. Advanced Java includes networking and database connectivity, which are not covered in core Java.

Advanced Java builds on core Java for more complex applications.

Quick Tip

Core Java is essential for understanding advanced Java concepts.

Q3(d). Write any eight applications of robots.

Solution: 1. Industrial Automation 2. Medical Surgery 3. Space Exploration 4. Surveillance and Security 5. Agriculture (automated farming) 6. Military Defense 7. Household Robots (vacuum cleaners) 8. Service Industry (restaurants, hotels)

Robots are used in a variety of industries for automation and precision.

Quick Tip

Robots play a crucial role in automation across industries.

Q3(e). Write the introduction of drones.

Solution: Drones, also known as Unmanned Aerial Vehicles (UAVs), are aircrafts that do not require a human pilot onboard. Drones are controlled remotely or autonomously via software. They have applications in fields such as agriculture, delivery services, surveillance, and even military operations. The technology has advanced significantly in recent years, making drones more accessible for various commercial and personal uses.

Drones are revolutionizing several industries with their versatility and functionality.

Quick Tip

Drones are used in various industries for monitoring, delivery, and automation.

Q4. (a) Define classes in OOP.

Solution: In Object-Oriented Programming (OOP), a class is a blueprint or template for creating objects. It defines a type in terms of the data it holds (attributes) and the operations (methods) that can be performed on that data. Classes are fundamental for creating objects, which are instances of a class, and they encapsulate both the state (attributes) and behavior (methods) of the objects.

Step 1: Structure of a Class. A class typically includes: - Attributes (also called fields or properties) that represent the state of an object. - Methods (functions) that define the behavior of the objects. - Constructors for initializing objects of the class.

Step 2: Example of a class. Consider the following simple class in Java:

```
class Car {
    String model;
    int year;

    // Constructor
    Car(String model, int year) {
        this.model = model;
        this.year = year;
    }

    // Method
    void displayInfo() {
        System.out.println("Model: " + model + ", Year: " + year);
    }
}
```

This class defines a blueprint for creating Car objects with two attributes: model and year, and one method: displayInfo, which prints the details of the car.

Quick Tip

A class is a blueprint for creating objects, containing attributes and methods that define their behavior.

Q4. (b) Write the benefits of Java.

Solution: Java is a widely used, powerful, object-oriented programming language with several benefits, including:

Step 1: Platform Independence. Java is platform-independent due to its ”Write Once, Run Anywhere” philosophy. Java programs are compiled into bytecode, which can be executed on any platform that has a Java Virtual Machine (JVM).

Step 2: Object-Oriented. Java is an object-oriented programming language, meaning it promotes concepts like inheritance, encapsulation, and polymorphism, which help in building scalable and reusable code.

Step 3: Security. Java provides a secure environment by using bytecode verification and a security manager to restrict access to certain system resources.

Step 4: Rich API. Java provides a rich set of libraries (API), which simplifies many programming tasks such as file handling, networking, and database access.

Step 5: Multithreading Support. Java supports multithreading, allowing the development of highly responsive and efficient programs that can perform multiple tasks concurrently.

Quick Tip

Java’s ”Write Once, Run Anywhere” feature makes it platform-independent and a great choice for cross-platform applications.

Q4. (c) Write about different editions of Java.

Solution: Java has different editions, which cater to various types of applications and devices:

Step 1: Java Standard Edition (SE). Java SE is the core edition, providing the foundation for building desktop applications, utilities, and basic network applications. It includes essential libraries for development such as the collections framework, networking, and file I/O.

Step 2: Java Enterprise Edition (EE). Java EE extends the Java SE with additional features for building large-scale, distributed, and multi-tiered enterprise applications. It includes technologies like Enterprise JavaBeans (EJB), JavaServer Pages (JSP), and Java Message Service (JMS).

Step 3: Java Micro Edition (ME). Java ME is used for developing applications on embedded and mobile devices. It provides a lightweight framework for resource-constrained environments, such as cell phones, smartcards, and small sensors.

Step 4: JavaFX. JavaFX is a platform for building rich user interfaces and is designed for applications running on desktops, mobile devices, and the web.

Quick Tip

Java editions serve different purposes: SE for core development, EE for enterprise, ME for mobile, and JavaFX for rich UIs.

Q4. (d) What are the methods of robot programming?

Solution: Robot programming refers to writing instructions for robots to perform specific tasks. There are several methods of robot programming:

Step 1: Teach Pendant Programming. In this method, the operator manually moves the robot using a teach pendant and records the robot's movements. This is useful for tasks that involve simple, repetitive actions.

Step 2: Offline Programming. Offline programming uses simulation software to create and test robot programs before they are implemented on the actual robot. This method allows for optimizing the robot's task sequences without interrupting production.

Step 3: Lead Through Programming. In this method, the robot is manually guided through the task while recording its movements. It is similar to teach pendant programming but can be done by physically guiding the robot through the task instead of using a device.
Step 4: Visual Programming. In this approach, robots are programmed by arranging visual blocks or flowcharts in a graphical interface. This method is beginner-friendly and is commonly used in educational settings.

Quick Tip

Visual and offline programming are effective for simplifying robot programming, especially in industrial and educational environments.

Q4. (e) Write about the classification of drones.

Solution: Drones, also known as Unmanned Aerial Vehicles (UAVs), are classified based on various factors such as their size, range, and intended use.
Step 1: Classification by Size. - **Micro Drones:** Small drones, often used for personal or educational purposes. They are typically lightweight and easy to fly. - **Mini Drones:** Slightly larger than micro drones, often used for recreational or basic professional tasks. - **Small Drones:** These drones are used for aerial photography and light surveying. - **Large Drones:** Used in commercial applications such as heavy payload delivery, industrial inspection, and military use.
Step 2: Classification by Range. - **Short Range Drones:** These drones are designed for indoor or small-scale outdoor operations and can travel up to a few kilometers. - **Medium Range Drones:** Typically used for commercial photography and surveillance, capable of covering longer distances. - **Long Range Drones:** These drones can fly for hundreds of kilometers and are used for large-scale surveys, military missions, and long-distance transportation.
Step 3: Classification by Purpose. - **Consumer Drones:** Used for personal entertainment, photography, and recreational flying. - **Commercial Drones:** Used for professional tasks such as agriculture, surveillance, mapping, and logistics. - **Military

Drones:** Designed for surveillance, reconnaissance, and tactical operations.

Quick Tip

Drones are classified by size, range, and purpose: consumer, commercial, and military drones.

Q5. a.Write the difference between object-oriented and structured programming.

Correct Answer: (A) Object-Oriented Programming (OOP) is based on objects, which encapsulate data and behavior together. Structured Programming (SP) uses a step-by-step approach where the program's flow is linear and divided into functions.

Solution: Object-Oriented Programming (OOP) and Structured Programming (SP) are two different paradigms, each with distinct principles for organizing and designing code.
1. **Object-Oriented Programming (OOP)**: - **Concept**: OOP focuses on objects that represent real-world entities. These objects are instances of classes and encapsulate both **data (attributes)** and **methods (behaviors)**. - **Core Concepts**: - **Encapsulation**: Data and methods are grouped into a single unit (the class). - **Inheritance**: A class can inherit methods and attributes from another class. - **Polymorphism**: Objects of different classes can share the same method name but perform different actions. - **Abstraction**: Hides complex implementation details and only exposes essential features. - **Example**: Consider a ‘Car‘ class that defines attributes like ‘speed‘ and ‘color‘, and methods like ‘accelerate()‘ and ‘brake()‘. A subclass ‘ElectricCar‘ could inherit these features and add more specific methods like ‘charge()‘.
2. **Structured Programming (SP)**: - **Concept**: SP follows a top-down approach, focusing on breaking the program into smaller sub-problems or functions. The program flow is sequential and linear, emphasizing clarity and simplicity. - **Key Characteristics**: - The use of **functions** (procedures) to divide the problem. - A linear flow of control using **loops**, **conditionals**, and **function calls**. - **Example**: A program that calculates the sum of numbers would have a function ‘add()‘ that takes two numbers as input and returns their sum.

3. **Major Differences**: - OOP organizes code around objects and promotes reusability and scalability, making it suitable for large, complex systems. - SP organizes code around functions and linear control flow, making it easier to understand for small programs but less flexible for larger systems.

Quick Tip

OOP is better for handling large, complex software projects due to modularity, while SP is simpler for smaller tasks where a clear, linear execution is needed.

Q5. b.How do you use inheritance in OOP? Write with examples.

Correct Answer: (A) Inheritance allows one class to inherit properties and methods from another class, promoting code reuse and extending functionality.

Solution: **Inheritance** is a key feature of Object-Oriented Programming (OOP) that allows a class to inherit properties and methods from another class, thereby enabling code reuse and extending the functionality of existing classes.
1. **How Inheritance Works**: - The child class (subclass) inherits all public and protected members (variables and methods) from the parent class (superclass). - The subclass can then **override** the methods of the parent class to provide its own specific implementation, or **extend** the functionality by adding new methods. - Inheritance is a ”is-a” relationship. For example, a ‘Dog‘ class **is a** type of ‘Animal‘, so it can inherit properties from the ‘Animal‘ class.
2. **Example**:

```
// Parent class (Superclass)
class Animal {
    String name;

    public void eat() {
        System.out.println("This animal is eating.");
```

```
    }

    public void sleep() {
        System.out.println("This animal is sleeping.");
    }
}

// Child class (Subclass) inheriting from Animal
class Dog extends Animal {

    public void bark() {
        System.out.println("The dog is barking.");
    }

    // Overriding the eat method
    @Override
    public void eat() {
        System.out.println("The dog is eating dog food.");
    }
}

public class Test {
    public static void main(String[] args) {
        Dog dog = new Dog();
        dog.name = "Buddy";
        dog.eat();  // Output: The dog is eating dog food.
        dog.sleep(); // Output: This animal is sleeping.
        dog.bark();  // Output: The dog is barking.
    }
}
```

- In this example: - The ‘Dog‘ class inherits the ‘eat()‘ and ‘sleep()‘ methods from the ‘Animal‘ class. - The ‘Dog‘ class also has a unique method ‘bark()‘. - The ‘Dog‘ class **overrides** the ‘eat()‘ method to provide specific behavior.
3. **Benefits of Inheritance**: - **Code Reusability**: By inheriting from a parent class, the child class does not need to write the same code again. - **Modularity**: Inheritance allows for better organization of code by creating hierarchies, making it easier to maintain and extend the codebase.

Quick Tip

Inheritance promotes reusable and maintainable code by establishing relationships between objects.

Q5. c.Write about key core Java concepts.

Correct Answer: (A) Core Java concepts include classes, objects, inheritance, polymorphism, abstraction, and encapsulation.

Solution: Core Java concepts are fundamental principles that form the basis of Java programming. These concepts enable developers to build structured, modular, and maintainable Java applications.
1. **Classes and Objects**: - **Class**: A class is a blueprint for creating objects. It defines the properties (attributes) and methods (behaviors) of objects. - **Object**: An object is an instance of a class, representing real-world entities with specific data.
2. **Inheritance**: - Inheritance allows one class (the child class) to inherit the properties and behaviors of another class (the parent class). It facilitates code reuse and makes the program more modular.
3. **Polymorphism**: - Polymorphism enables one interface (method) to be used for different data types. It can be achieved through **method overriding** (runtime polymorphism) and **method overloading** (compile-time polymorphism).
4. **Abstraction**: - Abstraction is the process of hiding the implementation details from the user and exposing only essential features. It can be implemented using **abstract

classes** and **interfaces**.

5. **Encapsulation**: - Encapsulation is the bundling of data and methods that operate on that data within a single unit (class). It restricts direct access to some of the object's components and ensures data integrity.

Quick Tip

Mastering these core Java concepts is essential for building robust and scalable Java applications.

Q5. d.Write about literals in Java.

Correct Answer: (A) Java literals are fixed values used directly in the code. They include integer, floating-point, character, and boolean literals.

Solution: In Java, literals represent constant values directly written into the code. They are of several types:

1. **Integer Literals**: Represent integer values. - Example: ‘int x = 100;‘

2. **Floating-Point Literals**: Represent decimal values. - Example: ‘double pi = 3.14159;‘

3. **Character Literals**: Represent a single character enclosed in single quotes. - Example: ‘char grade = ’A’;‘

4. **Boolean Literals**: Represent boolean values, either ‘true‘ or ‘false‘. - Example: ‘boolean isActive = true;‘

Literals can also be used to represent other data types, such as string literals, which are enclosed in double quotes.

Quick Tip

Literals provide a way to assign constant values directly to variables without the need for complex calculations or transformations.

(Q5.e) Describe AWT.

Solution:

AWT (Abstract Window Toolkit) is a set of application programming interfaces (APIs) provided by Java for building Graphical User Interfaces (GUIs). It allows developers to create windows, buttons, text fields, and other common graphical elements for user interaction. Here's how AWT works step-by-step:

Step 1: Understanding the role of AWT in Java applications:

AWT is used in Java to create and manage graphical user interfaces. It provides a framework that allows users to interact with Java applications through visual elements such as buttons, labels, and text fields.

Step 2: Components of AWT:

AWT provides a variety of components (known as controls or widgets) to design the GUI. Some common components are: - **Button**: A clickable button used to trigger actions. - **Label**: A non-editable text element that displays information. - **TextField**: A single-line input field where the user can type text. - **TextArea**: A multi-line text input field. - **Panel**: A container to group other components together.

Step 3: AWT Architecture:

AWT is platform-dependent because it relies on the native operating system's windowing system for rendering graphical components. AWT generates GUI components using the native code of the operating system. However, with the introduction of Swing (a more flexible framework), the limitations of AWT became evident. Swing provides more customizable components and is lightweight, unlike AWT which is dependent on native OS components.

Step 4: Example of AWT Code:

A simple example of using AWT to create a button:

```
import java.awt.*;
import java.awt.event.*;

public class AWTExample {
    public static void main(String[] args) {
```

```
        Frame frame = new Frame("AWT Example");
        Button button = new Button("Click Me");

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                System.out.println("Button clicked!");
            }
        });

        frame.add(button);
        frame.setSize(300, 300);
        frame.setVisible(true);
    }
}
```

Quick Tip

AWT is considered an older GUI framework in Java, and while still useful, Swing or JavaFX are generally preferred for more advanced applications due to their richer and more flexible set of components.

(Q5.f) Write the names of any five types of robots. Describe any one out of them.

Solution:

There are various types of robots used across industries for different purposes. Here are five examples:

1. **Industrial Robots** 2. **Service Robots** 3. **Autonomous Mobile Robots (AMRs)** 4. **Humanoid Robots** 5. **Medical Robots**

Step 1: Understanding Industrial Robots:

Industrial robots are machines designed to automate repetitive tasks in manufacturing processes. These robots are typically stationary and are used in controlled environments like

factories. They can carry out tasks such as welding, assembly, and packaging.

Step 2: Types of Industrial Robots:

- **Articulated Robots**: These have joints similar to a human arm, with a rotating base, multiple arms, and often several rotational axes. - **SCARA Robots**: SCARA stands for Selective Compliance Assembly Robot Arm, designed for assembly operations in confined spaces. - **Delta Robots**: Used for high-speed picking and packing tasks.

Step 3: Key Features of Industrial Robots:

- **Precision**: They provide high accuracy in repetitive tasks. - **Speed**: Industrial robots can operate faster than human workers, which increases production efficiency. - **Safety**: They can operate in dangerous environments, reducing the risk to human workers.

Step 4: Example of Industrial Robot Usage:

An example of an industrial robot is one used in an automobile assembly line, performing tasks like welding, assembly of parts, and painting. These robots are highly efficient and ensure consistent quality throughout the manufacturing process.

Quick Tip

Industrial robots are essential for mass production industries, as they help improve precision, reduce human error, and increase production speed.

(Q5.g) Describe the control system of a robot.

Solution:

The control system of a robot determines how the robot interacts with its environment and executes tasks. It involves several components working together to ensure proper operation.

Step 1: Basic Components of a Robot Control System:

1. **Sensors**: These are devices that gather data from the robot's surroundings or its internal states. Common sensors include cameras, ultrasonic sensors, and accelerometers. 2. **Actuators**: These are mechanical components that move or control parts of the robot, such as motors or hydraulic systems. 3. **Controller**: The brain of the robot, which

interprets inputs from the sensors and sends commands to the actuators. It processes information and makes decisions. 4. **Software/Algorithms**: The software running on the controller uses various algorithms (e.g., path planning, obstacle avoidance) to control the robot's actions.

Step 2: Types of Control Systems:

There are two main types of control systems used in robotics: - **Open-loop control system**: In this system, the controller sends commands to the robot's actuators without receiving feedback. It's typically used when feedback is unnecessary or difficult to obtain. - **Closed-loop control system**: This system constantly receives feedback from sensors and adjusts the actuators' behavior in real-time to ensure accuracy and correct performance. It's more advanced and used in robots requiring high precision.

Step 3: Feedback and Sensor Integration:

Sensors play a critical role in closed-loop control systems. For example, a robot might use a camera to track its position relative to an object. The feedback from the sensor helps the controller adjust the robot's movements.

Step 4: Example of Robot Control System:

Consider a robot vacuum cleaner, which uses a closed-loop system. It has sensors to detect obstacles and the dirt level on the floor. Based on this feedback, the controller adjusts the speed and direction of the robot to efficiently clean the room.

Quick Tip

The quality of a robot's control system directly impacts its ability to perform tasks accurately and reliably. Closed-loop systems are more complex but offer better performance in dynamic environments.

(Q5.h) Write a note on the application of drone.

Solution:

Drones, also known as Unmanned Aerial Vehicles (UAVs), have a wide range of applications across various industries. These applications capitalize on their ability to fly, access

hard-to-reach areas, and collect high-quality data from above.

Step 1: Key Applications of Drones:

1. **Military and Defense**: Drones are widely used for surveillance, reconnaissance, and targeted operations. They provide real-time information from the air and can be deployed for strategic advantage without risking human lives. 2. **Agriculture**: Drones help farmers monitor crop health, assess irrigation needs, and even spray fertilizers or pesticides. They are equipped with sensors that gather data for precision farming. 3. **Search and Rescue**: Drones are used in disaster areas to assess damage and locate survivors, especially in environments where it's unsafe for humans to go. 4. **Logistics and Delivery**: Companies like Amazon are experimenting with drones for delivering packages to customers. Drones are capable of delivering items faster, especially to remote locations. 5. **Entertainment**: Drones are used in filmmaking to capture aerial shots, providing unique perspectives that were previously difficult or expensive to achieve.

Step 2: Benefits of Using Drones:

- **Cost-effective**: Drones are cheaper to operate compared to manned aircraft or other traditional methods. - **Efficient**: Drones can cover large areas in a short amount of time, collecting data or delivering goods efficiently. - **Safety**: Drones can operate in dangerous environments (e.g., disaster zones) where it may be unsafe for humans.

Step 3: Future Potential:

Drones are expected to continue evolving with improvements in battery life, AI, and autonomous navigation. Their application in industries like infrastructure inspection, environmental monitoring, and urban planning is expected to grow significantly.

Quick Tip

While drones offer many benefits, their use also raises concerns about privacy and airspace regulations. It's important to understand local laws before deploying drones.
