

01. Arduino IDE और Sketch Workflow

Arduino IDE एक software environment है जिसमें Arduino board के लिए program लिखा, compile, debug और upload किया जाता है। Arduino program को sketch कहा जाता है और इसकी file का extension सामान्यतः .ino होता है।

Sketch के मुख्य भाग

```
void setup() {  
  // यह भाग केवल एक बार चलता है  
}  
  
void loop() {  
  // यह भाग बार-बार चलता है  
}
```

Write Sketch



Verify / Compile



Debug



Upload



Board Run

Arduino IDE में board और COM port चुनकर compiled program को board की memory में भेजा जाता है।

Compile, Debug और Upload

कार्य	अर्थ
Compile / Verify	Code को machine instructions में बदलना और syntax errors खोजना।
Debug	Error message, line number और logic को देखकर समस्या ठीक करना।
Upload	Compiled sketch को USB के द्वारा Arduino board पर transfer करना।
Serial Monitor	Board से आने वाला text/data देखना और serial input भेजना।

सामान्य workflow: Board connect करें → Tools से Board और Port चुनें → Verify करें → Error हो तो सुधारें → Upload करें → Serial Monitor से output देखें।

02. Embedded C Basics: Variables, Types और Operators

Arduino programming Embedded C पर आधारित है। Variable एक नामित memory location है जिसमें value store होती है। Identifier variable, function या object का valid नाम होता है।

Variables और Identifiers

```
int temperature = 25;
float voltage = 3.30;
char grade = 'A';
bool ledState = false;
```

Data Type	उपयोग
int	पूर्णांक values, जैसे 0, 25, -4
float	decimal values, जैसे 3.14
char	एक character, जैसे A
bool	true या false
byte	0 से 255 तक छोटी unsigned value

Arithmetic Operators और Assignment

Operator	कार्य
+ - * /	जोड़, घटाव, गुणा, भाग
%	remainder / शेषफल
=	right side की value left variable में रखना
+= -= *= /=	short assignment जैसे x += 2
++ --	value को 1 बढ़ाना या घटाना

```
int a = 10;
int b = 3;
int sum = a + b;
int remainder = a % b;
a += 5;
```

Identifier में space नहीं होता। वह letter या underscore से शुरू होना चाहिए और keywords जैसे int, if, for को identifier की तरह उपयोग नहीं करना चाहिए।

03. Constants, Literals और Conditional Statements

Constant वह value है जिसे program के दौरान बदलना नहीं चाहिए। Literal code में लिखी गई fixed value होती है, जैसे 10, 3.14, 'A' या true.

```
const int LED_PIN = 13;
const float LIMIT = 30.0;

int value = 25;
if (value > LIMIT) {
    digitalWrite(LED_PIN, HIGH);
}
```

Relational और Logical Operators

Operator	अर्थ
==	बराबर है
!=	बराबर नहीं है
> <	बड़ा / छोटा
>= <=	बड़ा या बराबर / छोटा या बराबर
&&	दोनों conditions true हों
	कम से कम एक condition true हो
!	condition का उल्टा

if-else Decision

```
if (temperature >= 30) {
    Serial.println("Fan ON");
} else {
    Serial.println("Fan OFF");
}
```

if statement condition true होने पर statements चलाता है। else condition false होने पर alternative statements चलाता है। कई choices के लिए else if chain उपयोग की जाती है।

ध्यान दें: comparison के लिए == और assignment के लिए = उपयोग होता है। इन दोनों को मिलाने से logic error हो सकता है।

04. Loops: while, do-while, for और switch

Loop किसी statement या block को बार-बार चलाने के लिए उपयोग होता है। Loop में initialization, condition और update को सही रखना जरूरी है।

while और do-while

```
int count = 0;
while (count < 5) {
    Serial.println(count);
    count++;
}

int x = 0;
do {
    Serial.println(x);
    x++;
} while (x < 3);
```

while loop में condition पहले check होती है, इसलिए loop zero times भी चल सकता है। do-while में block पहले एक बार चलता है और फिर condition check होती है।

for Loop और Nested Loop

```
for (int i = 0; i < 10; i++) {
    Serial.println(i);
}

for (int row = 1; row <= 3; row++) {
    for (int col = 1; col <= 3; col++) {
        Serial.print("*");
    }
}
```

switch Statement

```
switch (mode) {
    case 1: Serial.println("AUTO"); break;
    case 2: Serial.println("MANUAL"); break;
    default: Serial.println("INVALID");
}
```

Infinite loop तब बनता है जब condition कभी false नहीं होती, जैसे while(true) या ऐसा counter जिसे update न किया जाए। Arduino में loop() स्वयं लगातार चलता है, इसलिए अंदर के loops को सावधानी से लिखें।

05. Arrays: Single Dimension Array

Array समान data type की कई values को एक ही नाम के अंतर्गत store करता है। Single dimension array में elements एक line या list की तरह होते हैं। Index हमेशा 0 से शुरू होता है।

```
int readings[5] = {21, 23, 25, 24, 22};

Serial.println(readings[0]); // 21
readings[2] = 26;           // value बदलना
int last = readings[4];
```

Array declare और manipulate करना

कार्य	उदाहरण
Declaration	int data[5];
Initialization	int data[3] = {10, 20, 30};
Read	value = data[i];
Update	data[i] = newValue;
Length	int n = sizeof(data) / sizeof(data[0]);

```
int sensor[4] = {10, 20, 30, 40};
long total = 0;
for (int i = 0; i < 4; i++) {
    total = total + sensor[i];
}
float average = total / 4.0;
```



Index range से बाहर access करने पर गलत data या unpredictable behavior हो सकता है।

Array की सीमा का ध्यान रखें: यदि array में 5 elements हैं, तो valid indexes 0, 1, 2, 3 और 4 होंगे।

06. Functions: Prototype, Parameters और Return Type

Function reusable code का एक named block है। इससे program modular, readable और maintainable बनता है। Arduino में setup() और loop() भी functions हैं।

Function का structure

```
// Prototype
int add(int a, int b);

// Definition
int add(int a, int b) {
    return a + b;
}

// Function call
int result = add(4, 6);
```

भाग	अर्थ
Prototype	Compiler को function का नाम, parameters और return type पहले बताता है।
Formal parameter list	Function definition में आने वाले parameter names और types।
Return Type	Function किस type की value लौटाएगा; value न हो तो void।
Function Call	Function को चलाने का statement; arguments pass किए जा सकते हैं।

```
void blinkLED(int pin, int times) {
    for (int i = 0; i < times; i++) {
        digitalWrite(pin, HIGH);
        delay(200);
        digitalWrite(pin, LOW);
        delay(200);
    }
}
```

Function को छोटा और single-purpose रखें। इससे sensor reading, display update और actuator control को अलग-अलग test करना आसान होता है।



07. Digital और Analog Pins: LED तथा Button

Arduino pins devices से input लेने या output देने के लिए उपयोग होते हैं। Digital pin सामान्यतः HIGH या LOW दो states रखता है। Analog pin voltage level को ADC के द्वारा numeric value में बदलता है।

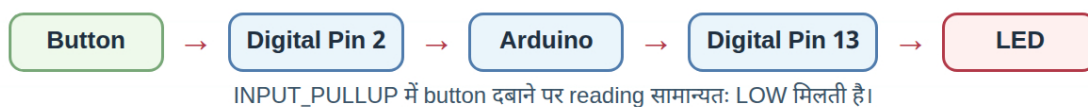
Pin type	कार्य
Digital input	Button या switch से HIGH / LOW पढ़ना
Digital output	LED, buzzer या relay को HIGH / LOW करना
Analog input	LDR या sensor का बदलता voltage पढ़ना, जैसे analogRead()
PWM output	कुछ digital pins पर analogWrite() से brightness या speed control

LED और Button Example

```
const int LED = 13;
const int BUTTON = 2;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(BUTTON, INPUT_PULLUP);
}

void loop() {
  int pressed = digitalRead(BUTTON);
  digitalWrite(LED, pressed == LOW ? HIGH : LOW);
}
```



LED के साथ current-limiting resistor लगाएँ। Button circuit में floating input से बचने के लिए pull-up या pull-down व्यवस्था रखें।

08. Sensors: DHT, LDR, MQ135 और IR

Sensor physical quantity को electrical signal में बदलता है। Arduino उस signal को read करके display, decision या actuator control के लिए उपयोग कर सकता है।

Sensor	क्या मापता है और pin method
DHT	Temperature और humidity; digital communication, DHT library
LDR	Light intensity; resistance बदलती है, analogRead() से value
MQ135	Air quality / gases का approximate analog signal
IR Sensor	Infrared light से obstacle या object detection; digital/analog output

DHT Sensor Example

```
#include <DHT.h>
#define DHTPIN 7
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

void setup() { Serial.begin(9600); dht.begin(); }
void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  Serial.println(t);
  delay(2000);
}
```

Analog sensor reading

```
int light = analogRead(A0);
int air = analogRead(A1);
int obstacle = digitalRead(4);
```

Sensor reading को raw value से meaningful unit में बदलने के लिए calibration और datasheet की जानकारी आवश्यक हो सकती है। MQ135 का output exact laboratory measurement नहीं माना जाना चाहिए।



09. LCD Display और Keypad Interfacing

LCD sensor data को user के सामने text के रूप में दिखाने के लिए उपयोग होता है। Keypad user से number, mode या command input लेने का तरीका है।

16×2 LCD Example

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  lcd.print("IoT Monitor");
}

void loop() {
  lcd.setCursor(0, 1);
  lcd.print("Temp: 25 C  ");
  delay(1000);
}
```

Sensors



Arduino



LCD Display

lcd.setCursor(column, row) से cursor की position बदलते हैं; row 0 या 1 होता है।

Keypad का विचार

Matrix keypad में rows और columns होते हैं। Arduino एक-एक row और column scan करके pressed key पहचानता है। Keypad library scanning को सरल बनाती है।

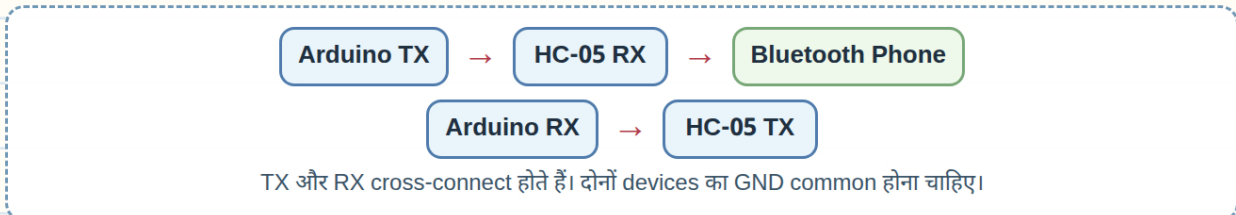
```
#include <Keypad.h>
// rows और columns के pins define करें
// keypad.getKey() से pressed key पढ़ें
char key = keypad.getKey();
if (key) { lcd.print(key); }
```

LCD output और keypad input मिलकर menu-based IoT device बना सकते हैं, जैसे password lock, temperature setting या manual control panel।

10. Serial Communication और HC-05 Bluetooth

Serial communication में data bits की stream के रूप में एक device से दूसरे device तक भेजा जाता है। Arduino में Serial Monitor और बाहरी modules के लिए UART serial communication उपयोगी है।

HC-05 Connection



Pin	कार्य
VCC	Module को supply; अपने module का voltage rating देखें
GND	Common ground
TXD	HC-05 से Arduino की ओर data
RXD	Arduino से HC-05 की ओर data; level compatibility देखें

SoftwareSerial Example

```
#include <SoftwareSerial.h>
SoftwareSerial bt(10, 11); // RX, TX

void setup() {
  Serial.begin(9600);
  bt.begin(9600);
}

void loop() {
  if (bt.available()) {
    char c = bt.read();
    Serial.write(c);
  }
}
```

HC-05 के RX pin की voltage requirement जाँचें। जरूरत होने पर voltage divider लगाएँ। गलत wiring या गलत baud rate से unreadable characters मिल सकते हैं।



11. Relay Module से 220V AC Control

Relay electrically controlled switch है। Arduino का low-voltage signal relay coil या module को control करता है और relay contacts अलग load को ON/OFF करते हैं। इससे microcontroller और load के बीच isolation मिल सकती है।

Arduino Pin



Relay Module



COM / NO Contacts



AC Load

Arduino signal केवल relay module को control करता है; load की power relay contacts से गुजरती है।

Basic Arduino control

```
const int RELAY = 7;
void setup() {
  pinMode(RELAY, OUTPUT);
  digitalWrite(RELAY, LOW);
}
void loop() {
  digitalWrite(RELAY, HIGH);
  delay(5000);
  digitalWrite(RELAY, LOW);
  delay(5000);
}
```

Contact	अर्थ
COM	Common terminal
NO	Normally Open; activation पर connect
NC	Normally Closed; activation पर disconnect

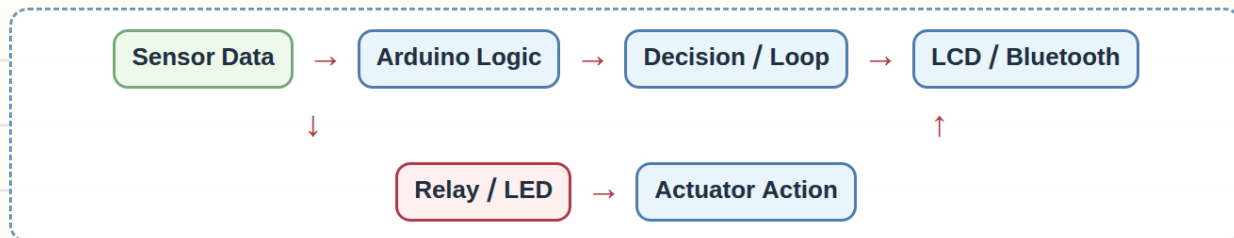
220V AC जानलेवा हो सकता है। Mains wiring केवल qualified person से कराएँ; power disconnect रखें, insulated enclosure, fuse और proper earthing का उपयोग करें। Breadboard पर exposed 220V wires न लगाएँ।

Exam concept: Arduino low-voltage control signal → relay coil/module → isolated switching contacts → AC appliance. Relay की voltage और current rating load से अधिक होनी चाहिए।



12. Building IoT Application: Complete Flow और Revision

एक complete application का flow



Stage	उदाहरण
Sense	DHT से temperature और LDR से light पढ़ना
Process	if-else, loops, arrays और functions से data संभालना
Display / Communicate	LCD पर दिखाना और HC-05 से phone को भेजना
Act	Relay, LED या buzzer को control करना
Test	Serial Monitor, compile messages और measured output से verify करना

Revision Checklist

Arduino IDE में Board और Port चुनना; sketch में setup() और loop(); variables और data types; arithmetic, relational और logical operators; if-else, switch और loops; single dimension arrays; function prototype, parameter और return type; digital तथा analog pins; DHT, LDR, MQ135 और IR; LCD और keypad; HC-05 serial communication; relay द्वारा AC load control.

Exam Tip: किसी interfacing answer में हमेशा pin connection, library / function, छोटा code flow और safety point लिखें।