Task 1

Deep Das

Question : Types of drones and their uses.

Answer :

1. Multirotor Drones:

Uses: Aerial photography, surveillance, inspection, search and rescue.

Example: DJI Phantom.

2. Fixed-Wing Drones:

Uses: Long-range surveillance, agricultural monitoring, mapping, and surveying.

Example: SenseFly eBee.

3. Single Rotor Drones:

Uses: Heavy payload delivery, longer flight times, aerial LIDAR scanning.

Example: Yamaha RMAX.

4. Hybrid VTOL (Vertical Take-Off and Landing) Drones:

Uses: Combines the advantages of multirotor and fixed-wing drones; useful for long-distance travel and precise take-offs and landings.

Example: Quantum Systems Tron.

Question : What is VTOL.

Answer:

VTOL (Vertical Take-Off and Landing): VTOL refers to aircraft that can take off, hover, and land vertically. This capability is particularly useful in areas with limited space for traditional runway operations. VTOL aircraft combine the benefits of helicopters (vertical takeoff and landing) with the range and speed of fixed-wing planes. Examples include the Harrier Jump Jet and the V-22 Osprey, as well as modern drones designed for similar functionality.

Question : Which is fast and better? c/c++.

Answer : When comparing C and C++, it is often noted that C is generally considered better for certain types of applications due to its simplicity and efficiency. C is known for its faster performance because it has less overhead and a simpler compilation process, making it ideal for low-level programming tasks such as embedded systems, system programming, and real-time applications. The minimalistic runtime and straightforward syntax of C allow developers to have fine-grained control over memory management and hardware, which can lead to more optimized and efficient code. Additionally, C's procedural programming paradigm makes it easier to understand and maintain for smaller, less complex projects. While C++ offers powerful features such as object-oriented programming, a more extensive standard library, and better code organization and reusability, these advantages often come with a trade-off in terms of performance and complexity. Therefore, for projects where performance is critical and the application requires direct hardware manipulation, C is often the preferred choice.

Question : Basic multifunction calculator.

Answer :

```c
#include <stdio.h>

int main() {
    char operator;
    double first, second;
    printf("Enter an operator (+, -, *, /): ");
    scanf("%c", &operator);
    printf("Enter two operands: ");
    scanf("%lf %lf", &first, &second);

    switch (operator) {
    case '+':
        printf("%.2lf + %.2lf = %.2lf", first, second, first + second);
        break;
    case '-':
        printf("%.2lf - %.2lf = %.2lf", first, second, first - second);
        break;
    case '*':
        printf("%.2lf * %.2lf = %.2lf", first, second, first * second);
        break;
    case '/':
        if (second != 0.0)
            printf("%.2lf / %.2lf = %.2lf", first, second, first / second);
        else
            printf("Error! Division by zero.");
        break;
    default:
        printf("Error! Operator is not correct");
    }

    return 0;
}
```

Question : Differentiate between single line and multi line comments with examples.

Answer :

Single-line comments start with `//` and extend to the end of the line. They are useful for brief annotations. For example, a single-line comment might be used to indicate that a variable is being initialized, or to describe a single statement in a loop.

Example : //hello deep

Multi-line comments start with `/*` and end with `*/`. They can span multiple lines, making them ideal for longer explanations or temporarily commenting out blocks of code. For example, a multi-line comment might be used to explain the purpose of a function, describe complex logic, or disable several lines of code during debugging.

Example:/* some big lines of code with mansion gina cinf s ciajd w icon woconad onc adoqndona cancans*/

In summary, single-line comments are best for brief, inline notes, such as marking where a variable is set or explaining a single line of code. Multi-line comments are suitable for detailed explanations, such as describing the functionality of a large section of code, outlining a function's purpose, or temporarily disabling code segments.