📝 Task

Generate runnable programming examples in C, Python, Java, Flask, and Django covering core concepts for each, and then review and present all generated examples.

Table of Contents

1. C - Core Concepts: Basic I/O and Functions

This example demonstrates the structure of a basic C program, including the `#include` directive, the `main` **function**, standard output (`printf`), and a simple custom function.

```
#include <stdio.h> // Include standard input/output library

// Function declaration/prototype
int calculate_sum(int a, int b);

int main() {
    // Variable declaration and initialization
    int num1 = 10;
    int num2 = 25;
    int result;

    printf("--- C Programming Example ---\n");
    printf("Hello, C World!\n");

    // Function call
    result = calculate_sum(num1, num2);

    // Print the result
    printf("The sum of %d and %d is: %d\n", num1, num2, result);
    return 0; // Indicate successful execution
}

// Function definition
int calculate_sum(int a, int b) {
    return a + b;
}
```

2. Python - Core Concepts: Data Structures (List, Dictionary) and Conditional Logic

This example showcases basic Python features: variable assignment, a **list**, a **dictionary**, a **loop** (`for`), and **conditional logic** (`if/else`).

```python
# --- Python Programming Example ---

# 1. Data Structures
fruits = ["apple", "banana", "cherry", "date"]
inventory = {
    "apple": 50,
    "banana": 100,
    "cherry": 75
}

print("Inventory Check:")

# 2. Iteration (Loop)
for item in fruits:
    # 3. Conditional Logic
    if item in inventory:
        count = inventory[item]
        print(f"We have {count} of {item}.")
    else:
        print(f"'{item}' is out of stock or not listed.")

# Simple function to demonstrate a lambda (anonymous function)
multiply = lambda x, y: x * y
print(f"\nResult of 5 * 8 using lambda: {multiply(5, 8)}")
```

3. Java - Core Concepts: Classes, Objects, and Methods

This example demonstrates the core principles of **Object-Oriented Programming (OOP)** in Java: defining a **class**, creating an **object**, and calling methods.

```
// --- Java Programming Example ---

// The Class Definition
class Dog {
    // 1. Instance Variables (Attributes)
    String breed;
    String name;

    // 2. Constructor
    public Dog(String name, String breed) {
        this.name = name;
        this.breed = breed;
    }

    // 3. Method (Behavior)
    public void bark() {
        System.out.println(name + " says Woof! I am a " + breed + ".
    }

    // Method to run main execution
    public static void main(String[] args) {
        System.out.println("--- Java Programming Example ---");

        // 4. Object Instantiation
        Dog myDog = new Dog("Buddy", "Golden Retriever");

        // 5. Method Call
        myDog.bark();
    }
}
```

4. Flask - Core Concepts: Routing and Web Server

This example demonstrates the **micro-framework** nature of **Flask**, showing how to set up a minimal web application, define a **route** (`/`), and return an **HTTP response**.

Note: To run this in a standard Colab environment, you may need to install the library (`!pip install Flask`) and use a tool like `ngrok` for external access.

```
# Install Flask if you haven't already: !pip install Flask
from flask import Flask

# 1. Create a Flask application instance
```

```
app = Flask(__name__)

# 2. Define a Route: The default URL (/)
@app.route('/')
def hello_world():
    # 3. Return an HTTP response
    return '<h1>Hello from Flask!</h1><p>This is the default route.<

# Define another route
@app.route('/info')
def show_info():
    return '<h2>Info Page</h2><p>Flask is a lightweight web framewor

# 4. Run the development server
if __name__ == '__main__':
    print("--- Flask Programming Example ---")
    print("Code defines the app structure, usually accessed via brow
    # app.run(host='0.0.0.0', port=5000) # Commented out for smooth
```

5. Django - Core Concepts: Project Structure and Views

This is an illustrative conceptual example for **Django**, focusing on the core **View** and **URL Routing** concepts.

Runnable `myapp/views.py` Example (Focus on the View):

```
# Note: This file represents the contents of 'myapp/views.py'
from django.http import HttpResponse

def simple_view(request):
    """
    A core Django View function that accepts an HttpRequest and retu
    """
    print("--- Django Programming Example (Conceptual View) ---")

    # 1. Business Logic
    context = {
        'framework_name': 'Django',
        'message': 'This data was generated by the view function.'
    }

    # 2. Return an HTTP Response (simple text/HTML)
    return HttpResponse(f"<h1>Hello from {context['framework_name']}
                        f"<p>{context['message']}</p>")
```

Conceptual `myproject/urls.py` Example (Focus on Routing):

```
# Note: This file represents the contents of 'myproject/urls.py'
from django.contrib import admin
from django.urls import path
# from myapp.views import simple_view # Assuming import

urlpatterns = [
    # 1. Routing Definition
    path('admin/', admin.site.urls),
    # 2. Map the root URL '' to the 'simple_view' function
    # path('', simple_view, name='home'),
]

print("--- Django Routing Example ---")
print("The path function maps a URL pattern (e.g., '/') to a view fu
```

6. Review and Presentation

Summary of Core Concepts Covered

Technology	Core Concepts Demonstrated	Paradigm
C	Functions, Basic I/O (`printf`), Variable Scope	Procedural
Python	Lists, Dictionaries, `for` Loops, `if/else` , Lambda	Multi-paradigm (Scripting)
Java	Classes, Objects, Methods, Constructors	Object-Oriented (OOP)
Flask	Routing (`@app.route`), Minimal App Setup, HTTP Response	Micro Web Framework
Django	Views (Business Logic), URL Routing	Full-Stack Web Framework