🔹 LangChain Runnables

1. What are Runnables?

- **Runnables** are the **basic building blocks** in LangChain.
- They represent **any unit of computation** (e.g., LLM call, prompt template, function, or chain).
- They can be composed into workflows where input flows through different steps.

---

2. Task-Specific Runnables vs Runnable Primitives

✅ Task-Specific Runnables

- Predefined for **specific use cases**.
- Examples:
  - PromptTemplate (formatting input for LLMs)
  - LLM (large language model call)
  - Retriever (fetching documents)
  - OutputParser (structuring model responses)

👉 These are like **ready-made tools for common tasks**.

✅ Runnable Primitives

- **Generic, low-level building blocks** to design custom workflows.
- They are flexible and can be combined into **structured pipelines**.
- Useful when task-specific runnables aren't enough.

---

3. Types of Runnable Primitives

🔹 (a) RunnableSequence

- **Definition:** Executes runnables **step by step**, passing the output of one as the input to the next.
- **Use Case:** When tasks must be done in a **pipeline**.

- **Non-code Example:**
  - **R1:** Write a blog outline →
  - **R2:** Expand outline into a full blog post.

🔹 (b) RunnableParallel

- **Definition:** Executes runnables **in parallel**, each receiving the same input.
- **Use Case:** When you want **multiple outputs from the same input**.
- **Non-code Example:**
  - Input topic = "AI" →
    - **LLM1:** Generate a Tweet
    - **LLM2:** Generate a LinkedIn post

🔹 (c) RunnablePassthrough

- **Definition:** Returns the input **as-is**, without modification.
- **Use Case:**
  - Useful for debugging or when you need to **forward data unchanged**.
- **Non-code Example:**
  - Input = {“topic”: “AI”} → Output = {“topic”: “AI”}

🔹 (d) RunnableLambda

- **Definition:** Lets you apply a **custom Python function** in the workflow.
- **Acts as middleware** for preprocessing, filtering, API calls, etc.
- **Use Case:**
  - When you need **custom logic** in between AI steps.
- **Non-code Example:**

  - Before sending text to an LLM, run a Lambda that converts all text to lowercase.

🔹 (e) RunnableBranch

- **Definition:** A **control flow component** – conditionally routes input to different runnables (like if/else).
- **Use Case:**
  - When you need **conditional workflows**.
- **Non-code Example:**
  - If query = "math problem" → send to **Calculator tool**
  - Else if query = "chat" → send to **LLM**

---

4. LangChain Expression Language (LCEL)

- A **shorthand syntax** to compose runnables.
- Makes workflows concise and readable.
- Example: Instead of writing RunnableSequence(r1, r2, r3), you can use:
- r1 | r2 | r3
- **Non-code Analogy:** Like a factory assembly line where each station (runnable) processes the item in sequence.

✅ In summary:

- **Runnables** = building blocks.
- **Task-specific runnables** = ready-made tools.
- **Runnable primitives** = flexible workflow operators.
- Types (Sequence, Parallel, Passthrough, Lambda, Branch) let you design **custom, structured AI pipelines**.