A. Basics of Data Ingestion (Conceptual)

Q1. In a RAG pipeline, what is the primary purpose of data ingestion?
a) To generate embeddings
b) To feed structured/unstructured data into retrievers
c) To train the language model
d) To visualize documents
Answer: b

Q2. Which of the following is NOT a common data source for RAG ingestion?
a) PDFs
b) APIs
c) Relational Databases
d) Compilers
Answer: d

Q3. Why is document chunking important during ingestion?
a) Reduces training time
b) Improves retrieval accuracy
c) Converts images into embeddings
d) Enhances LLM fine-tuning
Answer: b

Q4. Which strategy is most commonly used to split large documents in RAG pipelines?
a) Recursive character splitting
b) Token-based splitting
c) Semantic paragraph splitting
d) All of the above
Answer: d

Q5. Ingestion pipelines often involve metadata extraction. What is the key benefit?
a) Reduces vector dimensions
b) Enables filtered retrieval based on context
c) Prevents hallucination
d) Compresses embeddings
Answer: b

B. LangChain – Data Loaders

Q6. LangChain provides pre-built **document loaders**. Which loader would you use to ingest PDFs?

a) `PyPDFLoader`
b) `CSVLoader`
c) `APILoader`
d) `DocxReader`

Answer: a

Q7. Which LangChain class helps manage multiple loaders together?

a) `RecursiveCharacterSplitter`
b) `DocumentLoaderManager`
c) `DirectoryLoader`
d) `LoaderPipeline`

Answer: c

Q8. When ingesting documents from a REST API in LangChain, which loader is best suited?

a) `UnstructuredHTMLLoader`
b) `JSONLoader`
c) `APILoader`
d) `URLLoader`

Answer: d

Q9. In LangChain, **Text Splitters** are used to:

a) Create embeddings
b) Break text into smaller, retrievable chunks
c) Reduce hallucinations
d) Tokenize embeddings

Answer: b

Q10. If you want to ingest both structured and unstructured data from multiple formats, which LangChain loader can handle it automatically?

a) `UnstructuredFileLoader`
b) `DirectoryLoader`
c) `RecursiveLoader`

d) `MultiDocLoader`
Answer: a

C. LlamaIndex – Index Building & Ingestion

Q11. LlamaIndex was previously known as:
a) PineconeIndex
b) GPT-Index
c) VectorHub
d) FAISSIndex
Answer: b

Q12. Which LlamaIndex class handles PDF ingestion?
a) `SimpleDirectoryReader`
b) `GPTPDFReader`
c) `UnstructuredPDFLoader`
d) `PDFIndexReader`
Answer: a

Q13. While building indexes in LlamaIndex, chunk size affects:
a) Embedding dimensionality
b) Retrieval precision and recall
c) Model fine-tuning speed
d) LLM hallucination rate
Answer: b

Q14. Which type of index in LlamaIndex is best for large-scale vector search?
a) `TreeIndex`
b) `ListIndex`
c) `VectorStoreIndex`
d) `SummaryIndex`
Answer: c

Q15. LlamaIndex integrates seamlessly with which vector database for ingestion?
a) FAISS
b) Pinecone
c) Weaviate

d) All of the above
Answer: d

D. Unstructured.io – Intelligent Parsing

Q16. What is the core functionality of **Unstructured.io**?
a) Embedding generation
b) Extracting structured text from unstructured documents
c) Model fine-tuning
d) Vector indexing
Answer: b

Q17. Which file types can Unstructured.io handle?
a) PDF
b) HTML
c) Emails
d) All of the above
Answer: d

Q18. Unstructured.io is particularly helpful when dealing with:
a) Poorly formatted PDFs
b) Multi-language datasets
c) Pre-trained LLM models
d) Cloud deployment
Answer: a

Q19. In an ingestion pipeline, Unstructured.io is often combined with LangChain to:
a) Parse, clean, and structure data
b) Replace embedding models
c) Train OpenAI GPT models
d) Generate synthetic datasets
Answer: a

Q20. Which format is NOT directly supported by Unstructured.io?
a) PPTX
b) DOCX

c) Embedded SQLite DBs
d) EML
Answer: c

E. Haystack – End-to-End Ingestion & Retrieval

Q21. Haystack is primarily used for:
a) Document parsing only
b) Full ingestion + retrieval + generation pipelines
c) LLM fine-tuning exclusively
d) Embedding visualization
Answer: b

Q22. Haystack's `DocumentStore` is mainly used to:
a) Store raw text
b) Store vector embeddings and metadata
c) Replace FAISS
d) Split text into chunks
Answer: b

Q23. Haystack supports which ingestion formats by default?
a) PDF
b) JSON
c) Web Pages
d) All of the above
Answer: d

Q24. Which component of Haystack orchestrates the entire ingestion and retrieval pipeline?
a) Retriever
b) Reader
c) Pipeline
d) Chunker
Answer: c

Q25. Haystack integrates with which of the following tools for better ingestion?
a) Hugging Face models

b) ElasticSearch
c) FAISS
d) All of the above
Answer: d