Multiple-Choice Questions on Tracing (OpenAI Agents SDK)

**1. What does the Tracing feature in the OpenAI Agents SDK do?**

A. Improves the model's accuracy  
B. Collects a comprehensive record of events during an agent run  
C. Stores user conversations  
D. Optimizes API usage  
**Answer:** B — Tracing collects a detailed record of events like LLM generations, tool calls, guardrails, handoffs, and custom events during an agent run

**2. Where can you visualize and debug traced agent workflows?**

A. Local logs  
B. Traces dashboard  
C. GitHub repo  
D. CLI output  
**Answer:** B — The Traces dashboard allows visualization, debugging, and monitoring of workflows

**3. Is tracing enabled by default in the SDK?**

A. No, it's disabled by default  
B. Yes, it's enabled by default  
C. Enabled only for pro users  
D. Requires manual activation each time  
**Answer:** B — Tracing is enabled by default

**4. How can you globally disable tracing?**

A. Uninstall tracing plugin  
B. Set OPENAI\_AGENTS\_DISABLE\_TRACING=1 environment variable  
C. Use a disabled flag in code  
D. Impossible to disable  
**Answer:** B — Set the environment variable OPENAI\_AGENTS\_DISABLE\_TRACING=1 to globally disable tracing

**5. How can you disable tracing for a single run?**

A. Not possible  
B. Set tracing\_disabled flag in RunConfig  
C. Use Runner.disable\_tracing() method  
D. Rename configuration file  
**Answer:** B — Set agents.run.RunConfig.tracing\_disabled = True for that specific run

**6. Can tracing be used under Zero Data Retention (ZDR) policies?**

A. Yes, fully supported  
B. No, tracing is unavailable  
C. Only partial tracing available  
D. Only metadata is recorded  
**Answer:** B — Tracing is unavailable for organizations under a ZDR policy

**7. What are traces in the SDK?**

A. Single API calls  
B. End-to-end workflow operations composed of spans  
C. Logs written to disk  
D. Only LLM model generations  
**Answer:** B — Traces represent a complete workflow and are composed of spans

**8. What information does a trace include?**

A. Only metadata  
B. workflow\_name, trace\_id, group\_id, disabled, metadata  
C. Only LLM tokens  
D. Only success/failure status  
**Answer:** B — Traces include those properties

**9. What is a span in tracing?**

A. A data structure for conversation history  
B. An operation with start/end timestamps, trace associations, and parent relationships  
C. Just a label for debugging  
D. The final output text  
**Answer:** B — Spans represent specific operations with timing, trace IDs, parent IDs, and data

**10. Which span type wraps an entire agent’s execution?**

A. trace()  
B. agent\_span()  
C. generation\_span()  
D. function\_span()  
**Answer:** B — agent\_span() wraps each agent execution

**11. Which span wraps LLM outputs?**

A. guardrail\_span()  
B. generation\_span()  
C. handoff\_span()  
D. function\_span()  
**Answer:** B — LLM generations are wrapped in generation\_span()

**12. Which span type logs calls to function tools?**

A. agent\_span()  
B. generation\_span()  
C. function\_span()  
D. handoff\_span()  
**Answer:** C — Function tool calls are wrapped in function\_span()

**13. How are guardrails tracked in tracing?**

A. Not tracked  
B. With guardrail\_span()  
C. With function\_span()  
D. Not applicable  
**Answer:** B — Guardrails are wrapped in guardrail\_span()

**14. Which span is used for handoffs between agents?**

A. handoff\_span()  
B. generation\_span()  
C. agent\_span()  
D. response\_span()  
**Answer:** A — Handoffs are wrapped in handoff\_span()

**15. Which spans are associated with audio inputs and outputs?**

A. generation\_span() & function\_span()  
B. transcription\_span() & speech\_span()  
C. agent\_span() only  
D. response\_span() only  
**Answer:** B — Audio inputs use transcription\_span(), and audio outputs use speech\_span()

**16. What span groups related audio spans?**

A. audio\_group\_span()  
B. speech\_group\_span()  
C. agent\_span()  
D. response\_span()  
**Answer:** B — Related audio spans may be grouped under a speech\_group\_span()

**17. How is the default trace named if not set explicitly?**

A. "Default trace"  
B. "Agent workflow"  
C. "Run trace"  
D. It remains unnamed  
**Answer:** B — By default, the trace is named "Agent workflow"

**18. Can you customize trace processing destinations?**

A. No, only local logs available  
B. Yes, using custom trace processors  
C. Only OpenAI dashboard supported  
D. Only export to CSV  
**Answer:** B — You can configure custom trace processors to send traces elsewhere

**19. What span data types are defined in the SDK?**

A. Only AgentSpanData and GenerationSpanData  
B. Types like Agent, Generation, Function, Guardrail, Handoff, Custom, etc.  
C. Only text spans  
D. Only audio spans  
**Answer:** B — Numerous span data classes exist, including AgentSpanData, GenerationSpanData, FunctionSpanData, GuardrailSpanData, HandoffSpanData, CustomSpanData, SpeechSpanData, TranscriptionSpanData, ResponseSpanData, and SpeechGroupSpanData

**20. What does CustomSpanData include?**

A. Only timestamps  
B. A name and arbitrary data bag  
C. Model configuration only  
D. Input and output only  
**Answer:** B — CustomSpanData consists of a name plus a data dictionary for arbitrary custom events