Multiple-Choice Questions on Models (OpenAI Agents SDK)

**1. Which model flavor is recommended in the Agents SDK?**

A. OpenAICompletionModel  
B. OpenAIResponsesModel  
C. OpenAIRegularModel  
D. OpenAIStreamModel

**Answer:** B — The recommended model is the **OpenAIResponsesModel**, which calls the Responses API

**2. What alternative model flavor is supported besides Responses?**

A. OpenAIOpenModel  
B. OpenAICreateModel  
C. OpenAIChatCompletionsModel  
D. OpenAIStreamingModel

**Answer:** C — The alternative supported flavor is **OpenAIChatCompletionsModel**, which calls the Chat Completions API

**3. What is the default OpenAI model used if none is specified?**

A. gpt-4  
B. gpt-4o  
C. gpt-4.1  
D. gpt-5

**Answer:** C — By default, the SDK uses **gpt-4.1** due to its balance of predictability and latency

**4. How can you override the default model for all agents without specifying it in each Agent?**

A. DEFAULT\_MODEL = "..." in code  
B. Set the environment variable OPENAI\_DEFAULT\_MODEL  
C. Call Agent.set\_default\_model()  
D. Use Configuration.default\_model

**Answer:** B — You can set the OPENAI\_DEFAULT\_MODEL environment variable

**5. What happens when you use a GPT-5 reasoning model in the SDK?**

A. Custom ModelSettings must always be provided  
B. SDK defaults to "minimal" reasoning effort  
C. SDK applies sensible defaults: "low" reasoning and "low" verbosity  
D. SDK disables structured outputs

**Answer:** C — The SDK applies default ModelSettings with both reasoning.effort and verbosity set to "low" for GPT-5 models

**6. Which GPT-5 model names are supported with these defaults?**

A. gpt-5-large, gpt-5-small  
B. gpt-5, gpt-5-mini, gpt-5-nano  
C. gpt-5-x, gpt-5-y  
D. Only gpt-5

**Answer:** B — Supported models include gpt-5, gpt-5-mini, and gpt-5-nano

**7. For lower latency use, which model paired with what setting is recommended?**

A. gpt-5 with "high" reasoning effort  
B. gpt-5-mini or gpt-5-nano with reasoning.effort="minimal"  
C. gpt-4.1 with default settings  
D. Any model with temperature=1.0

**Answer:** B — Use gpt-5-mini or gpt-5-nano with reasoning.effort="minimal" for faster response; note some tools may not support minimal effort

**8. What happens if you pass a non–GPT-5 model name without custom model\_settings?**

A. SDK throws a configuration error  
B. SDK defaults to generic ModelSettings compatible with any model  
C. SDK treats it as GPT-5 during runtime  
D. SDK disables tracing

**Answer:** B — The SDK uses generic ModelSettings compatible with any model

**9. Which integration allows use of non-OpenAI models via a single interface?**

A. TensorFlow integration  
B. HuggingFace integration  
C. LiteLLM integration  
D. CustomREST integration

**Answer:** C — The SDK includes the **LiteLLM integration** for over 100 non-OpenAI models

**10. How are non-OpenAI models specified?**

A. Just the model name without prefix  
B. "litellm/<provider>/<model\_name>" with optional API key  
C. "openai/<model\_name>"  
D. Stored in a YAML config

**Answer:** B — Models use "litellm/<provider>/<model\_name>" syntax

**11. What is one alternative to LiteLLM for using non-OpenAI models?**

A. Writing raw HTTP requests manually  
B. Setting set\_default\_openai\_client with a custom base URL and API key  
C. Subclassing Agent directly  
D. Exporting a JSON model config

**Answer:** B — You can globally set an OpenAI-compatible client via set\_default\_openai\_client

**12. What does ModelProvider allow you to configure?**

A. Default reasoning strategy  
B. Tool loading logic  
C. Custom model provider at Runner.run level  
D. UI theme

**Answer:** C — It lets you set a custom model provider for all agents within a run

**13. What can be specified at the individual Agent level for model selection?**

A. Only an integer code  
B. Only default model  
C. The model parameter directly (name or Model implementation)  
D. No override allowed

**Answer:** C — You can set the model argument individually per Agent using a name or a Model instance

**14. What are the three ways to configure a specific model for an Agent?**

A. Name only; version; author  
B. Name only; name + ModelProvider; providing a Model implementation  
C. Loyalty tier; API token; prompt style  
D. URL; binary; JSON

**Answer:** B — You can pass a model name, name with a custom ModelProvider, or a direct Model implementation

**15. What warning does the SDK include about mixing model shapes in a workflow?**

A. It will slow down execution  
B. You may lose traceability  
C. Different shapes (Responses vs ChatCompletions) support different features; ensure feature compatibility  
D. It only works for GPT-5 models

**Answer:** C — Mixing model shapes can lead to loss of features like certain tools; ensure compatibility

**16. If you're using the Responses API but choose a provider that doesn't support it, what might happen?**

A. Error 404 may occur  
B. Automatic fallback to GPT-4  
C. SDK silences the error  
D. Model switches internally to ChatCompletions

**Answer:** A — You may get 404 errors

**17. How can you work around lack of Responses API support in some providers?**

A. Use concurrent requests  
B. Call set\_default\_openai\_api("chat\_completions") or use OpenAIChatCompletionsModel  
C. Disable API checks  
D. Wrap in a try-except

**Answer:** B — Either switch default to Chat Completions via environment or use OpenAIChatCompletionsModel

**18. What structured output issue might arise when using non-OpenAI providers?**

A. Models don't respond at all  
B. Model outputs may be invalid JSON due to missing schema support  
C. Responses are too verbose  
D. Models crash

**Answer:** B — Some providers don't support structured JSON schema, causing invalid responses

**19. What feature is not universally supported across all LLM providers?**

A. JSON schema outputs, multimodal input, hosted tools  
B. Text-only prompts  
C. Numeric token limits  
D. Keyword filtering

**Answer:** A — Many providers lack support for structured outputs, multimodal input, or certain hosted tools

**20. What best practice should developers follow when mixing providers?**

A. Ignore error logs  
B. Avoid using LiteLLM  
C. Filter out unsupported tools and multimodal inputs; use schema-supported providers  
D. Always use local dev models

**Answer:** C — Developers should filter unsupported features when mixing providers