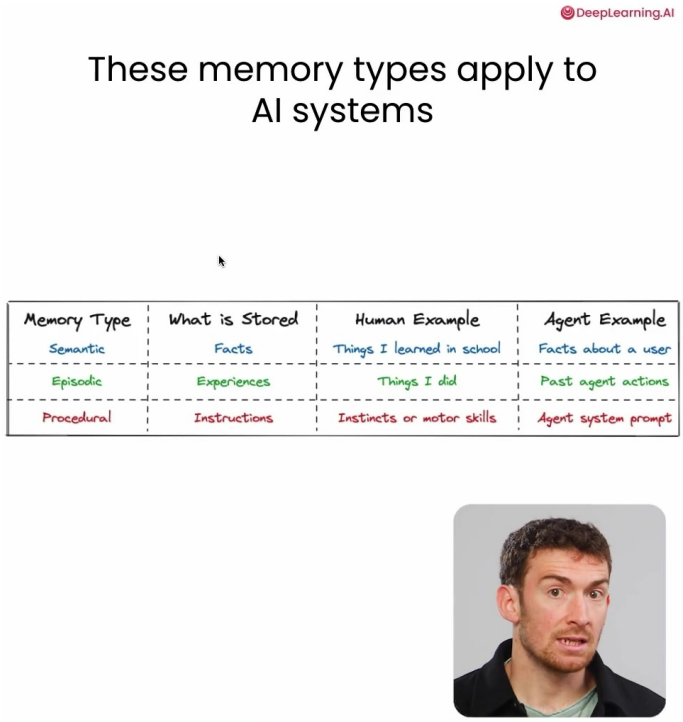
一、具有语义+情景+程序记忆的邮件助手 *00:02*

1. 程序记忆 00:10

-
- **定义**:　程序记忆是第三种也是最后一种记忆类型，以系统提示的形式存在于主代理和分类代理中。
- **特点**: 包含关于如何做事的指令，在代理中通常以提示指令的形式表示。
- **实现方式**: 通过自动更新提示指令来实现，而非使用硬编码值。

2. 内容回顾 00:19



- **语义记忆**: 存储事实信息，如用户资料、会议偏好、VIP联系人等
- **情景记忆**: 存储经验信息，如过去的代理操作，以少量示例形式存在
- **程序记忆**: 存储指令信息，如系统提示、写作风格、语气偏好等

3. 邮件助手构建 00:41

1）邮件助手构建的步骤

- **初始化**: 加载环境变量，定义用户配置和提示指令
- **存储设置**: 使用长期记忆存储而非硬编码值
- **更新机制**: 允许从外部更新提示指令并自动应用到未来运行中

2）邮件助手构建的代码 01:13

```
email = {
    "from": "Alice Smith <alice.smith@company.com>",
    "to": "John Doe <john.doe@company.com>",
    "subject": "Quick question about API documentation",
    "body": """
Hi John,

I was reviewing the API documentation for the new authentication serv

Specifically, I'm looking at:
- /auth/refresh
- /auth/validate

Thanks!
Alice""",
}
```

```
from langgraph.store.memory import InMemoryStore
```

```
store = InMemoryStore(
    index={"embed": "openai:text-embedding-3-small"}
)
```

```
# Template for formating an example to put in prompt
template = """Email Subject: {subject}
Email From: {from_email}
Email To: {to_email}
Email Content:
```
{content}
```
> Triage Result: {result}"""
```

-
- **核心组件**:
  - 分类路由器(triage_router)
  - 响应代理(response_agent)
  - 状态图(StateGraph)
- **工具集**: 包含写邮件、安排会议、检查日历可用性等工具

3）邮件助手构建的状态定义 01:47

- **状态对象**: 包含email_input和messages两个字段
- **命名空间**: 使用langgraph用户ID作为程序记忆的命名空间
- **默认值处理**: 当存储中不存在指令时自动添加默认值

4）修改取指令方式 02:28

```python
def triage_router(state: State, config, store) -> Command[
    Literal["response_agent", "__end__"]
]:
    author = state['email_input']['author']
    to = state['email_input']['to']
    subject = state['email_input']['subject']
    email_thread = state['email_input']['email_thread']

    namespace = (
        "email_assistant",
        config['configurable']['langgraph_user_id'],
        "examples"
    )
    examples = store.search(
        namespace,
        query=str({"email": state['email_input']})
    )
    examples=format_few_shot_examples(examples)

    system_prompt = triage_system_prompt.format(
        full_name=profile["full_name"],
        name=profile["name"],
        user_profile_background=profile["user_profile_background"],
        triage_no=prompt_instructions["triage_rules"]["ignore"],
        triage_notify=prompt_instructions["triage_rules"]["notify"],
        triage_email=prompt_instructions["triage_rules"]["respond"],
        examples=examples
    )
    user_prompt = triage_user_prompt.format(
        author=author,
        to=to,
        subject=subject,
        email_thread=email_thread
```

- **更新逻辑**: 首先尝试从存储获取指令，不存在则存入默认值
- **分类规则**: 更新ignore、notify、respond三种分类提示
- **优势**: 指令更新后会自动应用到未来运行中

5）邮件助手构建的工具 04:59

- **基础工具**: write_email、schedule_meeting、check_calendar_availability
- **记忆工具**: manage_memory_tool、search_memory_tool
- **工具集成**: 通过create_react_agent函数集成所有工具

6）邮件助手构建的逻辑修改 05:20

- **系统消息更新**: 从长期记忆存储获取最新提示指令
- **代理创建**: 使用更新后的提示创建邮件代理
- **优势**: 代理行为可通过更新存储中的指令而改变

7）邮件助手构建的代码 06:17

```python
from langgraph.prebuilt import create_react_agent
```

```python
tools= [
    write_email,
    schedule_meeting,
    check_calendar_availability,
    manage_memory_tool,
    search_memory_tool
]
response_agent = create_react_agent(
    "openai:gpt-4o",
    tools=tools,
    prompt=create_prompt,
    # Use this to ensure the store is passed to the agent
    store=store
)
```

```python
email_agent = StateGraph(State)
email_agent = email_agent.add_node(triage_router)
email_agent = email_agent.add_node("response_agent", response_agent)
email_agent = email_agent.add_edge(START, "triage_router")
email_agent = email_agent.compile(store=store)
```

- **核心函数**: create_prompt负责生成系统提示
- **配置处理**: 从config获取langgraph用户ID
- **错误处理**: 处理存储中不存在指令的情况

8）应用案例 06:48

- 例题:更新邮件助手指令

```python
store.get(("lance",), "agent_instructions").value['prompt']
```

```
"Use these tools when appropriate to help manage John's tasks effici
ently."
```

9）题目解析

- **更新过程**: 使用create_multi_prompt_optimizer优化多个提示

- **反馈格式**: 最简单的反馈格式是字符串形式
- **更新条件**: 根据when_to_update值决定是否更新提示
- 例题:优化邮件助手指令 08:18

```
from langmem import create_multi_prompt_optimizer
```

```
conversations = [
    (
        response['messages'],
        "Always signs your emails `John Doe`"
    )
]
```

10）题目解析

- **优化器创建**: 使用Claude-3-Sonnet模型进行提示优化
- **提示定义**: 每个提示包含名称、值、更新指令和更新条件
- **更新逻辑**: 先判断是否需要更新，再应用更新指令生成新提示
- 例题:更新邮件助手指令 12:57

```
for i, updated_prompt in enumerate(updated):
    old_prompt = prompts[i]
    if updated_prompt['prompt'] != old_prompt['prompt']:
        name = old_prompt['name']
        print(f"updated {name}")
        if name == "main_agent":
            store.put(
                ("lance",),
                "agent_instructions",
                {"prompt":updated_prompt['prompt']}
            )
        else:
            raise ValueError
```

```
updated main_agent
```

```
store.get(("lance",), "agent_instructions").value['prompt']
```

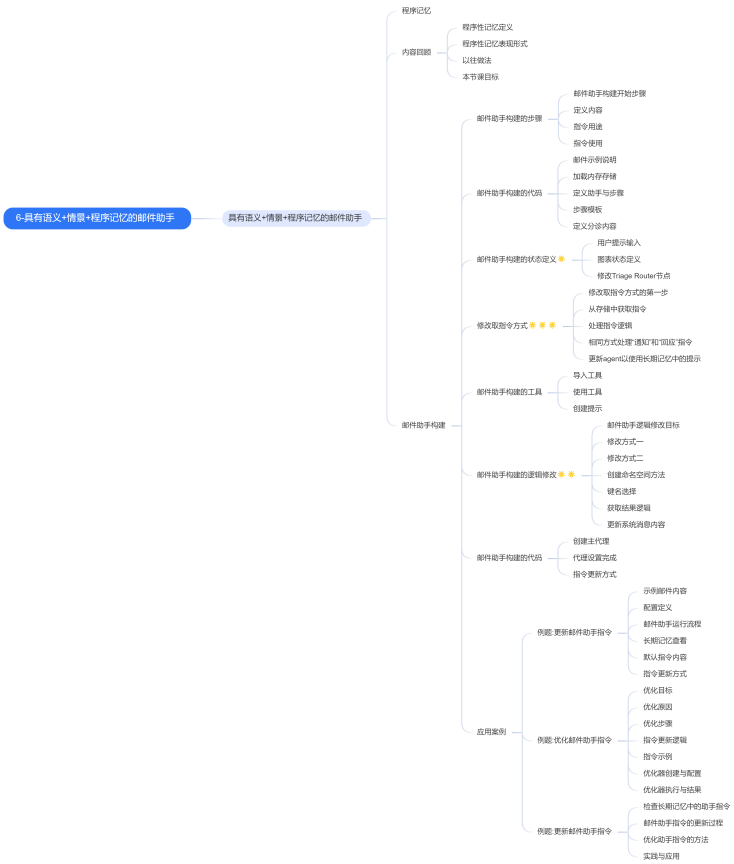
```
'Use these tools when appropriate to help manage John\'s tasks effic
iently. When sending emails, always sign them as "John Doe".'
```

11）题目解析

- **结果验证**: 检查更新后的提示是否包含反馈内容
- **效果测试**: 重新运行代理验证新指令是否生效
- **存储检查**: 确认长期记忆中的指令已更新

二、知识小结

知识点	核心内容	考试重点/易混淆点	难度系数
程序性记忆（Procedural Memory）	通过系统提示（prompt instructions）形式实现，指导AI如何执行任务（如忽略/通知/回复邮件规则）	**硬编码提示 vs. 动态更新提示**的存储与调用逻辑	★★★
长期记忆存储（Long-term Memory Store）	使用命名空间（如用户ID）和键值对（如triage_ignore）存储动态提示，支持默认值回退机制	命名空间格式要求（必须为元组）及键名设计规范	★★
多提示优化器（Multi-prompt Optimizer）	基于用户反馈（如"总署名为John Doe"）自动更新相关提示，采用Claude 3 Sonnet模型优化	**反馈触发条件**（when_to_update）与**更新指令**（update_instructions）的匹配逻辑	★★★★
代理行为更新流程	1. 从记忆库检索当前提示; 2. 对比反馈生成新提示; 3. 仅更新差异部分至记忆库	更新判定条件（if updated_prompt != old_prompt）的边界处理	★★★
动态提示应用验证	通过重新运行代理验证更新效果（如新增签名规则后自动署名）	记忆库键值（agent_instructions）与代理节点的数据流同步	★★


6-具有语义+情景+程序记忆的邮件助手
具有语义+情景+程序记忆的邮件助手
程序记忆
内容回顾
程序性记忆定义
程序性记忆表现形式
以往做法
本节课目标
邮件助手构建
邮件助手构建的步骤
邮件助手构建开始步骤
定义内容
指令用途
指令使用
邮件助手构建的代码
邮件示例说明
加载内存存储
定义助手与步骤
步骤模板
定义分诊内容
邮件助手构建的状态定义
用户提示输入
图表状态定义
修改Triage Router节点
修改取指令方式
修改取指令方式的第一步
从存储中获取指令
处理指令逻辑
恒可方式处理"通知"和"回应"指令
更新agent以使用长期记忆中的提示
邮件助手构建的工具
导入工具
使用工具
创建提示
邮件助手构建的逻辑修改
邮件助手逻辑修改目标
修改方式一
修改方式二
创建命名空间方法
键名选择
获取结果逻辑
更新系统消息内容
邮件助手构建的代码
创建主代理
代理设置完成
指令更新方式
应用案例
例题:更新邮件助手指令
示例邮件内容
配置定义
邮件助手运行流程
长期记忆查看
默认指令内容
指令更新方式
例题:优化邮件助手指令
优化目标
优化原因
优化步骤
指令更新逻辑
指令示例
优化器创建与配置
优化器执行与结果
例题:更新邮件助手指令
检查长期记忆中的助手指令
邮件助手指令的更新过程
优化助手指令的方法
实践与应用