实验 2：请求生命周期观察

1. 本节定位

本实验把第 3 讲中的生命周期概念落到可观测代码路径和阶段日志上。你不需要在本节优化系统，但需要把一次请求拆成阶段，并用证据说明 prefill 和 decode 为什么不能混成同一种工作负载来理解。

2. 核心问题

为什么长 prompt 首先放大的通常是 prefill 代价，而不是所有阶段都被平均放大?

3. 学习目标

完成本节后，你应能够：

- 把一次请求拆成创建、排队、prefill、decode、完成、释放几个阶段。
- 找出这些阶段在代码里的近似锚点。
- 通过长短 prompt 对比，说明 TTFT 和 decode 行为为何不同。
- 画出一张能够被日志支撑的请求生命周期时序图。

4. 开始前

4.1 先修要求

- 已完成实验一。
- 已经知道请求入口和主循环的大概位置。
- 能区分 prompt 长度与输出长度。

4.2 本节材料

- 理论配套：build/tutorials/Tutorial_02_请求生命周期与PrefillDecode.pdf
- 参考代码：src/code/nano-vllm-hust/nanovllm/engine/llm_engine.py
- 状态对象：src/code/nano-vllm-hust/nanovllm/engine/sequence.py
- 学生练习：src/code/nano-vllm-hust/student_exercises/nanovllm/engine/llm_engine.py
- 作业包：src/experiments/nano-vLLM实验课/experiment_2_请求生命周期观察/assignment_spring-2027/

4.3 环境检查

- 教学环境至少能完成两组不同 prompt 长度的运行。
- 你可以找到日志输出或自行加入轻量日志。
- 你知道如何控制模型、输出长度和 batch 条件。

5. 提交内容

本节提交物必须包含：

1. 一张请求生命周期时序图。
2. 一张长短 prompt 对比表。
3. 一段约 500 字分析，解释 prefill 和 decode 的区别。
4. 一段日志摘录，证明阶段划分不是凭空假设。

6. 时间安排建议

- 10 分钟：回顾实验一的请求入口与主循环。
- 20 分钟：确认阶段划分与插桩点。
- 25 分钟：记录阶段时间与长短 prompt 对照。
- 20 分钟：绘制时序图并写出解释。
- 10 分钟：整理提交材料。

7. 实验任务

任务 1：明确阶段划分

在你的代码理解中明确以下阶段：

1. 请求创建。
2. 请求入队。
3. prefill 开始。
4. prefill 结束。
5. decode 循环开始。
6. decode 循环结束。
7. 请求完成。
8. 状态释放。

如果代码中没有天然阶段标签，可以通过日志人为划分，但必须写清近似依据。

任务 2：确定日志记录点

建议插桩点：

- request enqueue
- prefill begin
- prefill end
- decode step begin
- decode step end
- request finish
- state free

建议日志格式：

```
[timestamp] request_id stage prompt_len generated_len batch_size
```

记录要求：日志必须足够区分阶段，不能只记录总时长。

任务 3：整理长短输入对照

至少准备两组输入：

- 短输入：16-64 tokens
- 长输入：512-2048 tokens

控制条件：

- 使用同一模型。
- 尽量固定输出长度。
- 说明是否使用相同 batch 条件。

任务 4：记录分阶段结果

请填写下表：

case	prompt_len	output_len	total_latency	TTFT	avg_step	prefill 是否主导	解释
short							
long							

填写要求：解释必须回到阶段差异，不能只写“长输入更慢”。

任务 5：绘制请求生命周期图

时序图至少包含以下对象：

- Client
- Request Queue
- Scheduler / Engine Loop
- Model Runner
- Sequence / KV State
- Response

要求明确标出控制权切换位置，以及状态在哪里被更新。

8. 证据要求

以下情况不能视为合格证据：

- 用总时长替代阶段分析。

- 插桩点太少，无法区分 prefill 和 decode。
- 图里画了很多对象，但没有标出状态更新位置。
- 长短 prompt 对比没有控制输出长度或 batch 条件。

9. 提交核对

- 已附生命周期时序图。
- 已附长短 prompt 对比表。
- 已附日志摘录。
- 已说明阶段划分的近似边界。
- 已回答 prefill 与 decode 为什么不能混看。

10. 评分关注点

- 阶段划分是否清楚。
- 日志与时序图是否对应。
- 解释是否回到 TTFT、prefill 和 decode 的差异。
- 是否诚实说明了近似划分的边界。

11. 与后续实验的衔接

本节的阶段日志和时序图会直接进入后续实验：

- 实验三：观察调度如何改变谁先进入下一阶段。
- 实验五：把阶段日志纳入规范实验记录。

12. 提交模板

12.1 基本信息

姓名 / 组员:
学号:
实验日期:
模型与环境:

12.2 日志记录点

stage	文件 / 函数 / 位置 你的近似依据
request enqueue	
prefill begin	
prefill end	
decode step	
request finish	
state free	

12.3 长短输入对照表

case	prompt_len	output_len	total_latency	TTFT	avg_step	prefill 是否主导	解释
short							
long							

12.4 生命周期图说明

时序图文件名 / 页面编号:

12.5 日志摘录

```
[timestamp] request_id stage prompt_len generated_len batch_size
```

12.6 分析说明

请用约 500 字说明：prefill 和 decode 的主要差异是什么？长 prompt 放大的主要是哪一阶段？你的证据是什么？