ISTP-A 行为操作系统（Behavioral OS）对照手册

> **目的：**作为你日常执行、决策、规划、切换任务、避免拖延与保持动力的可视化锚点。所有内容基于 ISTP 的认知结构（Ti–Se–Te）设计。

---

1. ISTP 的核心认知模型：Ti–Se–Te 的对立统一

Ti：内向思考（主导功能）

- 逻辑、结构、理解本质。
- 喜欢拆解、精确、内部一致性。
- 过强时：过度分析、完美主义、启动困难。

Se：外向感知（辅助功能）

- 动手、体验、即时反馈。
- 高反应力、可快速进入状态。
- 过弱时：行动动力不足；过强时：只做短期快感。

Te：外向思考（第三功能）

- 执行、效率、结果导向。
- 不追求完美，只要"能跑"。
- 过弱时：行动少、系统性不足；过强时：粗糙急躁。

ISTP-A 的理想模式 = Ti ↔ Se ↔ Te 循环

1. **Ti**：找到结构 / 原理 / 关键点。
2. **Se**：从最小行动开始，立即动手验证。
3. **Te**：整合反馈 → 调整策略 → 推进下一步。

> **一句话总结：** ISTP 的效率来自不断的"微结构 → 微行动 → 微反馈"循环，而不是长期大规划。

---

2. ISTP-A 行为操作系统（Behavioral Operating System）

BOS = Playground → Modules → System → Pipeline → Context Snapshot

1）Playground（探索层）

- 不承诺、不结构化，不追求完美。
- 玩想法、玩动作、玩代码、玩原型。
- 让 Se 感到“有趣 → 想继续”。

2）Modules（模块层）

- 所有任务都拆成 **最小可独立完成模块**：
- 每个模块 5–45 分钟。
- 每个模块内可自测、自运行、自验证。
- 不依赖全局系统即可推进。

3）System（集成层）

- 只包含结构：
- 项目骨架、文件目录、主流程、config、registry。
- 禁止写细节（避免更改某模块导致系统崩溃）。

4）Pipeline（执行层）

- 每周推进每个重点项目 1–2 个模块。
- 每天推进 1 小模块（不做跨项目多线程）。
- 保持渐进式复利。

5）Context Snapshot（上下文快照）

每次结束任务时写两行：

```
progress: 我做到哪里？
next entry: 下一次从哪里开始？（必须非常小）
```

- 这是 ISTP 防止“冷启动失败”和“忘记上下文”的核心。

---

3. ISTP-A 多任务优先级系统（POPS）

使用五维排序，而不是传统工具：

1. **Cost of Inaction（不做的损耗）**
2. **Dependency（系统依赖）**
3. **Return / Leverage（回报/杠杆）**
4. **Visibility（可见反馈）**
5. **Friction（启动阻力，反向因素）**

优先级公式：

```
Priority = 0.35*C_inaction + 0.25*Dependency + 0.15*Return + 0.10*Visibility -
0.15*Friction
```

执行方式： - 每周只推进 **优先级最高的 1–2 个项目**。 - 次要项目保持“松散推进”。 - 每两周重算优先级。

---

4. 各任务类型的 ISTP-A 执行模板

以下是你之前所有情景的系统化版本。

📌A. 写 CS 会议论文（Paper BOS）

Playground

- notebook 试模型 / 试 prompt / 写碎片化段落。

Modules

- Method 子节
- 实验图表
- 结果段落
- RW 子节
- 伪代码
- 实验 config

System

- 论文 skeleton
- 3 主方向：Method / Experiments / Narrative
- 每个实验 = 一份 config

Pipeline

- 每周 2–3 个模块
- 每天 3 分钟入口（改一句话 / 加一张图）

Snapshot

- “RW 到第 2 段，下次补 related 工作 X。”

📌B. 大型代码框架（实验 codebase / SaaS / 工程）

Playground

- Jupyter 测试功能
- 小 demo script

Modules

- models/
- trainers/
- datasets/
- utils/
- auth/ payment/ RAG/ UI 等功能子模块

每个模块：**self-contained + 可独立运行 + 不依赖系统**。

System

- registry（集中接入）
- config（参数中心）
- pipeline（train / run / deploy 入口）
- 禁止写细节算法

Pipeline

- 新 idea → playground → module → registry → config → run

📌C. 求职（MLE / AE / RE）

Playground

- 浏览 JD
- 看面经
- 收集岗位要求

Modules

- 简历模块
- 项目模块
- 技术补全模块
- STAR 模块
- 公司匹配模块
- Mock 面试模块

System

- 求职 config（role, skills required, visa, stage）
- 求职主循环（apply → interview → feedback → adjust）

📌D. 健身

Playground

- 玩动作、器械、轻量 cardio

Modules

- Push
- Pull
- Legs
- Core
- Cardio
- Mobility

每个 5–20 分钟，任意选择一个即可。

System

- 每周 3–5 模块
- 不做固定日程

Entry

- 10 个俯卧撑 / 1 分钟跑步 / 15 深蹲

📌E. PhD Thesis

Playground

- 碎片化写段落
- 图表草稿
- related work notes

Modules

- 每章拆成 5–15 小节
- 每节一个 md/tex 文件
- 每个图一个占位

System

- thesis skeleton 可编译 PDF
- 每章结构 + 图表目录

📌F. EB1A / O1 移民申请

Playground

- 证据碎片
- 项目事实记录
- 推荐信素材

Modules

- Criterion X 子节
- Exhibit 说明
- 推荐信草稿
- 时间线
- PR 子段落

System

- Petition Letter skeleton
- Exhibit master list
- 推荐信格式

📌G. 创业（AI SaaS）

Playground

- 试 API
- 试 prompt
- demo UI

Modules

- 技术子模块（RAG/embedding/auth/payment）
- UI 模块
- 用户模块
- 增长模块

System

- app routing
- api pipeline
- registry
- config
- deploy 脚本

---

5. ISTP-A Daily / Weekly 模式

Daily（每天）

1. 做一个小模块（10–30 分钟）
2. 使用“3 分钟启动法”
3. 写 snapshot

Weekly（每周）

1. 优先级最高的两个项目 → 各推进 1–2 模块
2. 其他项目 → 松散推进一次（保持不掉线）
3. 小型 review（调整模块树）

Bi-weekly（每两周）

- 重算优先级（POPS）
- 重构 System 层（轻微）
- 删除失效任务（保持系统简洁）

---

6. ISTP-A 的三句核心心法

1. **先跑 baseline，让系统先能运行。**
2. **从最小模块开始，保持低摩擦推进。**
3. **用 snapshot 确保可以随时恢复上下文。**

这就是你的 ISTP-A 操作系统。

你可以每天看一眼这份手册，让它作为你的 Se 提示和行为锚点。