🖥️ AP Computer Science A 수업 개요 (총 3 회, 45 분 수업 기준)

과목 특징

- Java 언어를 기반으로 한 본격적인 프로그래밍 수업
- 대학 컴퓨터공학 1 학년 수준의 내용을 선이수
- 문제 해결력, 알고리즘 사고, 객체지향 설계 능력 강화

수업 목표

- AP 시험 구조 및 출제 경향 파악
- 프로그래밍 핵심 개념 (조건문, 반복문, 클래스 등) 학습
- 학생 스스로 문제를 분석하고, 실제 코드로 구현할 수 있도록 지도

수업 구성

회차	주제	주요 내용
1 회	프로그래밍은 왜 중요한가?	- 대학과 실무에서의 활용 사례 소개- Java 언어와 AP 시험 구조 설명- 전공/비전공에 관계없이 필요한 컴퓨터 사고력 강조
2 회	AP CS A 의 주요 난이도 분석	- 객체지향 개념, 배열/리스트, 재귀 등의 학습 포인트- 초보자가 실수하기 쉬운 오류 유형- 강사의 실무 경험을 토대로 한 효과적인 학습 전략 제시
3 회	스스로 앱을 만들 수 있는 사람 되기	- 매일 학습 루틴 구성법- Leetcode, VSCode 등 추천 리소스 소개- 간단한 프로젝트 시연 및 학생 질문 응답

강사 소개

10 년 이상 웹/서버 풀스택 개발자 경력 보유.

실제 서비스와 프로젝트를 기반으로, 단순 이론이 아닌 실용적 지식을 전달.

🌐 AP Computer Science Principles 수업 개요 (총 3 회, 45 분 수업 기준)

과목 특징

- 컴퓨터 과학의 기초 개념 중심의 과목 (프로그래밍 비중은 비교적 낮음)
- 데이터, 네트워크, 알고리즘 등 디지털 사회 핵심 개념 학습
- 실제 프로그램 제작(Create Task)을 통해 창의적 문제 해결 경험 제공

수업 목표

- AP 시험 구조 및 평가 요소 이해
- 컴퓨팅 사고력과 정보 처리 능력 향상

- 실생활 문제를 디지털 방식으로 해결할 수 있는 능력 개발

수업 구성

회차	주제	주요 내용
1 회	누구나 알아야 할 디지털 언어	- 비전공자도 활용 가능한 코딩 사례 소개- AP CSP 의 커리큘럼 구조 및 시험 방식 설명- 다양한 전공과 연계 가능한 컴퓨팅 활용 예시 제시
2 회	추상화와 알고리즘, 어렵지 않다	- 핵심 개념(추상화, 알고리즘, 데이터 등) 시각적 설명- Create Task 수행 전략 및 주제 선정법 소개- 강사 예시 프로젝트 공유 (간단한 앱/웹 인터페이스)
3 회	나만의 프로그램, 나만의 결과물	- 단계별 프로젝트 계획 수립 방법 안내- 학습 루틴 및 추천 리소스 정리- 개별 질문 응답 및 선택 주제 토론 활동

강사 소개

풀스택 개발자 및 시스템 운영 경험 보유.

AP CSP 에서 요구하는 프로그램 구현 및 문서 작성 전 과정 실무 수준 지도 가능.

🔗 참고 사항

- 학생들은 A 와 Principles 중 한 과목만 선택하는 것이 일반적이며, 본 수업들은 각각 독립적으로 설계되어 있습니다.
- 수업은 한국어로 진행되며, 필요시 기초 **Java** 코드**,** 영어 설명자료 번역 지도 포함 가능합니다.

明白了！你的想法比刚才我给出的方案**更成熟**：
你希望**不仅仅讲课程内容**，而是要通过：

- 举**生活化/未来应用的例子** ➔ 让学生知道“编程不仅是考试，它关乎未来”
- 展示**编程技能的实际价值** ➔ 让学生感觉学了就能在现实中用起来
- 强调**老师本人（你）丰富的实战经验** ➔ 让学生意识到：“我能直接从一个真正会的人身上学到真正有用的东西”，不是只为考试而学。

✅ 这样组合，**会更打动人、提高报名意愿**。

结合你的思路，我重新给你框一个整体风格

（注意：这套比之前的“行政版本”更有画面感和说服力，真正是“想让学生来上课”的写法）

统一思路（适用于 AP CS A & Principles 开场）

第一节课的前半段（15 分钟）应该**这样引导**：

> 想象一下，你在未来读大学，无论是经济、心理学还是传媒专业，都需要处理大量数据。你需要用 Python 分析数据，或者写简单程序整理调查结果。
>
> 这时，同班同学还在苦学基础编程，你已经在高中时，通过 AP 课程，掌握了写程序的核心能力。
>
> 更重要的是，编程，不只是考试技能。
>
> - 它能让你更快解决问题
> - 它能让你自动化你的生活
> - 它能让你比别人更早接触到人工智能、大数据、网络安全这样的未来行业
>
> 而这门课，不只是教你怎么应付考试。
> 我本人有多年**全栈开发经验、服务器维护经验**，我会教你**怎么用真正的方法学编程**，而不是死记硬背。
>
> 无论你以后是理工、商科、艺术，编程，都会成为你未来的第二语言。
>
> 这门课，是你为未来自己准备的一份特别的礼物。

🔵 然后再顺势讲考试内容（比如 MCQ 部分怎么考，Create Task 怎么做）。
🔵 中间穿插真实例子（比如：“Python 在心理学数据分析中的应用”、“网页制作在市场营销中的应用”）。

每节课加上的要素（升级版结构）

节次	内容	加强版思路
第 1 节	未来愿景、编程重要性、AP 考试结构	用未来生活举例 + 展示实用技能意义 + 带出考试内容
第 2 节	课程难点分析	告诉他们“编程难在思维改变，但有高手带，走得更快更稳”
第 3 节	资源与支持系统	告诉他们“有丰富实战经验的老师做后盾”，并规划清晰学习路径

再细化一点的话，比如具体场景举例：

方向	例子	解释方式
数据科学	"心理学专业需要用 Python 跑问卷分析"	让文科生也觉得编程有用
商业	"创业做一个简单的商品销售网站，你需要写后端"	让有商业梦的学生感兴趣
艺术与创意	"动画、游戏开发、美术数字工具全离不开脚本语言"	让兴趣生找到编程关联点
AI 与未来行业	"懂代码的人，比不会的人更快适应 AI 工具，比如自己训练小模型"	让想走前沿科技的学生被吸引

你的个人介绍（强力加分项）

一定要在第一节课或者简介里加一句：

> 作为你的老师，我拥有多年全栈开发经验，参与过实际网站、应用和服务器系统搭建。
> 这意味着，我能教你的，不是纸上谈兵的知识，而是真正“如何把代码变成现实中能用的东西”。

学生听到这里，会感觉：
"这个老师不是念课本，是能带着我做出东西的人。"

📈 这样学生的信任感和期待感自然就起来了。

✨ 总结

你真正需要设计的是一种：

- 【愿景驱动】 ➔ 编程=未来不可缺少的技能
- 【个人价值背书】 ➔ 你是有真实开发背景的老师
- 【考试融入其中】 ➔ 学了不仅为了考试，而是掌握真正的技能

而不是单纯介绍大纲内容！

如果你觉得这个思路对了，我可以继续帮你：

- 直接给出**每节课具体怎么讲（分分钟安排）**
- 写一份**开课宣传文案**，可以发给学生/家长，提升报名率

要继续吗？
如果要的话告诉我：“继续给出每节课安排”，或者“给我开课宣传文案”，我可以很快接着给你！🎯

要继续吗？🔥

好，Yuki！我现在为你提供一份**具体到每节课的分钟安排**、讲什么、说什么例子、用什么方式引导的详细课程内容。以下是针对 **AP Computer Science A** 的 3 节课，紧接着我会再给你 **AP Computer Science Principles** 的版本。

🖥️ AP Computer Science A 入门课程内容（每节 45 分钟）

📘 第 1 节：未来的你需要代码

目标：激发学生兴趣 + 初步了解 AP CS A 考试结构

时间安排与内容：

时间	内容
0:00 - 5:00	轻松开场：问大家有没有用过 ChatGPT、玩过 Minecraft 模组、看过 AI 绘画，顺势引入：“这一切背后都离不开代码。”
5:00 - 15:00	举例：- 大学心理学分析问卷 ➔ Python- 新闻系用数据画图 ➔ Excel + JavaScript- 甚至文科生也逃不开写程序处理信息结论：**编程是未来的第二语言**
15:00 - 25:00	简介课程内容：- Java 语法基础- 类和对象（面向对象编程）- 数据结构（数组、ArrayList）- 控制流程与递归配图：课程结构树状图
25:00 - 35:00	讲解考试结构：- 40 道选择题（MCQ）- 4 道编程题（FRQ），含类设计题和数组处理题- 总时长 3 小时附加：College Board 打分规则
35:00 - 40:00	展示一个简单 Java 程序（"Hello, name!"）并改写：输入名字 → 输出问候语
40:00 - 45:00	总结：“学会这门课，你可以写出小游戏、编程机器人、或为你的人生简历添一行极有力的技能。”Q&A 时间

📕 第 2 节：为什么说“编程难”？但你不是一个人

目标：识别难点 + 建立信任感 + 展示你能提供的帮助

时间	内容
0:00 - 5:00	复习上节课：Java 课程内容 + 考试
5:00 - 10:00	提问：“你觉得编程难在哪？”（收集关键词写在黑板上）
10:00 - 25:00	深度讲解三大难点：1. **OOP 面向对象**（类、对象、继承）2. **语法复杂性**（Java 比 Python 更严谨）3. **抽象逻辑能力要求高**（如何理解递归、条件分支）举例：- Java 类 = 现实世界中的“对象工厂”- 递归 = “拿镜子对着镜子”一样的概念
25:00 - 35:00	自学 vs 有人带的区别：- 没人帮你看错误，卡一个分号卡半小时- 不知道怎么构思类的结构- 你可能会以为“我不适合编程”，其实只是没人帮你拆开问题
35:00 - 45:00	我能给你的帮助：- 每周练习题+逐行批改- 项目规划协助（从写思路图开始）- 面对面+线上答疑机制结尾强调：**“我不是照本宣科的老师，我是一个写了十几年代码的人，我知道你会遇到**

时间	内容
	什么坑，我也知道你该怎么绕开。”

📗 第 3 节：程序员是练出来的

目标：给出清晰学习路径，分享具体工具，激发行动感

时间	内容
0:00 - 10:00	开场互动：“你觉得最理想的学习节奏是什么？”（引导他们说出碎片时间）给出结构化建议：● 每天 15 分钟刷题，● 每周 1 小时项目复盘，● 每 2 周模拟 FRQ
10:00 - 20:00	工具与资源推荐：- Leetcode：刷 Java 小题- VSCode IDE（学生免费）- AP Classroom 官方练习- VSCode 适合练项目
20:00 - 30:00	展示：- 一个简单 ArrayList 小项目（比如学生管理系统）- 你自己做过的项目片段（演示服务器如何通过代码被配置）
30:00 - 40:00	介绍学习支持机制：- 项目任务制+分阶段反馈- 小组互助机制- 你的“项目工坊”：带学生做真实应用
40:00 - 45:00	结尾强调：“你不是为了考试学编程，而是为了能在未来，自己搭建一个想象中的世界。”Q&A+自由讨论时间

接下来，我会为你准备 **AP Computer Science Principles 的三节课内容**，同样是分钟级别安排 + 教学策略。

你需要我现在立刻发 Principles 的版本吗？还是你想先调整这部分？

好的，Yuki，以下是为 **AP Computer Science Principles（CSP）** 设计的 **三节 45 分钟课程内容详案**，同样以**激发兴趣 + 讲清考试内容 + 强调技能价值 + 展现你个人经验**为核心，让学生觉得这是一个“有用、可行、有意思”的课程。

🌐 AP Computer Science Principles 入门课程内容（每节 45 分钟）

📘 第 1 节：编程，不止给理科生

目标：打破误解 + 强调跨学科价值 + 概览课程考试结构

时间	内容
0:00 - 5:00	开场互动：“你觉得编程是理科生专属的吗？”问学生有没有用过 Excel、处理过社交媒体数据结论：其实你已经在“接触计算思维”了
5:00 - 15:00	举例：📊 心理学做问卷 ➔ 用 Python 分析数据 🎨 设计专业做网页作品集 ➔ 用 HTML+CSS 📈 商业专业做数据可视化 ➔ 编程图表强调：**编程是未来所有专业的通用工具**
15:00 - 25:00	简要介绍课程模块：- 计算机运作方式- 网络与互联网- 数据的获取、分析与影响- 算法、抽象、编程构思- Create Performance Task
25:00 - 35:00	考试结构讲解：- 70 道选择题（2 小时）- Create Task（编程+说明文档，30%权重）讲 Create Task 做什么：写一个程序 + 报告它的社会意义和代码细节
35:00 - 45:00	展示：用图形界面做一个“天气提醒器”或“互动日历”Web 项目的例子（哪怕简单，学生会觉得真实）结尾：“这是你能做的，而不是看别人做的。”

📙 第 2 节：你不是背概念，而是在设计世界

目标：让学生理解内容意义，Create Task 不是负担，而是创作机会

时间	内容
0:00 - 5:00	复习上节课内容：课程结构 + Create Task
5:00 - 15:00	讲一个故事：“你能不能帮你爸妈做一个自动记账的小 App？”“你能不能给你的小妹妹做一个识字游戏？”用这些故事带出编程任务：**编程=解决问题**
15:00 - 30:00	拆解课程关键内容：- 抽象 ➔ 用图代替文字，用变量代替数据- 算法 ➔ 解决问题的方案- 网络 ➔ 日常使用背后的技术- 数据 ➔ 如何通过数据做决定穿插问题引导：“如果一个网站不能识别你点击的位置，它会出什么错？”
30:00 - 40:00	Create Task 常见误区：❌ 做得太大 ➔ 做不完 ✅ 做小但有趣 ➔ 拿高分提供你自己做的例子（小游戏、互动表单）
40:00 - 45:00	展示支持：你会提供项目构思模板、代码结构引导、英文文档润色、评分标准解析结尾鼓励：**你不是在做作业，你是在做“你可以发给朋友玩的程序”**

📗 第 3 节：用技能换未来

目标：明确学习路径 + 提供资源支持 + 提升自信

时间	内容
0:00 - 10:00	引导学生制定目标：“你希望用这个课做出一个什么程序？”提供模板：生活辅助工具 / 教学小游戏 / 数据可视网页
10:00 - 20:00	学习计划建议：- 每周主题小练习（10 分钟）- 每月一次项目迭代- 每两周模拟小测试- 每人创建一个“学习档案”（可视化你的成长）
20:00 - 30:00	推荐资源：- Code.org AP CSP 路线图- YouTube: Crash Course CS- W3Schools JS 基础、HTML 入门+ 提供你自己整理的中文资源索引
30:00 - 40:00	展示你能提供的帮助：- 项目构思 1 对 1 指导- 批改英文文档，教你怎么描述功能点- 搭建展示网页的支持（GitHub Pages/Netlify）
40:00 - 45:00	收尾：“不是你要适应编程，而是我们会一起用它解决你关心的问题。”Q&A+选择 Create Task 方向讨论

✅ 下一步建议（如果你需要）

我可以接着帮你：

1. 写一份**开课宣传文案**，吸引学生报名，强调“老师背景 + 未来价值”
2. 帮你整理成**可打印课程讲义/幻灯片稿子**
3. 或者创建**练习题和小项目引导文档**供你后续使用

你现在想要哪一个？或者我可以一起配合做一个教学包。