

// laboratório codex 03 · microeconomia

python + ia + app

>_ Laboratório Codex 03

Simulador de micro em Python, com IA

Curvas de indiferença que se movem, excedentes coloridos, monopólio contra concorrência e uma matriz de jogos editável: a Microeconomia num app feito por você.

// apostila do aluno III · o projeto em etapas

01 Modelos em funções

03 Mercados e excedentes

05 Teoria dos jogos

07 Stackelberg

02 Consumidor e produção

04 O app em abas

06 Relatório em Excel

08 Apresentação



Prof. Caio Haffner

Mestre em Desenvolvimento Econômico · Unicamp

1 Como usar esta apostila

✖ LEIA TUDO, NA ORDEM

Leia esta apostila **desde a primeira palavra** e faça tudo o que ela manda, na ordem. Não pule direto para os exercícios: cada página prepara a seguinte. Se ainda não fez a **Apostila I** (inclusive a página "PARE. Leia antes de tudo"), comece por ela.

Esta é a **Apostila III**, que continua a I (instalação e primeiros passos) e a II (exercícios no terminal). Agora você vai construir **um projeto completo**: um **simulador de microeconomia**: escolha do consumidor com curvas de indiferença, produção, estruturas de mercado com excedentes coloridos e teoria dos jogos, tudo num app interativo.

1. Modelos

Funções testadas

2. Gráficos

Consumidor e produção

3. Mercados

Estruturas e excedentes

4. App

Abas interativas

5. Relatório

Jogos e Excel

★ O PROJETO: SIMULADOR DE MICROECONOMIA

Cada aba é um modelo da Apostila II em gráfico interativo: a restrição orçamentária e as curvas de indiferença, a produção com rendimentos decrescentes, a comparação entre concorrência, Cournot e monopólio com os excedentes sombreados e uma matriz de jogos editável que encontra o equilíbrio de Nash.

Ao final, você deverá conseguir

Habilidade	Como saber que consegui	Ok
Programar modelos	Funções testadas com os números da Apostila II.	☐
Desenhar o consumidor	Restrição orçamentária, indiferença e cesta ótima.	☐
Mostrar a produção	PMe e PMg com rendimentos decrescentes.	☐
Comparar mercados	Concorrência, Cournot e monopólio com excedentes.	☐
Resolver jogos	Matriz editável com equilíbrio de Nash.	☐
Explicar	O app escreve o que cada gráfico mostra.	☐

1

REGRA DE OURO

Você é responsável pelo código que entrega, mesmo quando a IA o escreveu. **Não entregue código que você não consegue explicar.** No projeto, vale mais uma: **a cada etapa, o app precisa continuar funcionando.** Teste antes de seguir.

2 Antes de começar

Você precisa do que montou nas Apostilas I e II: VS Code, Python e Codex funcionando. Se faltar algo, volte à **Apostila I, seções 3 a 7**. A novidade são três **bibliotecas** que dão ao Python gráficos, telas e planilhas.

Instale as bibliotecas (uma vez só)

No terminal do VS Code, digite:

TERMINAL

```
python -m pip install streamlit matplotlib pandas openpyxl
```

Biblioteca	Para que serve
pandas	Trabalhar com tabelas (como uma planilha dentro do Python).
matplotlib	Fazer gráficos: linhas, barras, pizza, dispersão.
streamlit	Transformar o programa num aplicativo que abre no navegador.
openpyxl	Salvar e ler arquivos do Excel (.xlsx).

Para conferir, digite:

TERMINAL

```
python -c "import streamlit, matplotlib, pandas, openpyxl; print('tudo certo')"
```



CHECKPOINT

O terminal mostrou `tudo certo`.



SE DER ERRO

Se aparecer "pip não é reconhecido" ou "python não é reconhecido", troque `python` por `py` (como na Apostila I). Se a instalação demorar, é normal: são bibliotecas grandes. Se travar, copie a mensagem e use o prompt universal de diagnóstico.

Crie a pasta do projeto

Crie a pasta `Laboratorio_Codex_MICR03` em Documentos e abra no VS Code. Ao longo das etapas, ela vai ficar assim:

TERMINAL

```
Laboratorio_Codex_MICR03/  
├─ dados/           ← arquivos de dados (CSV)  
├─ calculos.py      ← as contas (funções)  
├─ graficos.py      ← os gráficos  
├─ app.py           ← a tela (Streamlit)  
└─ relatorio.xlsx   ← gerado pelo app
```

i GUARDE VERSÕES

Ao terminar cada etapa com o app funcionando, copie a pasta inteira com o nome da etapa (por exemplo, `Laboratorio_Codex_MICR03_etapa2`). Se algo quebrar depois, você volta.

3 Seu primeiro app

Antes do projeto, um teste rápido das ferramentas novas. Crie um arquivo `teste_app.py` com o código abaixo. Você pode digitar ou pedir ao Codex.

```
import streamlit as st  
import matplotlib.pyplot as plt  
  
st.title("Meu primeiro app")  
valor = st.number_input("Valor aplicado (R$)", value=1000.0)  
taxa = st.slider("Taxa ao mês (%)", 0.0, 2.0, 1.0)  
meses = st.slider("Meses", 1, 60, 12)  
  
saldos = [valor * (1 + taxa / 100) ** m for m in range(meses + 1)]  
fig, ax = plt.subplots()  
ax.plot(saldos)  
ax.set_xlabel("Mês")  
ax.set_ylabel("R$")  
st.pyplot(fig)  
st.metric("Valor final", f"R$ {saldos[-1]:,.2f}")
```

Para abrir o app, digite no terminal:

TERMINAL

```
python -m streamlit run teste_app.py
```



CHECKPOINT

O navegador abriu um app com o título "Meu primeiro app", dois controles deslizantes e um gráfico. Mexa nos controles: o gráfico muda na hora.

Complete a tela

TERMINAL

Valor aplicado: 1000 · Taxa ao mês: 1,00 · Meses: 12

Valor final: R\$ _____



ARMADILHA: O FORMATO DO NÚMERO

O app mostra o valor no formato americano (vírgula no milhar e ponto nos centavos). Peça ao Codex: *"mostre os valores no formato brasileiro, com ponto no milhar e vírgula nos centavos"*. Você vai usar isso em todo o projeto.



COMO O STREAMLIT FUNCIONA

Cada vez que você mexe num controle, o Streamlit roda o programa inteiro de novo, de cima para baixo. Por isso o gráfico se atualiza sozinho. Para fechar o app, volte ao terminal e aperte `Ctrl` + `C`.



LEMBRETES DAS APOSTILAS ANTERIORES

Peça sempre um **plano** antes do código · teste com contas feitas por você · peça o **diagnóstico antes da correção** · aceite vírgula ou ponto · só aprove o que entender.

4

Etapa 1 · Os modelos em funções

Os modelos da Apostila II reunidos num arquivo e testados.

PROMPT 1

Crie `calculos.py` com funções para: cesta ótima Cobb-Douglas, produção Cobb-Douglas (Q , PM_e , PM_g), mercado em concorrência perfeita, Cournot com n empresas e monopólio (para $P = a - b \cdot Q$ e custo c), excedentes e peso morto, e equilíbrio de Nash numa matriz 2×2 .

Escreva testes com `assert` usando os números da Apostila II.

Teste	Esperado	A função devolveu
Cesta (R 1.000, P_x 10, P_y 20, α 0,4)	(40, 30)	
Monopólio ($P = 100 - 2Q$, $c = 20$)	Q 20, P 60, lucro 800	
Cournot 2 empresas (a 100, c 10)	q 30 cada, P 40	
Nash (dilema)	(Baixo, Baixo)	

5

Etapa 2 · O consumidor e a produção em gráficos

O gráfico mais famoso do consumidor: a restrição orçamentária tocando a curva de indiferença mais alta possível.

PROMPT 2

Crie graficos.py com:

1. grafico_consumidor: a restrição orçamentária, três curvas de indiferença (a da cesta ótima e duas vizinhas) e a cesta ótima marcada;
2. grafico_producao: produção total, produto médio e produto marginal pelo número de trabalhadores, marcando onde o PMg começa a cair.



CHECKPOINT

A curva de indiferença da cesta ótima toca a restrição orçamentária em um só ponto (40, 30).

6

Etapa 3 · Mercados e excedentes em gráfico

O gráfico que resume a Microeconomia: quanto cada estrutura de mercado tira do consumidor.

PROMPT 3

Em `graficos.py`, crie `grafico_estruturas`: a demanda, o custo marginal e, para a estrutura escolhida (concorrência, Cournot com n empresas ou monopólio), o preço e a quantidade, com o excedente do consumidor, o lucro e o peso morto sombreados em cores diferentes.



CHECKPOINT

No monopólio ($P = 100 - 2Q$, $c = 20$), o triângulo do peso morto tem área 400.

7 Etapa 4 • O app em abas

Cada modelo numa aba, com controles que movem os gráficos.

PROMPT 4

Crie app.py com Streamlit e abas:

1. "Consumidor": sliders para renda, preços e α ; o gráfico e a cesta ótima;
2. "Produção": sliders para A , K , α e β ; o gráfico e os rendimentos de escala;
3. "Mercados": a , b , c e a escolha da estrutura (e de n no Cournot); o gráfico e uma tabela comparando as três estruturas.

TERMINAL

```
python -m streamlit run app.py
```

Teste no app	Resultado
Px de 10 para 8	
Cournot com 10 empresas: preço	
$\alpha + \beta = 1,2$: rendimentos	

8 Etapa 5 · Teoria dos jogos

Uma aba com a matriz de payoffs editável. O app encontra os equilíbrios de Nash sozinho.

PROMPT 5

Acrescente a aba "Jogos": uma matriz 2x2 editável (st.data_editor) com os ganhos de A e B,
o destaque das melhores respostas, os equilíbrios de Nash e a estratégia dominante, se houver, e uma frase explicando o resultado. Comece com o dilema dos prisioneiros.

Jogo	Equilíbrio(s) de Nash
Dilema dos prisioneiros (da Apostila II)	
Coordenação: (10,10) (0,0) / (0,0) (10,10)	

Etapa 6 · O relatório em Excel

O app gera um relatório dos experimentos, uma aba por modelo.

PROMPT 6

Acrescente um botão "Baixar relatório" com uma aba por modelo (Consumidor, Produção, Mercados, Jogos), com os parâmetros usados, as tabelas e a frase de explicação.



CHECKPOINT

O Excel tem 4 abas com os parâmetros que estavam na tela.

10 Desafio extra · Stackelberg: quem decide primeiro

E se uma empresa escolhe a quantidade **antes** da outra? No modelo de Stackelberg, a líder ganha vantagem.

PROMPT 7

desafio

Acrescente à aba Mercados o duopólio de Stackelberg (líder e seguidora) para $P = a - Q$ e custo c : quantidades, preço e lucros, comparando com Cournot. Explique a vantagem do primeiro a jogar antes de escrever o código.

11 Use o Codex como revisor do projeto

REVISAR A ORGANIZAÇÃO

Revise o meu projeto como um programador experiente revisaria o de um iniciante. Não faça alterações. Aponte: funções muito longas, código repetido, nomes pouco claros e cálculos misturados com a tela. Separe o que é obrigatório do que é opcional.

REVISAR O CONTEÚDO

Revise os cálculos do meu projeto como um professor de Microeconomia. Não faça alterações. Confira as condições de ótimo, as áreas dos excedentes, os eixos dos gráficos e as hipóteses e liste o que pode estar errado.



GRÁFICO BONITO NÃO É GRÁFICO CERTO

Confira pelo menos um número de cada gráfico com a calculadora ou com a Apostila II. A IA erra eixos, unidades e somas com frequência.

Entrega: apresente o seu projeto

Um projeto só existe quando alguém o vê funcionando. Prepare uma **apresentação de 3 minutos** e um registro para o seu portfólio.

Roteiro de 3 minutos

Tempo	O que mostrar
30 s	O problema que o app resolve e para quem.
1 min 30 s	O app funcionando: mexa nos controles e mostre o peso morto do monopólio diminuindo quando entram mais empresas.
30 s	Um erro que a IA cometeu e como você descobriu.
30 s	O que você faria a seguir, se tivesse mais tempo.

O que entregar

- ☐ A pasta do projeto, com todos os arquivos .py e os dados
- ☐ O relatório em Excel gerado pelo app
- ☐ Três prints do app (telas principais)
- ☐ Um arquivo LEIAME.md explicando o que o app faz e como rodar (peça um rascunho ao Codex e revise)
- ☐ O diário de prompts: pelo menos 5 prompts importantes e o que aconteceu



PARA O SEU PORTFÓLIO

Um post no LinkedIn com um vídeo curto do app funcionando ("Construí em Python, com ajuda de IA, um app de simulação de microeconomia") mostra a recrutadores algo que poucos estudantes têm: **uma ferramenta real feita por você.**

Reflexão individual

1. Qual etapa foi a mais difícil, e como você a resolveu?

2. Em que momento a IA errou, e como você percebeu?

3. O que você aprendeu de Microeconomia construindo o app que não tinha aprendido só com a teoria?

13 Como você será avaliado

Critério	0	1	2	3
O app funciona	Não abre	Abre com erros	Funciona	Funciona e trata entradas inválidas
Conteúdo	Cálculos errados	Alguns certos	Quase todos certos	Todos certos e conferidos
Gráficos	Nenhum	Sem títulos ou eixos	Corretos	Corretos e fáceis de ler
Relatório	Não gera	Gera incompleto	Gera completo	Completo e organizado em abas
Uso do Codex	Não usou	Só copiou	Usou com compreensão	Usou de forma crítica
Apresentação	Não apresentou	Confusa	Clara	Clara, no tempo e com o erro da IA
Registro da auditoria humana (opcional, vale ponto extra)	☐			

Total máximo: 18 pontos, convertidos para a escala da disciplina.

14 Consulta rápida

Comandos

TERMINAL

```
python -m pip install streamlit
matplotlib pandas openpyxl
python -m streamlit run app.py    # abre
o app
Ctrl + C                          #
fecha o app
```

Streamlit

<code>st.title · st.header</code>	Títulos
<code>st.number_input · st.slider</code>	Entradas de número
<code>st.selectbox · st.file_uploader</code>	Escolha e upload
<code>st.metric · st.dataframe</code>	Indicador e tabela
<code>st.pyplot(fig)</code>	Mostrar um gráfico
<code>st.tabs · st.sidebar</code>	Abas e barra lateral
<code>st.download_button</code>	Botão para baixar o Excel

Glossário

Biblioteca

Código pronto que você instala para ganhar recursos (gráficos, telas).

pip

O instalador de bibliotecas do Python.

DataFrame

Tabela do pandas, com linhas e colunas.

App web

Programa que roda no navegador.

Curva de indiferença

Cestas com a mesma utilidade.

Isoquanta

Combinações de insumos com a mesma produção.

Matriz de payoffs

Tabela dos ganhos de cada jogador em cada situação.

1

LEMBRE-SE

A IA é uma ferramenta. Conhecimento, julgamento, validação e responsabilidade continuam sendo de quem programa. Nada nesta apostila é recomendação profissional: é aprendizado.

Todo material produzido com IA precisa de auditoria humana. Sempre. A IA escreve rápido e com segurança, mas pode errar uma conta, inventar uma regra ou usar um dado desatualizado. E ela escreve o erro com a mesma confiança do acerto.

Esta apostila foi escrita com ajuda de IA e revisada pelo professor, conta por conta. Mesmo assim, algum erro pode ter escapado. Agora o auditor é você: encontre, comprove e relate.



REGRA DO LABORATÓRIO: IA SEMPRE PASSA POR AUDITORIA HUMANA

A IA não sabe quando errou: o texto errado sai tão convincente quanto o certo.

Quem assina o resultado é uma pessoa: você no trabalho, o analista no relatório, o professor na apostila.

Auditar não é desconfiar da ferramenta. É a etapa que transforma um rascunho da IA num material confiável.

Vale para tudo: código, contas, textos, gráficos e esta apostila.

O que procurar

Tipo de erro	Exemplo	Como conferir
Conta ou número	Um total que não fecha, uma porcentagem errada	Refaça na calculadora
Regra da disciplina	Fórmula trocada, conceito mal explicado	Compare com o livro ou o material da aula
Referência	"Veja a seção 9", quando o assunto está na 11	Vá até a seção indicada
Prompt	Um prompt que faz o Codex errar sempre do mesmo jeito	Rode o prompt duas vezes
Dado desatualizado	Selic, IPCA, regra de imposto	Confira no site oficial (Banco Central, IBGE, Receita Federal)
Texto	Frase confusa, erro de português	Leia o trecho em voz alta

Como auditar com o Codex

PROMPT DE AUDITORIA

use com cuidado

Vou colar um trecho da minha apostila de [DISCIPLINA].

Atue como revisor. Não reescreva o texto.

1. refaça cada conta e mostre o cálculo;
2. aponte afirmações que podem estar erradas ou desatualizadas;
3. para cada problema, diga como eu posso confirmar sozinho.

Trecho: [COLE AQUI]



A IA TAMBÉM ERRA NA AUDITORIA

O Codex pode apontar erros que não existem. Só relate o que você confirmou com uma conta, uma fonte oficial ou um teste.

Registro da auditoria

#	Seção e página	O que está escrito	O que deveria ser	Como confirmei
1				
2				
3				

Como enviar

- 1 Preencha a tabela acima.
- 2 Envie cada erro pelo formulário: **laboratorio-codex.pages.dev/auditoria**
- 3 Aguarde a confirmação do professor.



O QUE VOCÊ GANHA

Cada erro confirmado vale **+1 ponto extra** na avaliação (até 3 pontos), e o seu nome entra na **lista de revisores** da próxima versão da apostila. Se duas pessoas relatarem o mesmo erro, vale o primeiro envio.