Tutorial V.III - Free Tasks

Programming: Everyday Decision-Making Algorithms

Introduction

This is the final tutorial of the course. You'll have the opportunity to work on a project of your choice, implementing it with the tools learned throughout this course and leveraging AI assistance. Choose one of the projects below based on your interests! In the last session, you will then briefly demo your project in order to pass the course. We only expect prototypes here and it is perfectly fine if things still have bugs or are not fully functional. We just expect you to put some time 2-3 hours outside of class in your project in order to pass. Make it as good as you can!

> **Development Environment**
>
> We recommend developing these projects in standalone `.py` files rather than notebooks. Use an AI-assisted editor like VS Code to help with implementation.

> **AI Assistance**
>
> For each project, start by breaking down the requirements into smaller tasks. Use these as the basis for your AI prompts to generate initial code structure and implementations. Don't try to build everything at once - iterate!

Pygame Games

These projects use the `pygame` library to create interactive graphical games.

1. Jump & Run Platformer

Create a side-scrolling platformer game where the player jumps between platforms, avoids obstacles, and collects items.

Features to implement:

- Player character with left/right movement and jumping
- Gravity and collision detection with platforms
- Obstacles that end the game on contact
- Collectible items that increase score
- Score display and game over screen

Key library: `pygame`

2. Snake Game

Build the classic Snake game where the snake grows longer each time it eats food.

Features to implement:

- Snake that moves in four directions (arrow keys)
- Food that appears at random positions
- Snake grows when eating food
- Game over when hitting walls or itself
- Score tracking based on snake length

Key library: `pygame`

3. Flappy Bird Clone

Create a game where a bird flies through gaps in pipes by tapping/clicking to stay airborne.

Features to implement:

- Bird that falls due to gravity
- Click/spacebar to make the bird "flap" upward
- Pipes that scroll from right to left with gaps
- Collision detection with pipes and ground
- High score tracking

Key library: `pygame`

4. Space Invaders

Build a classic arcade shooter where you defend against descending alien invaders.

Features to implement:

- Player ship that moves left/right at the bottom
- Shooting projectiles upward
- Rows of aliens that move and descend
- Aliens shooting back at the player
- Multiple waves with increasing difficulty

Key library: `pygame`

5. Tic-Tac-Toe with AI

Build a Tic-Tac-Toe game where you play against an unbeatable AI opponent.

Features to implement:

- 3x3 game board displayed in terminal
- Player makes moves by entering positions (1-9)
- AI opponent using minimax algorithm
- Win/lose/draw detection
- Option to play again

Key library: Built-in Python only (no external libraries needed)

6. Receipt Scanner (OCR)

Build a tool that extracts text from receipt images and organizes the data.

Features to implement:

- Load an image file (photo of a receipt)
- Use OCR to extract text from the image
- Parse extracted text for key info (total, date, store name)
- Display results and export to CSV
- Handle multiple receipts

Key libraries: `pytesseract`, `Pillow`, `pandas`

Note: You'll need to install Tesseract OCR on your system separately.

7. Interactive Stock Dashboard

Build a web dashboard for exploring stock market data.

Features to implement:

- Input field to enter stock ticker symbols
- Fetch historical price data using yfinance
- Interactive price chart with date range selection
- Basic statistics (high, low, average, % change)
- Compare multiple stocks on one chart

Key libraries: `streamlit`, `yfinance`, `plotly`

Run with: `uv run streamlit run your_app.py`

8. Grade Calculator Dashboard

Build a dashboard to track your grades and calculate your average grade.

Features to implement:

- Input courses with names and credit hours
- Add assignments with grades and weights per course
- Calculate current grade for each course
- Visualize grade distribution with charts
- "What-if" calculator: what grade do I need on the final?

Key libraries: `streamlit`, `pandas`, `plotly`

Run with: `uv run streamlit run your_app.py`

9. Your Own Idea!

Come up with an idea of your own!

Getting Started Tips

1. Pick one project that interests you most
2. Break it down into small, manageable pieces
3. Start simple - get the basic version working first

4. Use AI to help with specific parts
5. Iterate - add features one at a time

> ⚠ Installing Libraries
>
> Remember to install required libraries using:
>
> ```
> uv add pygame  # or any other library
> ```

Congratulations on completing the tutorials! We hope you enjoyed the course so far and learned a lot!