

> _

RDPQuest - A prompt-like RPG

by Leonardo Manzato

GitHub: leomzto | edX: Leo_Manzato

Rio Paranaíba, Brazil - MG

July, 10th of 2025



Project Overview

\$_ RDPQuest is a text-based RPG playable through the terminal.








\$_ The project aims to provide an engaging adventure experience with character creation, combat, exploration, and progression, all in C language.

\$_ Main features include character creation system, battles against enemies and bosses, dungeon exploration with loot collection, class choices, and level progression.

\$_ Developed entirely in C, leveraging console input/output for interaction.

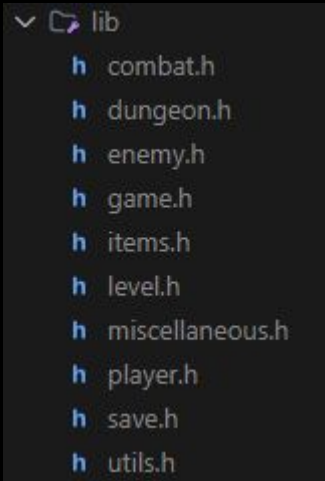
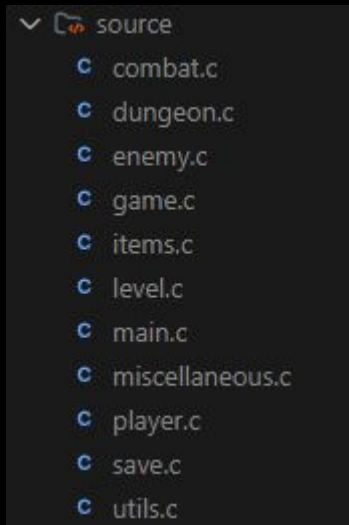
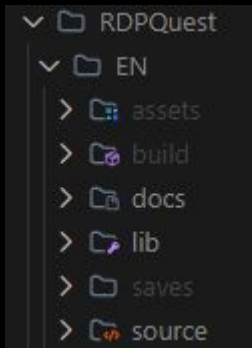


Main Features

- \$_  Character creation with class selection
- \$_  Turn-based battle system
- \$_  Dungeon exploration with random encounters
- \$_  Leveling system with XP gain
- \$_  Enemy variety
- \$_  Save system
- \$_  Loot drops and inventory system



Code Architecture



- RDPQuest (main directory)
- assets (icon)
- build (compiled files and .exe)
- docs (documentation)
- lib (with the headers)
- saves (game save)
- source (with the .c files)



Gameplay Demo

```
$_ make RDPQuest
```


```
$_ ./RDPQuest
```

```
Starting Game
```


```
Progress: [#####] 100%
```




Challenges Faced

\$_  Code organization in C:


Learning how to structure a multi-file project using .h and .c files

\$_  Battle system logic:


Managing turns, HP, enemy behavior, and edge cases (like defeat and healing)

\$_  Randomness and replayability:

Implementing random events and enemy spawns without breaking flow

\$_  Memory management:

Avoiding segmentation faults and properly handling strings and pointers

\$_  User input handling:

Ensuring clean interaction and avoiding infinite loops or crashes



What I Learned

\$_ C programming in practice

Applied knowledge of pointers, structs, functions, and memory allocation

\$_ Modular code structure

Split logic into multiple .c and .h files for better organization

\$_ Debugging and testing

Used printf debugging, handled segmentation faults, and tested edge cases

\$_ Game logic design

Designed systems for character progression, combat mechanics, and dungeon flow

\$_ Iterative development

Improved the game step by step, learning from bugs and user experience

\$_ Communication through code

Wrote cleaner, more readable code with meaningful naming and comments



Future Improvements

\$_ 🧠 Enemy AI

Make enemies behave differently based on class or difficulty

\$_ 🗺️ Multiple dungeons and areas

Expand the world with new challenges, events, and environments

\$_ 🎨 Better user interface

Add colored text and cleaner formatting for a more immersive terminal experience



>_ 🎓 "This was CS50!"

Conclusion && Thanks

\$_ 📍 RDPQuest was a great challenge that helped me grow as a developer.

\$_ 💡 I applied C programming to create a full RPG game, with structured code, logic design, and user interaction.

\$_ 🧠 I learned a lot about modular code, memory management, and building complex systems from scratch.

\$_ 📁 Feel free to explore the code:

Repository: github.com/leomzto/RDPQuest

GitHub: github.com/leomzto/

LinkedIn: linkedin.com/in/leomzto

🙏 Special thanks to:

\$ _CS50 and the Harvard team

\$ _My family, for supporting me during the course

\$ _And everyone who tested or gave feedback!