

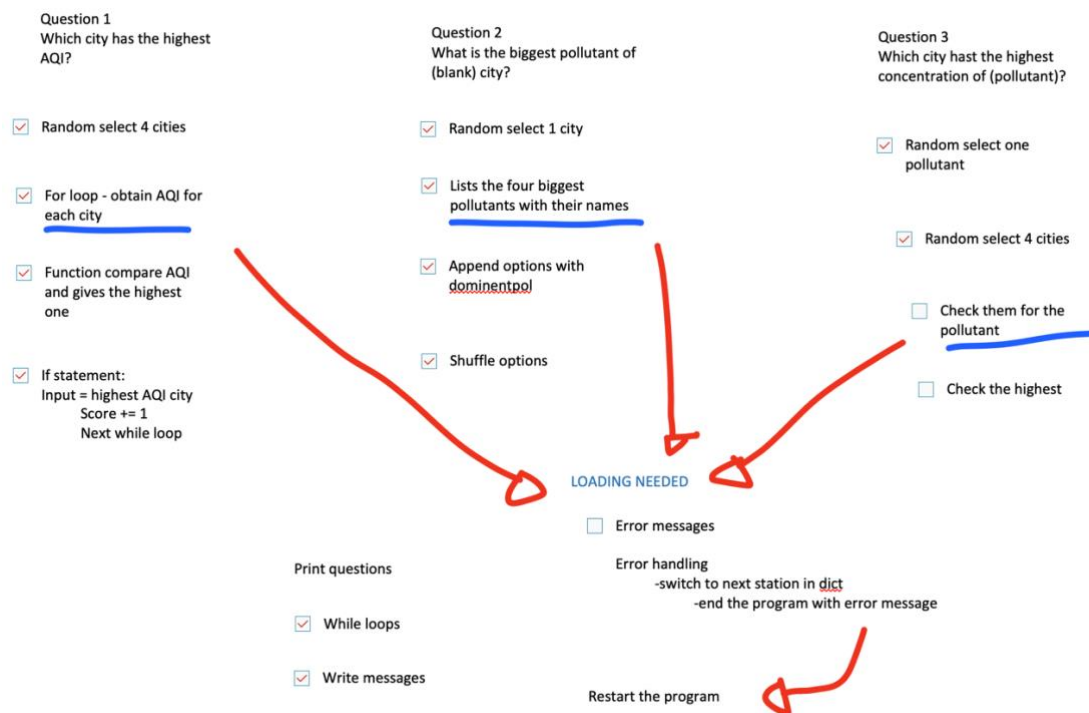
CART-351-2252-A  
André Neder de Almeida  
Student ID: 40208953

## Project #1 – Creative Coding in the Terminal

### “Who Wants to Breath a Million Air”

After not having written a line of code in nearly a year, learning Python from the ground up has been quite a challenge, but an enjoyable one. Despite not being totally friendly with Python, I really enjoyed making something of my own using no other than an air quality API.

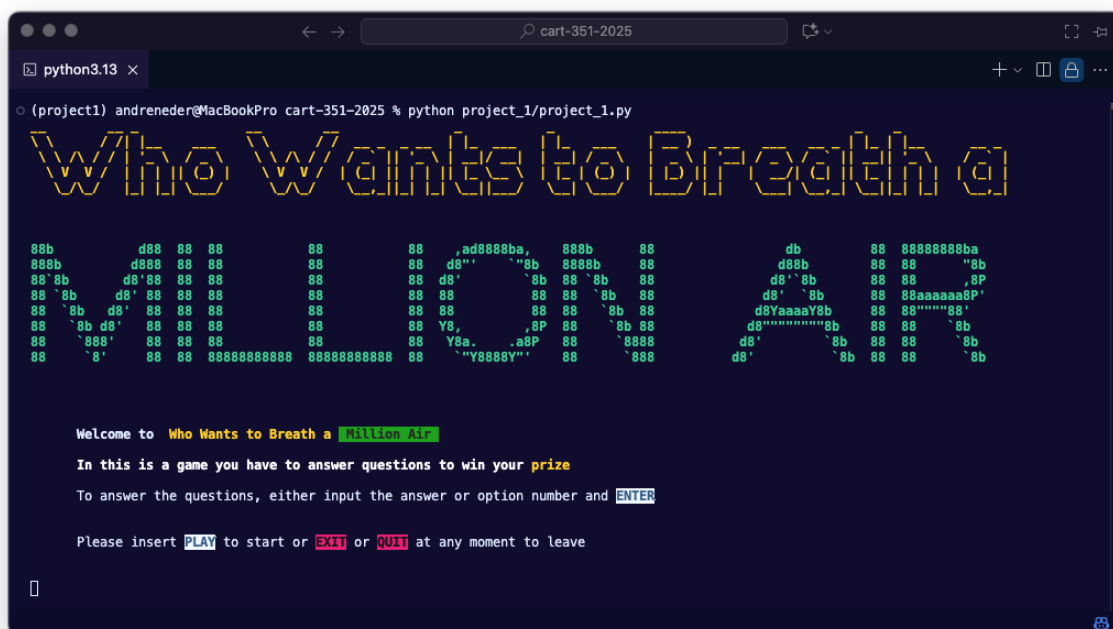
Once I came up with the name *Who Wants to Breath a Million Air*, based on the popular game show, I still wanted to keep it simple and re-use some of the code for iterations from exercise one that used the same API. So, I came up with the idea of having three questions that would quiz the user on cities and AQI levels at random.



*My planning attempt*

The very first thing I wanted to explore were the artistic libraries for fonts and colors from Python, I particularly used *art* and *rich* a lot in the program which made the results quite interesting

and visually attractive. Suffice to say that such libraries worked really well and made the titles stand out.



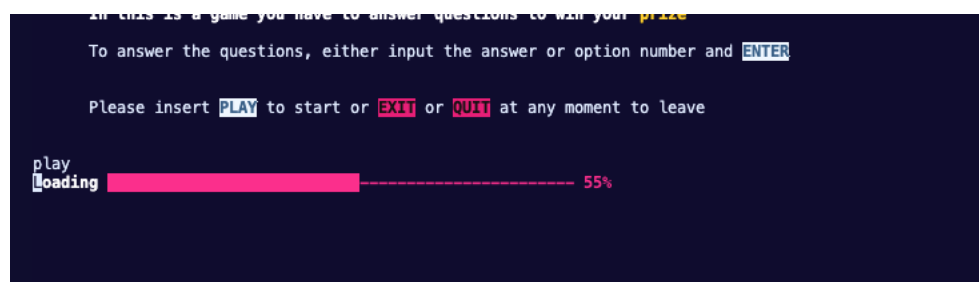
```
python3.13 x
cart-351-2025
(project1) andreneder@MacBookPro cart-351-2025 % python project_1/project_1.py

Who Wants to Breathe a Million Air

Welcome to Who Wants to Breathe a Million Air
In this is a game you have to answer questions to win your prize
To answer the questions, either input the answer or option number and ENTER
Please insert PLAY to start or EXIT or QUIT at any moment to leave
```

One library I could not get around with was *keyboard* and *pynput*. I wanted to have the option to do simple keyboard inputs such as spacebar to start, but the incompatibility cross platform that I found in the documentation and forums just made it so hard to make a seamless experience, requiring often that the program had the necessary permissions or to run in `sudo`. I abandoned that idea.

Coming from JavaScript and being used to dividing my code in states or classes, I have not explored that yet in Python which caused a challenge on organizing my code. I resourced to while loops when necessary but left some of the code that fetched API info out so it could be loaded. I added a loading bar that I adapted from the tutorial by [NeuralNine](#). The loading screen worked really well, despite having to look a lot in the documentation to adapt it and taking some valuable time.



```
In this is a game you have to answer questions to win your prize
To answer the questions, either input the answer or option number and ENTER

Please insert PLAY to start or EXIT or QUIT at any moment to leave

play
Loading [pink bar] 55%
```



*Loading bar before and after completion*

Even though the loading bar gave some valuable feedback, I would often get fetching errors from the API which required me to add error messages and game unsatisfactory endings. Most I mostly solved it by doing work arounds to try and fetch results form the next valuable station but often, that would fail too, which led me to use if statements to make the sure that the code wouldn't bleed errors. Even after submission I hope to keep working on this to solve such issues, should time allow.



I also think I could have done a better job creating functions, for example, all the times that I fetched cities, could have been reduced to a function with return values, but being new to the language made re-writing it more natural.

In general, I really enjoyed making this first program and, despite of all its' flaws, it still runs and works. My main motivation to making projects like this is really to briefly entertain a classmate or someone who is going through the same learning process as me. This encourages me to take on challenges like this and further my studies in Python.

