**Creating a Daily Affirmation Generator in Python: An Inspirational Coding Journey**

**Introduction**

In today’s busy world, starting each day with positivity can make a huge difference in our outlook. That’s why I created a Daily Affirmation Generator using Python! This simple program delivers uplifting affirmations to inspire you and set a positive tone for the day. In this blog post, I’ll walk you through the journey of building the generator, the challenges I encountered, and the unique features I incorporated to make it a memorable experience for users. Even if you’re new to programming, I’ll explain each step in easy-to-understand terms.

**Why Build a Daily Affirmation Generator?**

Affirmations are powerful for encouraging self-belief, gratitude, and resilience. As a beginner programmer, I wanted to create something that combined coding with a positive impact. The Daily Affirmation Generator allows users to receive random affirmations or select from specific categories, like "Abundance," "Growth," and "Healing." This project not only helped me learn essential coding skills but also gave me a tool to spread positivity.

**How the Program Works**

This program is designed to deliver a unique affirmation each day, offering two main options:

1. **Random Affirmation:** This option randomly selects an affirmation, giving you a delightful surprise each day.
2. **Choose a Category:** If you have a particular focus in mind, you can select a category (like "Healing" or "Self-Love") to receive an affirmation aligned with that theme.

To ensure users don’t see the same affirmation repeatedly, I implemented a special feature that “remembers” which affirmations have already been shown, even if you close and reopen the program. This ensures each affirmation is unique until all have been displayed, at which point the list resets.

**Behind the Scenes: Key Components**

**Organizing Affirmations by Category**

The affirmations are organized into categories, which makes it easy for the program to pull affirmations based on a theme. This way, users can choose an affirmation that matches their mood or goal for the day.

**Ensuring Unique Daily Affirmations with a JSON File**

One challenge I encountered was figuring out how to make the program “remember” which affirmations were shown, even after closing it. This led me to explore **JSON files**.

**What’s a JSON File?** JSON (JavaScript Object Notation) is a simple format for storing and organizing data in a structured way. Although it originated from JavaScript, JSON can be used in various programming languages, including Python. Think of a JSON file as a mini storage unit for data—ideal for saving information that you want to access later.

In this project, the JSON file acts as a “memory” for the program. Each time you view an affirmation, it’s recorded in the JSON file. When you open the program again, it checks the JSON file to avoid repeating affirmations until all options have been displayed. Without this setup, the program would “forget” past affirmations each time it restarted, leading to repeats. With the JSON file, the experience feels smoother and more engaging.

For those interested in learning more, resources like [JSON.org](https://www.json.org) or the Python JSON documentation can be helpful.

**Creating a Welcoming Design with Unicode Characters**

I wanted the program’s welcome screen to feel inviting, so I used decorative symbols like hearts and stars to create a friendly design. These symbols are known as **Unicode characters**.

**What Are Unicode Characters?** Unicode is a universal system for encoding symbols from all languages, plus additional symbols, emojis, and icons. Each Unicode character has a unique code, making it display consistently across platforms. Here are a few examples:

* 💖 (Sparkling Heart) - Code: U+1F496
* ★ (Star) - Code: U+2605
* 🌈 (Rainbow) - Code: U+1F308

Using these characters, I designed a welcoming border of hearts around the program’s introductory message. Aligning these symbols was a bit challenging since they can display differently depending on the system, but after some trial and error, I created a layout that reflects the uplifting spirit of the program.

A white background with black hearts

Description automatically generated

**Running the Program: The Importance of if \_\_name\_\_ == "\_\_main\_\_":**

You might notice a line of code at the end of our program: if \_\_name\_\_ == "\_\_main\_\_":. Although it looks complex, this line is crucial for our program to run as a standalone project.

This line essentially tells Python, "Run this part of the code if this script is executed directly." Without it, the program wouldn’t start automatically. After researching this feature, I realized it’s especially useful for standalone projects like mine, ensuring the program initiates as soon as it’s opened.

**Key Features of the Daily Affirmation Generator**

* **Simple User Interaction:** Text-based prompts make it easy for users to navigate and choose their desired affirmation.
* **Unique Daily Affirmations:** Using the JSON file, the program prevents repeats, ensuring each day’s affirmation is fresh.
* **Personalized Categories:** Users can select affirmations based on themes, adding a personal touch.
* **User-Friendly Code:** The project introduces core programming concepts like file handling, random selection, and user input, making it accessible for beginners.

**Exploring the Code on GitHub**

If you’d like to see the full code or try running it yourself, you can find it on my GitHub repository. This project is perfect for beginners, combining positive impact with practical coding skills.

**Conclusion**

Building the Daily Affirmation Generator taught me coding essentials, from random selection and user input to file handling with JSON. It also showed me that coding can have a meaningful, positive impact.

For any beginner interested in coding, this project is a great starting point. I hope this generator brings some positivity to your day, just as it has for me. Happy coding and stay inspired!