```
import json

from typing import List




class PromptGenerator:

    """A class for generating custom prompt strings."""


    def __init__(self) -> None:

        """Initialize the PromptGenerator object."""

        self.constraints: List[str] = []

        self.commands: List[str] = []

        self.resources: List[str] = []

        self.performance_evaluation: List[str] = []

        self.response_format = {

            "thoughts": {

                "text": "thought",

                "reasoning": "reasoning",

                "plan": (

                    "- short bulleted\n- list that conveys\n-"

                    " long-term plan"

                ),

                "criticism": "constructive self-criticism",

                "speak": "thoughts summary to say to user",

            },

            "command": {

                "name": "command name",
            "args": {"arg name": "value"},
        },
    }

def add_constraint(self, constraint: str) -> None:
    """
    Add a constraint to the constraints list.

    Args:
        constraint (str): The constraint to be added.
    """
    self.constraints.append(constraint)

def add_command(self, command: str) -> None:
    """
    Add a command to the commands list.

    Args:
        command (str): The command to be added.
    """
    self.commands.append(command)

def add_resource(self, resource: str) -> None:
    """
    Add a resource to the resources list.
```

```
        Args:
            resource (str): The resource to be added.
        """
        self.resources.append(resource)

    def add_performance_evaluation(self, evaluation: str) -> None:
        """
        Add a performance evaluation item to the performance_evaluation list.

        Args:
            evaluation (str): The evaluation item to be added.
        """
        self.performance_evaluation.append(evaluation)

    def generate_prompt_string(self) -> str:
        """Generate a prompt string.

        Returns:
            str: The generated prompt string.
        """
        formatted_response_format = json.dumps(
            self.response_format, indent=4
        )
        prompt_string = (

f"Constraints:\n{''.join(self.constraints)}\n\nCommands:\n{''.join(self.commands)}\n\nResources:\n{''.j
oin(self.resources)}\n\nPerformance"

            f" Evaluation:\n{''.join(self.performance_evaluation)}\n\nYou"

            " should only respond in JSON format as described below"

            " \nResponse Format:"

            f" \n{formatted_response_format} \nEnsure the response"

            " can be parsed by Python json.loads"

        )


        return prompt_string
```