

## Copilot

You can add a check in the setter to ensure that only the keys “T” and “D” are allowed. Here’s how you can modify the setter:

```
class MyClass:
    def __init__(self, params: dict = {"T": 1.0, "D": 0.5}) -> None:
        self._params: dict = params

    @property
    def params(self) -> dict:
        """ Get or set the parameters """
        return self._params

    @params.setter
    def params(self, params) -> None:
        if isinstance(params, dict):
            if set(params.keys()).issubset({"T", "D"}):
                self._params = params
            else:
                raise ValueError("Invalid keys in params. Only 'T' and 'D' are allowed.")
        elif isinstance(params, tuple) and len(params) == 2:
            key, value = params
            if key in {"T", "D"}:
                self._params[key] = value
            else:
                raise ValueError("Invalid key in params. Only 'T' and 'D' are allowed.")
        else:
            raise ValueError("params must be a dict or a tuple of length 2")
```

Now, if you try to update the params with a key other than “T” or “D”, it will raise a ValueError. For example:

```
obj = MyClass()
try:
    obj.params = ("X", 2.0) # This will raise a ValueError
except ValueError as e:
    print(e)
```

This will print: “Invalid key in params. Only ‘T’ and ‘D’ are allowed.” This ensures that only the keys “T” and “D” can be used in the params dictionary.