```python
import csv
import os
import sys


class CitrusTracker:

    def __init__(self):
        self.log_file: str = "citrus_info.csv"
        self.citruses: list = self.read_csv()
        print(self.citruses)

    def read_csv(self) -> list:
        """
        Lees de content van het csv bestand om de data
        te gebruiken.

        :return file_content: list, een lijst met dictionaries met de citrus info
        """
        file_content = []

        with open(os.path.join(sys.path[0], self.log_file), "r") as file:
            file_content = list(csv.DictReader(file))

        return file_content

    def write_csv(self) -> None:
        """
        Schrijf de nieuwe data naar het csv bestand
        """
        headers = self.citruses[0].keys()

        with open(os.path.join(sys.path[0], self.log_file), "w") as file:
            writer = csv.DictWriter(file, headers)
            writer.writeheader()
            writer.writerows(self.citruses)
```

```
        with open(os.path.join(sys.path[0], self.log_file), "w") as file:
            writer = csv.DictWriter(file, headers)
            writer.writeheader()
            writer.writerows(self.citruses)

    def add_new_citrus(self, new_information: str) -> None:
        """
        Sla nieuwe infromatie op in het csv bestand.

        :param self, huidige object
        :param new_information: str, string met de nieuwe informatie, commas seperated
        """
        headers = list(self.citruses[0].keys())

        new_information = {k: v for k, v in zip(headers, new_information.split(","))}

        self.citruses.append(new_information)

        self.write_csv()

    def pp_info(self, citrus_variant: str) -> None:
        """
        Pretty print de informatie van een specifieke citrus.

        :param self, huidige object
        :param citrus_variant: str, welk type citrus je wilt printen
        """
        fruits = list(filter(lambda fruit: fruit['variant'] == citrus_variant, self.citruses))

        for fruit in fruits:
            location, acres, family, variant, date_planted = fruit.values()
            print(f"{variant} of the {family} family is planted in {location} on a {acres} acre field,")
            print(f"and was planted on {date_planted}\n")


def main() -> None:
    """
    Main programma met alle functionaliteit bij elkaar.
    """
```

```python
def main() -> None:
    """
    Main programma met alle functionaliteit bij elkaar.
    """
    citrus_tracker = CitrusTracker()
    print(citrus_tracker.citruses)

    print("""Welkom bij de citrus tracker.
Hier kan je de oogst van de citrus bomen bijhouden
[i] info van een citrus fruit
[a] nieuwe citrus fruit toevoegen
[q] eindig het programma""")

    quit_program = False

    while quit_program is False:
        command = input("> ").lower()
        if command in ('q', 'quit'):
            quit_program = True
        elif command in ('i', 'info'):
            user_citrus = input("Give a citrus type:\n> ")
            citrus_tracker.pp_info(user_citrus)
        elif command in ('a', 'add'):
            print("Please input new information for a new tree:")
            print("Enter 5 fields (location, acres, family type, variant and date planted)")
            print("Seperate by comma's")
            new_information = input("> ")

            if "," not in new_information:
                print("Not seperated by commas")
            else:
                citrus_tracker.add_new_citrus(new_information)


if __name__ == "__main__":
    main()
```