Assignment Report: Animal Shelter

Name: Alain Niganze
Email: niganzealain@gmail.com
Telephone: 0783943932
Date: 16/06/2024

Introduction

I have developed an animal shelter management program as part of my assignment. The program allows users to manage a list of animals in the shelter through a menu-driven interface, providing options to add, remove, find, and sort animals. The project also includes unit tests to ensure the correctness of the implemented functionalities.

Project Structure

The project directory is organized as follows:

```
AnimalShelter/
├── bin/
├── product/
│   ├── administration.c
│   ├── administration.h
│   ├── animal.h
│   ├── animal_shelter.c
├── test/
│   ├── administration_test.c
├── Makefile
```

Implementation

The program consists of several components:

1. `animal.h`: Defines the `ANIMAL` struct and species enumeration.
2. `animal_shelter.c`: Contains the main program logic, including menu-driven interaction.
3. `administration.h`: Declares the functions for managing the animal array.
4. `administration.c`: Implements the functions for adding, removing, finding, and sorting animals.
5. `administration_test.c`: Contains unit tests for the functions in `administration.c`.

Functionality

- Add Animal: Users can add a new animal by entering its name, species, and age.
- Remove Animal: Users can remove animals by specifying their name.
- Find Animal by Name: Users can search for an animal by name.
- Sort Animals by Age: Users can sort the animal list by age.
- Sort Animals by Name: Users can sort the animal list by name.

Compilation

The project is compiled using the `Makefile`. The `Makefile` ensures the program and tests are compiled and placed in the `bin` directory.

To compile the project, run:
bash
make

To compile and run the tests, use:
bash
make adminTest
./bin/administrationTest

To clean the project, use:
bash
make clean

To run the main program, use:
bash
./bin/animal_shelter

Example Usage

1. Running the Program:
   bash
   ./bin/animal_shelter

2. Adding an Animal:

   Enter animal name: Whiskers
   Enter animal species (0: Cat, 1: Dog, 2: GuineaPig, 3: Parrot): 0

Enter animal age: 3

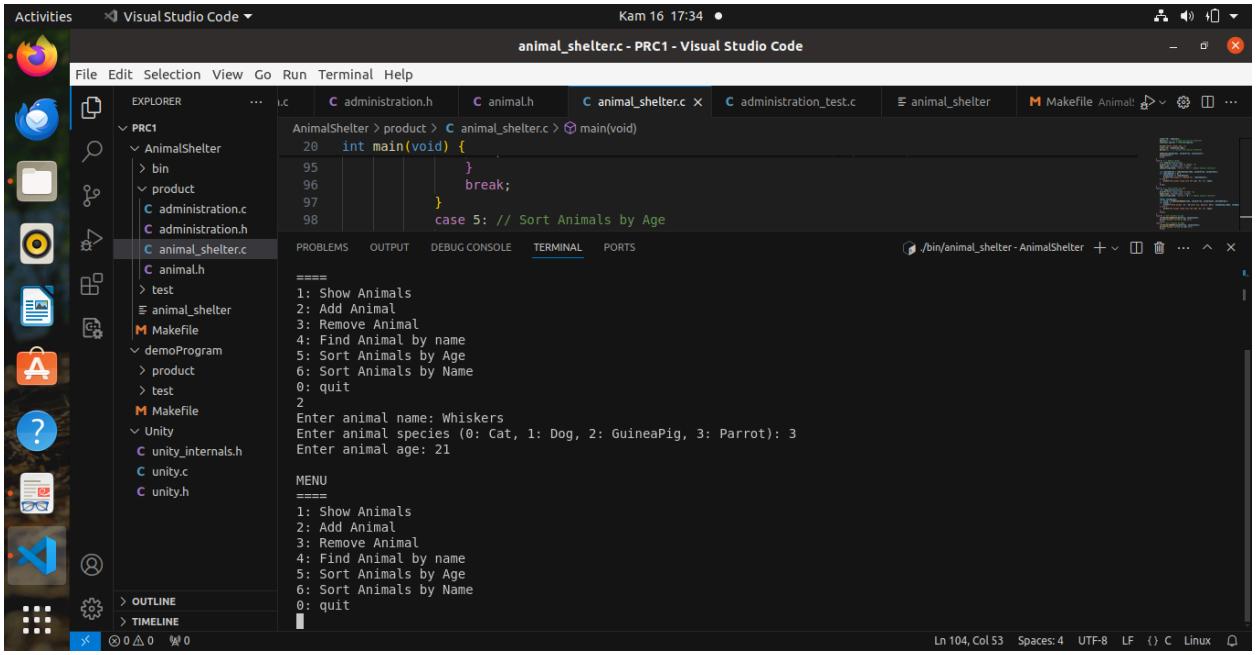


Figure1: show addition of animals in animal_shelter.

3. Removing an Animal:

Enter animal name to remove: Whiskers

4. Finding an Animal by Name:

Enter animal name to find: Whiskers

5. Sorting Animals by Age:

Animals sorted by Age.

6. Sorting Animals by Name:

Animals sorted by Name.

Screenshots

Here are some screenshots showing the program in action:

1. Program Menu:

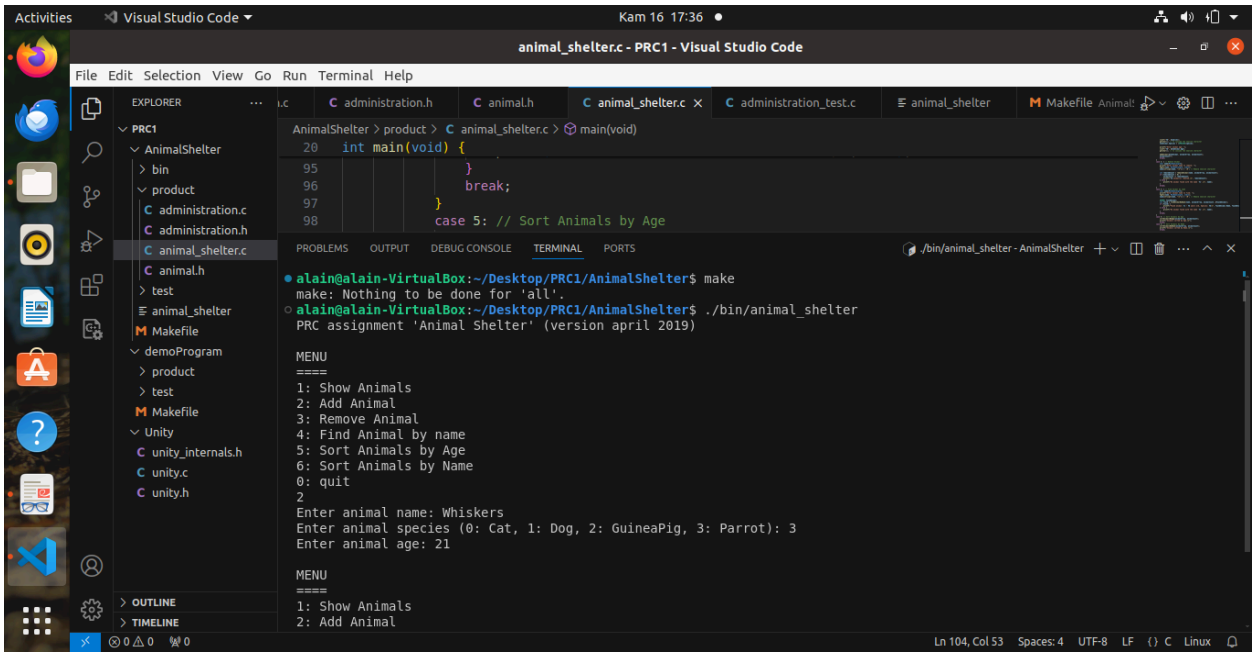


Figure2: shows![Program Menu](screenshots/menu.png).

2. Adding an Animal:

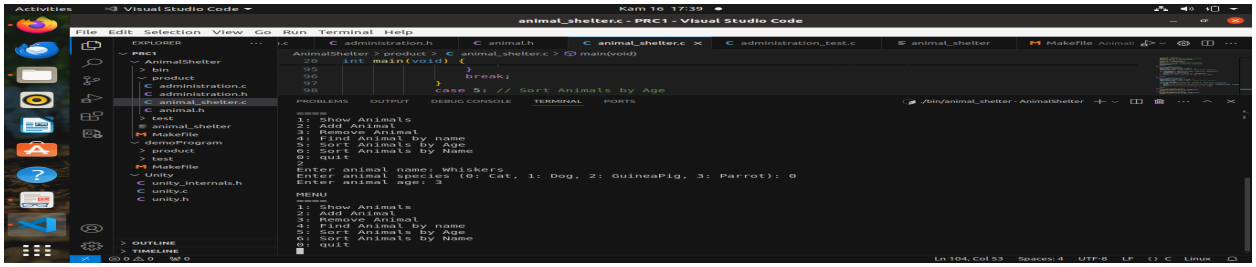


Figure3: ![Adding an Animal](screenshots/add_animal.png).

3. Removing an Animal:

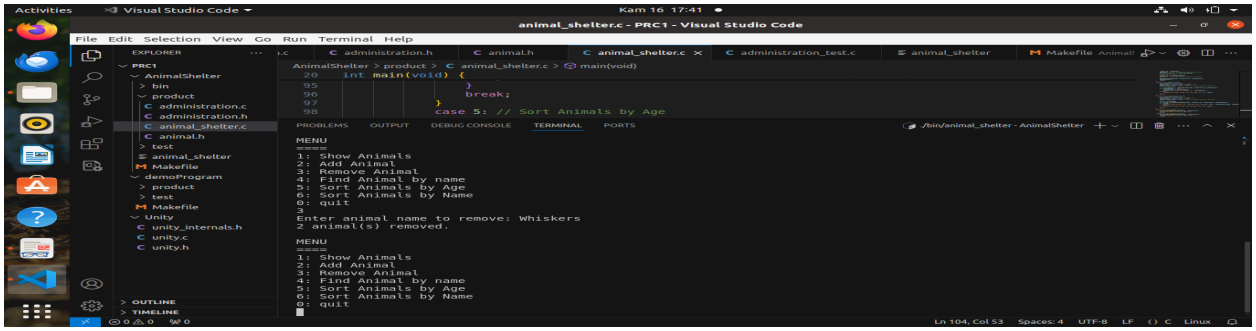


Figure4: ![Removing an Animal](screenshots/remove_animal.png).

4. Finding an Animal by Name:

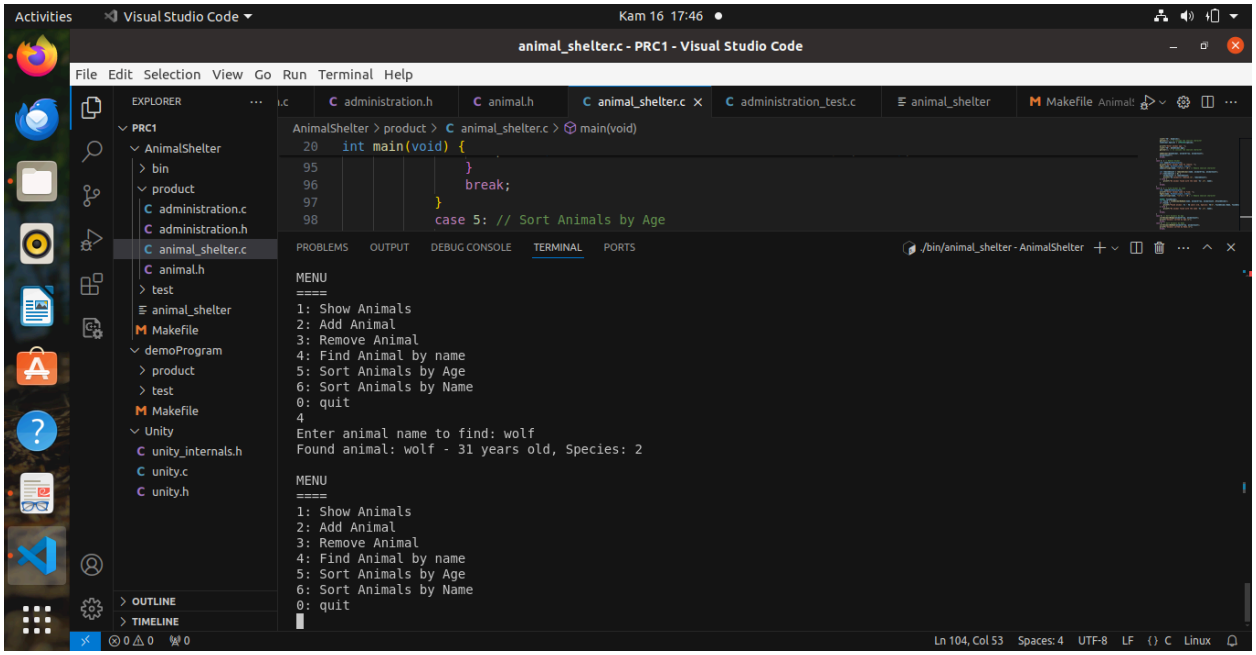
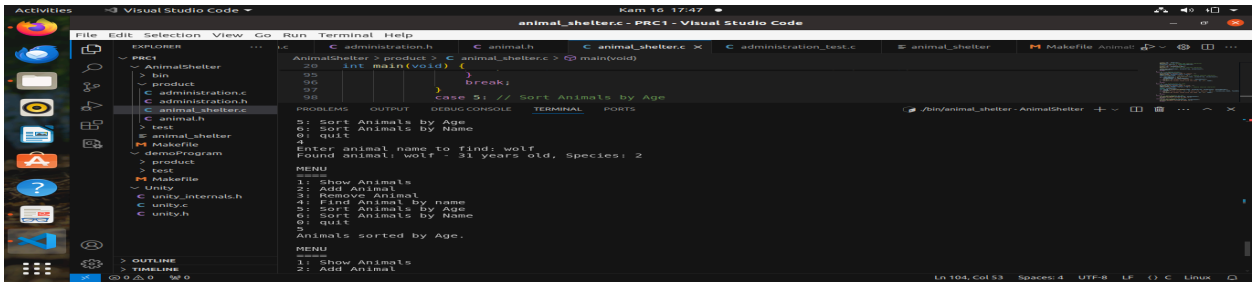


Figure5: ![Finding an Animal](screenshots/find_animal.png)

5. Sorting Animals by Age:



After sorted you see that..

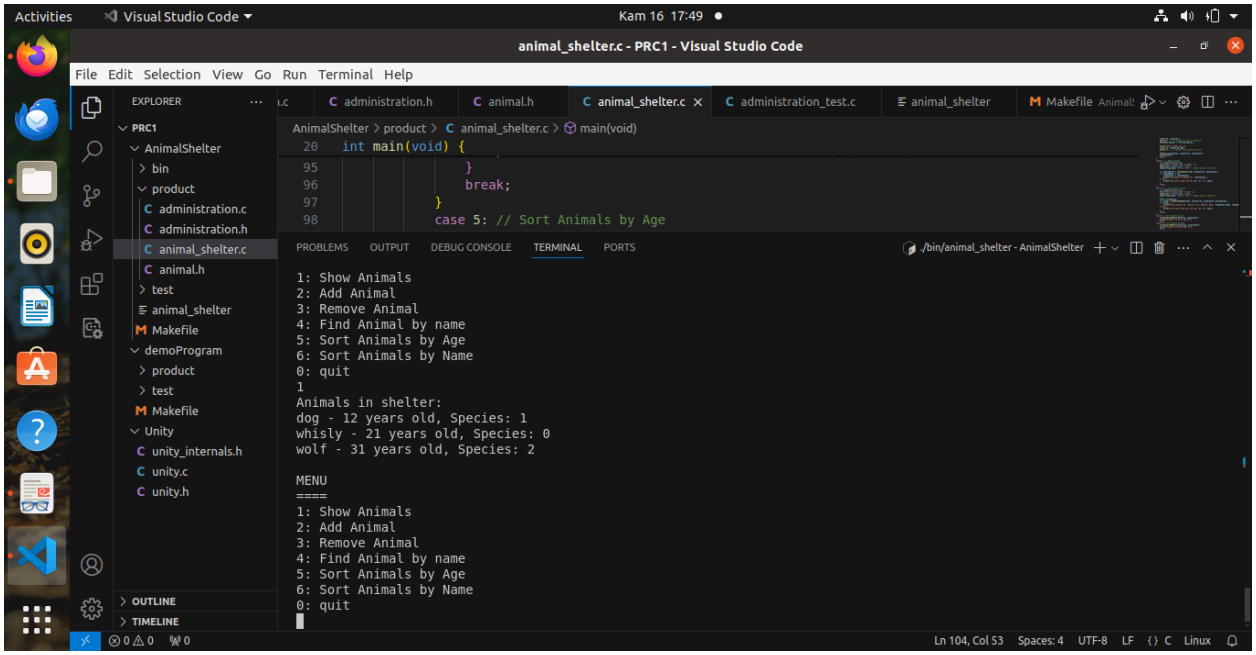


Figure6-2: shows animals [Sorting by Age](screenshots/sort_age.png).

6. Sorting Animals by Name:

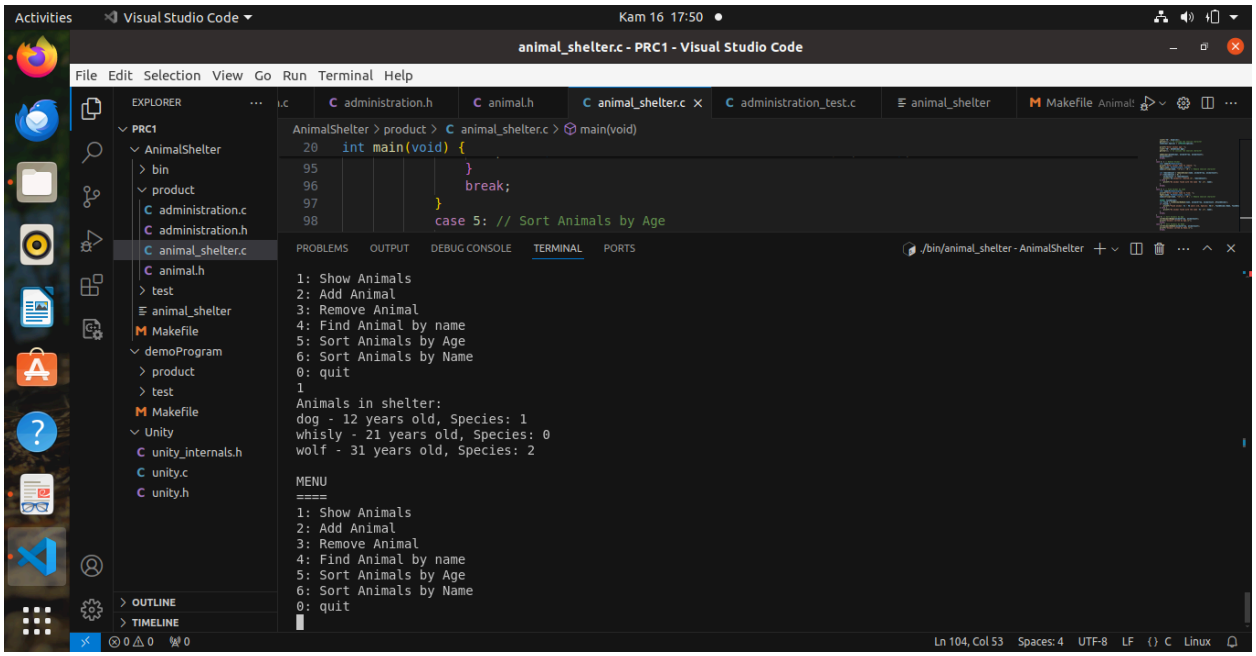


Figure7: shows sorted animals [Sorting by Name](screenshots/sort_name.png).

Conclusion

The animal shelter management program successfully implements the required functionality. Users can interact with the program through a menu to manage the list of animals. The project is well-structured and includes comprehensive unit tests to ensure reliability.

This assignment provided a good exercise in C programming, including working with structs, arrays, and implementing basic sorting algorithms. The Makefile setup ensures easy compilation and management of the project.

Since the code I've built is very long, it's difficult to capture a screenshot that would be helpful. Instead, you can find it directly on GitHub.

End of report.