Algorithms and Programming

Virtual Pet Breeding and Simulation Game

- This project is a virtual pet breeding and simulation game using the C programming language to develop a new application. In the project, struct, pointer, filing, string, array and functions such as C basic features of the language will be used and data structures, filing operations, string operations and topics such as functions will be applied. In addition, user interface design, time management and game will offer the opportunity to develop development skills. Ready-made functions will not be used in the project.

The tasks required from the student in the project are as follows:

1. Pet Structure (Struct):

- A pet structure (struct) must be defined for the project. In the pet structure the animal information such as type, name, energy level, happiness level, mood.
- The struct structure should be used to keep track of the pet's properties and status.

2. Filing Procedures:

- The pet's data should be stored in a file. Pet data must be saved and uploaded to the file using filing operations.
- Appropriate functions should be created for file reading and writing operations.

3. User interface:

- A user interface should be designed that allows the user to control and feed their pet.
- The user should be offered options such as feeding the pet, playing games, cleaning.
- Tasks should be presented in which the user must meet the pet's needs and increase his happiness.

4. Energy and Happiness Control:

- The pet's energy and happiness levels should be dynamically monitored.
- Energy and happiness levels should be updated as the user feeds the pet, plays games and does other activities.
- The pet's status should be notified to the user when their energy and happiness levels drop below a certain level.

Functions that need to be created:

- `add_pet`: The user creates a new pet. It will not allow the creation of a new animal with the same name by checking the existing animals in the `pet_data.txt` file.
- `pet_update`: The user can change the name of the pet he wants.
- `delete_pet`: The user can delete any pet they want.
- `feed_pet`: You can increase the energy level by giving food to the pet. Each feeding adds +15 points to energy level and +10 points to happiness level.
- `give_water`: By giving the pet water, you quench its thirst. Each quenching adds +5 to energy level and +5 to happiness.
- `play_games`: Time for the pet to play with its toy You can separate. Each game-playing action has energy level -15, happiness level. Adds + 15 points.
- `clean_pet`: Health of the pet by meeting the cleaning needs you can protect. Each cleansing action adds -10 points to energy level and -5 points to happiness level.
- `love_pet`: You can increase its happiness by showing love and attention to the pet. Each act of love adds 0 to the energy level and +5 to the happiness level.
- `put_to_sleep`: You can provide a comfortable place and environment for the pet to sleep. Each sleep action adds +5 to energy level and 0 to happiness level.
- `reward_pet`: Reward your pet for good behavior You can provide motivation. Each reward action adds 0 points to energy level and +10 points to happiness level.

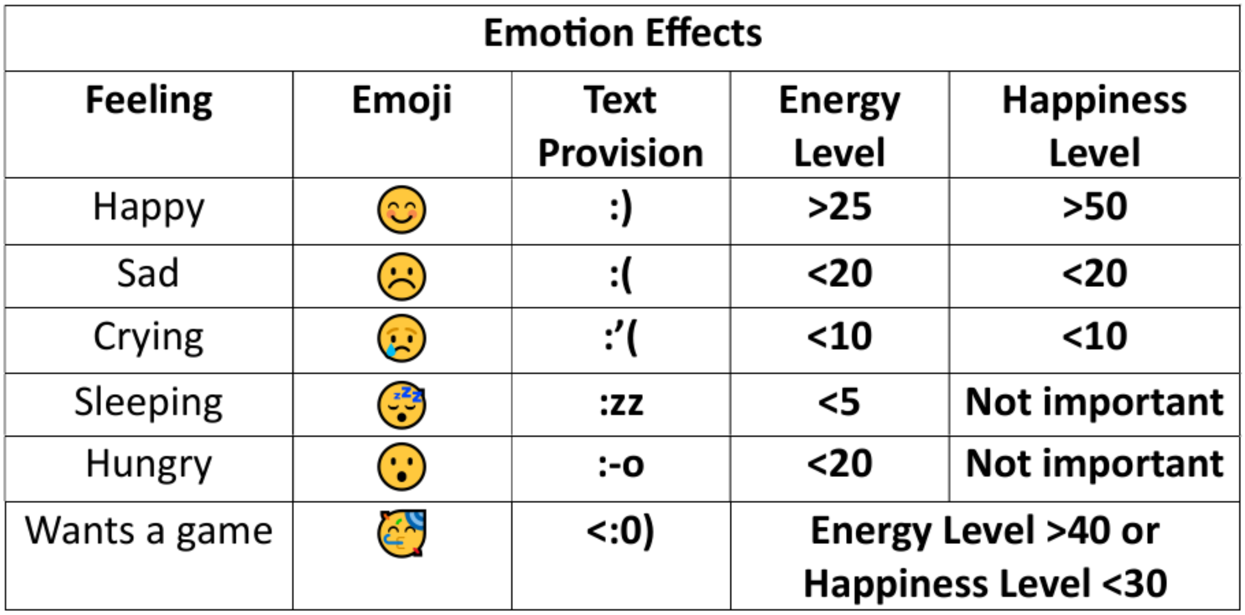
- Type, name, energy level and happiness level of the pet initially `pet_data.txt` will be specified in the file. At the start of the game, each pet will have an energy level of 60 and a happiness level of 60.

Example `pet_data.txt` file content:
`Cat,Mia,60,60`

- When the pet's energy level drops below +5, no action will be allowed and the user will put to sleep will be requested. Other functions will not work unless the user puts the pet to sleep. When the pet's happiness level drops below +5, the user should only be allowed actions that will increase happiness.
- Output of each transaction `pet_output.txt` file will be saved.
- At least 3 pets must be created in the game. The user can play the game by adding as many animals as he wants in the game. The output file for each pet should be created separately.

5. Visual Effects:

- Different emotional states of the pet (hungry, unhappy, wanting to play) should be expressed visually. Mood states as indicated in the table below `emotion.txt` should be kept on file.



Emotion Effects				
Feeling	**Emoji**	**Text Provision**	**Energy Level**	**Happiness Level**
Happy	😊	**:)**	**>25**	**>50**
Sad	☹️	**:(**	**<20**	**<20**
Crying	😢	**:'(**	**<10**	**<10**
Sleeping	😴	**:zz**	**<5**	**Not important**
Hungry	😮	**:-o**	**<20**	**Not important**
Wants a game	🥳	**<:0)**	**Energy Level >40 or Happiness Level <30**	

Sample `emotion.txt` content:

```
1,happy,:),EL>25,HL>50
2,sad,:(,EL<20,HL<20
....
....
```

After each selection, according to the emotional state of the pet, the above-mentioned emotional effects will be selected from the `emotion.txt` file and printed both to the output file (`pet_output.txt`) and to the screen.