Implementation requirements:

- 3 user-defined classes (one class can be a driver class)
- 4+ data members in two of the classes
- 1+ array of objects from a class that you created.
- 6+ if-else statements
- 4+ loops (while loops, for loops, do-while, in total)
- 2+ nested loops
- File IO (both reading from a file and writing to a file)

Project feature requirements:

- The project must have interactive components (ask the player for inputs, create menus for choices, and so on). It's a game!
- Game stats should be displayed at each turn. It's more exciting and meaningful! Also, these stats help debug the code.
- Your project must include:
    - At least 5 menu options (other than Quit/Exist)
    - At least two of these options must have a second layer of menu options
    - At least 2 menu options(primary or secondary layer of the menu) should include a random component, at least one each from the following:
        - The value of the variable is selected at random from a certain range of values (i.e. select a value at random between 1 and 6)
        - A probability value determines one of the outcomes (i.e. there is a 60% change a certain event will occur)

Inspiration -

- Bandersnatch: different endings depending on the choice
- Runescape: different skills that you can develop throughout the game while doing quests
- Golden compass

Requirements

- Currency for buying stuff in the game
- Characters and character traits/strengths, affect how the game is played
- Game stats (tracking score)
- Animals that you pick: ferret vs parrot vs monkey, wolf vs snow leopard vs owl
- (tier system)
- Skills: obedience, speaking, navigation, fighting, agility (flying, swimming etc.), rescue/loyalty
- Challenges: basic training and obedience, pet gets lost and finds you once you call them, owner gets hurt and pet helps them, fight (final challenge)

Menu Ideas

- 2 layer menu: challenges, then levels. If the user has completed a level, display it as "level 1 (completed)".

Header files

Class Animal

Private:

Static const int SIZE= 8; (this is however many skills we have)

String skills [SIZE];

String animalName;

String username;

String animalType;

Public:

Void setUsername (string);

Void setAnimal(string);

Void setAnimalName (string);

Void increaseSkillLevel(int, int, bool); //takes position in matrix and bool if they finished the challenge

String getAnimalName();

String getUsername(); /

Void getStats(); //get animals current stats

Void writeAnimalStats(string); // we write animal stats to a file once a user finishes with that animal, this will read that file when the user wants to see its stats

Bool sufficientSkillLevel(int, int);

Class Challenges

Private:

Static const int COLUMNS;

bool challenges[5][COLUMNS]; //matrix to set completed levels

Public:

Void setCompletedLevel(bool); //set completed challenge at level

Bool checkLevels(); //checks if all levels for a challenge are finished (so it won't be offered in a menu anymore

Bool finishedChallenges(); //checks if all the challenges are finished (so we only give the user the option to quit or move on to the next tier)

Class Play

Private:

Int userChoice;

Public:

Void playChallenge1(int);

// call menu for the levels

// cout << enter a level << endl;

// cin >> level

// user that cin value for the switch statement

// change challenges matrix based on if they finished the level or not

Void playChallenge2( );

Void playChallenge3( );

```
        Void playChallenge4( );
        Void playChallenge5( );
        Void playChallenge6( );

Class Menus
    Private:
        Int userInput;
    Public:
        String animalMenuTier1( ); //3 animal options for tier 1, displays stats
            // cout << “enter a number” << endl

        String animalMenuTier2( ); //3 animal options for tier 2, displays stats
        Int challengeMenu1( ); //displays levels (including what is completed and what
        isn’t), returns the level user chooses
        Int challengeMenu2( );
            // cout << enter a level << endl;
            // cin >> level
        Int challengeMenu3( );
        Int challengeMenu4( );
        Int challengeMenu5( );
        Int challengeMenu6( );
        Int endOfTierMenu( );
```

Outline: Animal Tamer

Program starts by setting the scene and describing the world the adventure game is set in, then you are taken to the first challenge: choose your animal. You choose which animal you want to tame, give it a name, and provide your username. A text file will be read in, with all the animals and their levels for different skills, and you will pick between 3 animals for each tier. For the first tier or first animal you tame, the options are a ferret, parrot, or monkey, the second tier is a wolf, snow leopard, or owl, and finally when you have trained your animals from the two tiers to a sufficient level, you will unlock the polar bear: once you have tamed it you have won the game.

The user is taken through a series of challenges in different locations, their animal increases their level in a given skill if the user makes the right decisions in the challenge, else they will loop back to the beginning of the challenge and must try again. When you have gone through level one of each of the challenges, the program will display the animal’s stats and give you a menu for different challenges you can choose to do to further improve the animal’s skills to the necessary level to move on. Each animal will need to go through a certain number of

levels to be trained in that skill, Once you have finished enough levels to tame the animal, you will have the option to either quit the game, pick an animal from the next tier, or continue moving through the levels of the challenges. After finishing the second tier you unlock the polar bear, where instead of having challenges related to improving your skill, you look for the golden compass with the help of the polar bear to eventually win the game. Along the way you can collect gold to buy certain items, and random challenges will pop up that can either give bonus points to your animal's skill, or will cause your animal to run away which will cause you to go back to the beginning.

BEGIN GAME:
"Welcome to Animal Tamer, the game designed to take you through the steps of pet ownership and training, with three levels of taming, challenges and trials.

File:
int readAnimals(string file1);
int readScores(string file2);

int skills[7][6];

Animals.txt
ferret
2,2,3,1,4,3
parrot
4,5,3,3,4,2
monkey
2,1,2,1,4,5
wolf
1,1,3,2,1,3
snow leopard
3,2,1,3,1,1
owl
4,1,2,3,1,1
polar bear
2,1,2,4,2,2

obedience, speaking, navigation, fighting, agility, rescue/loyalty

Parrot: 2,3,1,4,5,4 = 19 obedience challenge 3 levels, 2 levels speaking

Animals[7];

Animal array, skill level array

Pikachu

Attack	defense	combat
5	3	1