Project Design

The Zoo Tycoon game should have six required classes; the zoo class, the animal class, the tiger class, the penguin class, the turtle class, and the NewAnimal class. The Animal class will be a pure virtual function that will be the base class for the tiger, penguin, turtle, and NewAnimal classes. Within the zoo class, there are five required member variables. Of the five, four of them will be overridden by the derived class, as they have different values for the each of those classes.

The Zoo class will house the main variables and functions for the project. It will have a variable to hold the amount of money the zoo currently has, dynamic arrays for each of the animal types, and a function that will run the game. The game should ask the user if they want to continue to the next day at the end of every loop. In that loop, the age of each animal will be increased, they will be fed, which means that we reduce our cash amount, and we determine how much we made from the payoff, which increases our cash.

A random event then occurs, which is determined by a separate function. Each of these random events has an equal chance to occur. The three random events are one of the species getting sick, one of the species having a baby, and an attendance boom.

The game repeats until the user ends the game.

Pseudocode

```
Animal class
        Private:
                Int age;
                Double cost;
                Int numBabies;
                Double baseFoodCost;
                Double payoff;
        public:
                Animal();
                ~Animal();
                // Getters
                Int getAge();
                Double getCost();
                Int getNumBabies();
                Double getBaseFoodCost();
                Double getPayoff();
                // Setters
                Void setAge(int ageIn);
                Void setCost(double costIn);
                Void setNumBabies(int babiesIn);
                Void setBaseFoodCost(double foodCostIn);
                Void setPayoff(double payoffIn);
                // increase age
                Void increaseAge();
```

These are repeated for the four derived classes.

```
Zoo class
        Private:
                Double cash;
                Int currentDay;
                // Tiger array
                Tiger * tigerArray;
                // Penguin array
                Penguin * penguin Array;
                // Turtles array
                Turtle * turtleArray;
                // NewAnimal array
                NewAnimal * newAnimalArray;
        Public:
                Tiger* getTigers; // returns the tigerArray
                Penguin* getPenguins; // returns the penguinArray
                Turtle* getTurtles; // returns the turtleArrya
                NewAnimal* getNewAnimals; // returns the newAnimalArray


                runGame();      // function to run the game
```

```
ageAnimals(); // to age the animals at the beginning of the day
feedAnimals(); // To feed the animals and decrement cash
performRandomEvent(); // Does a random event
welcomeMessage(); // Intro message
endDay(); // Message sent out at the end of the day
addTiger(tigerIn); // Adds a tiger to the array
addPenguin(penguinIn);// Adds a penguin to the array
addTurtle(turtleIn);// Adds a turtle to the array
addNewAnimal(newAnimalIn); // Adds a newAnimal to the array
sickAnimal(); // Kills one of the animals
attendanceBoom(); // Increases the cash
babyBorn(); // Adds a new animal to a random species
calculateProfit(); // Calculates how much the zoo made
checkBalance(); // Checks the banks balance
```

Test Tables

Test Case	Expected Output	Actual Output
Step through to day 30	The animals will breed quite heavily, and the zoo will lose a lot of money	All the animals died. The zoo made money off it.
Add a new animal	The end of day board should correctly update and the question will no longer be asked	The end of day board updated and the question was no longer asked.
Set the tigers to 0 and see how quickly the zoo loses money	With no tigers, the zoo will lose money	The zoo lost money
Valgrind Project2	There should be no memory leaks	There were no memory leaks, but there was an issue with trying to access index out of bound. That was fixed.

Reflection

This project was more difficult than expected. While the initial structure of the classes and their inheritance was no problem, I ran into issues when trying to determine how the animals would be handled within the Zoo class. While I initially had success, I ran into some issues with the increased size of the dynamic array.

I am not certain that the final iteration I programmed is the best one, but it’s the one that worked at the end. During my attempts to get it to work correctly, I tried multiple versions of the function. However, I believe the issue that was plaguing me was caused by the function that called the function that increased the size of the array. In the first function, I increased numAnimals by however many babies were related to the animal in question, but then I used that variable when trying to determine what information needed to be kept for the new array. This caused multiple variable out of bounds errors to occur, and I was too fixated on the double array function to find the source of the problem.

Eventually, it was found and the function ran without issue. If I were to work of this program again, however, I would spend more time focused on specific animals. As it stands now, my program does not differentiate between the animals within the species. As per the specifications of the project, there is not a requirement to randomize which animal within the species dies, simply that one of the species is selected at random. However, this makes for slightly unsatisfying gameplay. As it stands, I am not displeased with how the project turned out.