** Program name: Zoo Tycoon

** Author: Nicole Beecher

** Date: 10/22/2017

Program Design

Create classes:

1. animal
2. tiger – inheritance from animal
3. penguin – inheritance from animal
4. turtle – inheritance from animal
5. zoo

1. animal:
    a. Parent class
    b. Initializes variables to default values
2. Tiger
    a. Inheritances from animal class
    b. Refers to default values but returns different values for tigers
        i. Ex (cost of food) default is 10 – tiger is default times 5
3. Penguin
    a. Inheritances from animal class
    b. Refers to default values but returns different values for penguins
4. Turtle
    a. Inheritances from animal class
    b. Refers to default values but returns different values for turtles
5. Zoo
    a. Initializes start of game (asks user to buy animals)
    b. Runs game with infinite while loop
    c. Holds functions for random events
    d. Arrays for each animal

Test Table

Test Case	Input Values	Function	Expected Outcomes	Observed Outcomes
User must buy at least 1 of each animal	Input > tiger, penguin, turtle	Zoo – startGame()	Loop back to input asking for a number greater than 0	Loop back to input asking for a number greater than 0
If user buys 1 tiger then bank account subtracts $10,000	Input = 1	Zoo – startGame()	Bank account = bank account - 10000	Bank account = bank account - 10000
If user buys 1 penguin then bank account subtracts $1,000	Input = 1	Zoo – startGame()	Bank account = bank account - 1000	Bank account = bank account - 1000
If user buys 1 turtle then bank account subtracts $100	Input = 1	Zoo – startGame()	Bank account = bank account - 100	Bank account = bank account - 100
If random event lands on a babyEvent for tiger then 1 baby is added to the array	No input	Zoo – randomEvent(), babyEvent()	1 tiger at age 1 day is added to tiger array	1 tiger at age 1 day is added to tiger array
If random event lands on a babyEvent for penguin then 5 babies are added to the array	No input	Zoo – randomEvent(), babyEvent()	5 penguins at age 1 day are added to penguin array	5 penguins at age 1 day are added to penguin array
If random event lands on a babyEvent for turtle then 10 babies are added to the array	No input	Zoo – randomEvent(), babyEvent()	10 turtles at age 1 day are added to turtle array	10 turtles at age 1 day are added to turtle array
If random event lands on a sickness for tiger then 1 random tiger is removed from the array	No input	Zoo – randomEvent(), sickness()	1 tiger is selected at random in array and set to 0 – then loops through array looking for 0 and moves to end of array then subtracts array by 1	1 tiger is selected at random in array and set to 0 – then loops through array looking for 0 and moves to end of array then subtracts array by 1
If random event lands on a sickness for penguin then 1 random penguin is removed from the array	No input	Zoo – randomEvent(), sickness()	1 penguin is selected at random in array and set to 0 – then loops through array looking for 0 and moves to end of array then subtracts array by 1	1 penguin is selected at random in array and set to 0 – then loops through array looking for 0 and moves to end of array then subtracts array by 1
If random event lands on a sickness for turtle then 1 random turtle is removed from the array	No input	Zoo – randomEvent(), sickness()	1 turtle is selected at random in array and set to 0 – then loops through array looking for 0 and moves to end of	1 turtle is selected at random in array and set to 0 – then loops through array looking for 0 and moves to end of

			array then subtracts array by 1	array then subtracts array by 1
If random event lands on a zAttend for tiger then a random number between 250 and 500 is added to the bank account for each tiger	No input	Zoo – randomEvent(), zAttend()	A random number between 250 and 500 is set then multiplied by each tiger and added the total to the bank account	A random number between 250 and 500 is set then multiplied by each tiger and added the total to the bank account

Reflection

This project was heavily focused on inheritance. I learned how to use variables from a parent class and manipulate the return in separate child classes.

The hardest part with this project was the sheer size of it. I don't think I've written that much code ever. It was difficult to keep track of where I was and stay organized.

I also learned to stay simple. I overthink every function / code / outcome and think I need to do something spectacular I haven't learned yet. I need to stay focused on the basics and try to find the simplest way to implement code.