

Introduction:

This document is in two sections. The first section is just how to run it and the main testing and complying with screen shots. The second section is the answers to the questions provided for this test.

Part 1:

How to Run:

In order to run this, I used Visual Studio with the C# compiler, opening the documents through Visual Studio should work. I believe Visual Studio Code should work as well so long as there is a compiler set up in the VS Code. Otherwise an online text editor should work as well.

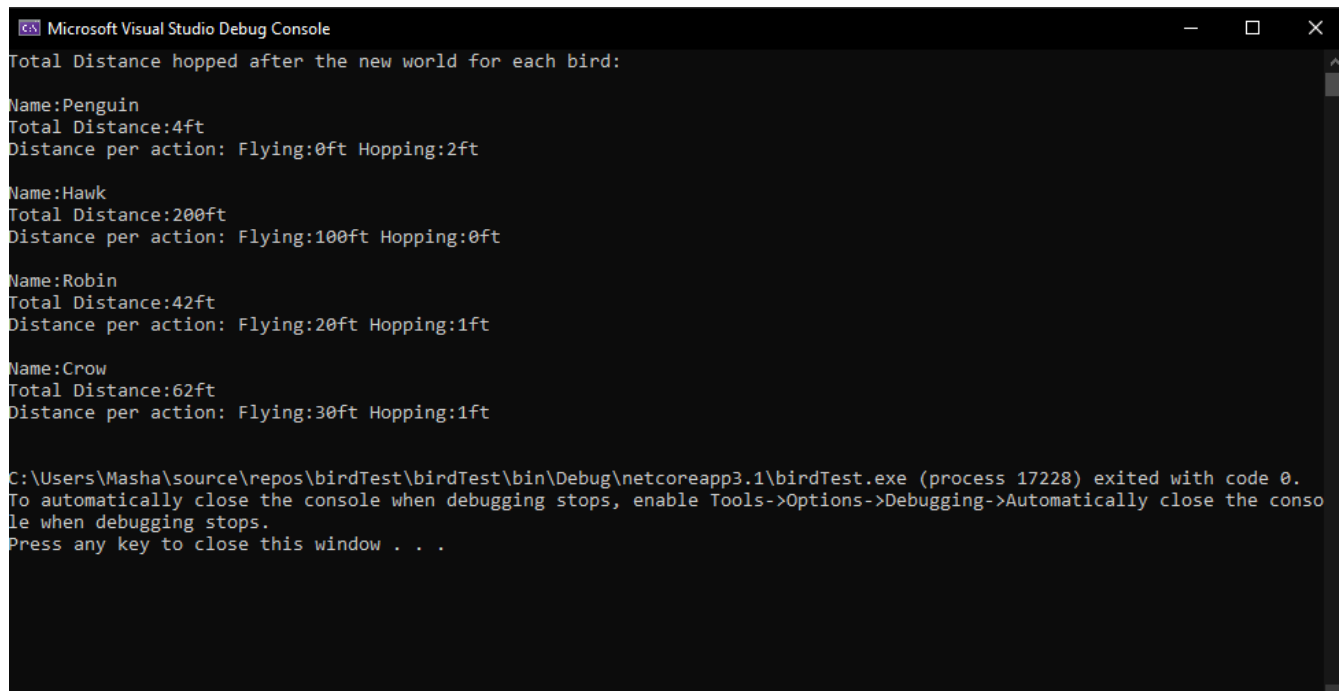
Testing:

In this section I'll be showcasing all the testing. The testing will be done in multiple ways below:

- A perfect world where there are no restrictions in temperature or wind conditions.
- Testing Penguin, Hawk, Robin, and Crow with wind changes.
- Testing Penguin, Hawk, Robin, and Crow with temperature changes.
- Testing with both.

Test 1: No temperature or wind conditions for all birds

```
World one = new World(0, 0); //default with no changes
World two = new World(0, 0); //new world with changes
```



```
Microsoft Visual Studio Debug Console
Total Distance hopped after the new world for each bird:
Name:Penguin
Total Distance:4ft
Distance per action: Flying:0ft Hopping:2ft
Name:Hawk
Total Distance:200ft
Distance per action: Flying:100ft Hopping:0ft
Name:Robin
Total Distance:42ft
Distance per action: Flying:20ft Hopping:1ft
Name:Crow
Total Distance:62ft
Distance per action: Flying:30ft Hopping:1ft
C:\Users\Masha\source\repos\birdTest\birdTest\bin\Debug\netcoreapp3.1\birdTest.exe (process 17228) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Results:

Penguin hops both worlds, but does not fly in either.

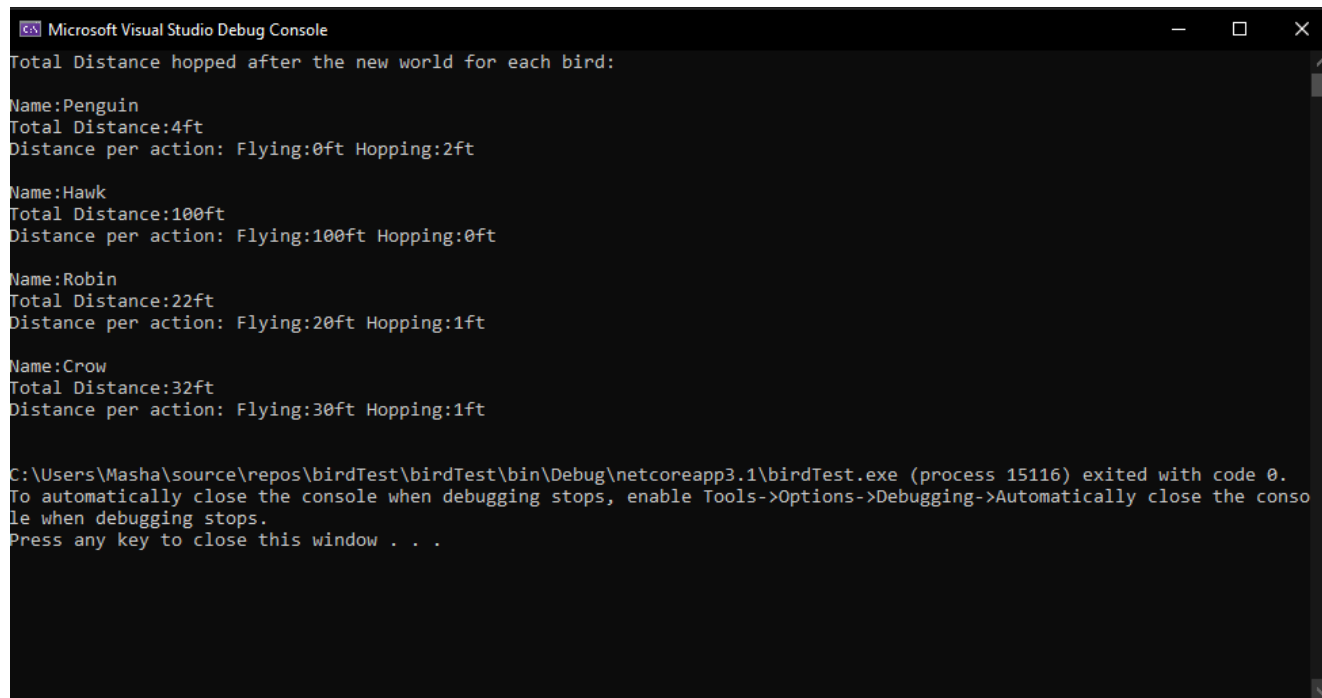
Hawk flies both in both worlds, but does not hop in either.

Robin hops and flies in both worlds.

Crow hops and flies in both worlds.

Test 2: Test for Hawk and the wind speed is 41

```
World one = new World(0, 0); //default with no changes
World two = new World(41, 0); //new world with changes
```



```
Microsoft Visual Studio Debug Console
Total Distance hopped after the new world for each bird:
Name:Penguin
Total Distance:4ft
Distance per action: Flying:0ft Hopping:2ft

Name:Hawk
Total Distance:100ft
Distance per action: Flying:100ft Hopping:0ft

Name:Robin
Total Distance:22ft
Distance per action: Flying:20ft Hopping:1ft

Name:Crow
Total Distance:32ft
Distance per action: Flying:30ft Hopping:1ft

C:\Users\Masha\source\repos\birdTest\birdTest\bin\Debug\netcoreapp3.1\birdTest.exe (process 15116) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Results:

Penguin hops both worlds, but does not fly in either.

Hawk flies in world 1 but not in world 2, but does not hop in either.

Robin does not fly in world 2, but hops and flies in world 1.

Crow does not fly in world 2, but hops and flies in world 1.

Test 3: Test for Robin and the wind speed is 21

```
World one = new World(0, 0); //default with no changes
World two = new World(21, 0); //new world with changes
```

```
Microsoft Visual Studio Debug Console

Total Distance hopped after the new world for each bird:

Name:Penguin
Total Distance:4ft
Distance per action: Flying:0ft Hopping:2ft

Name:Hawk
Total Distance:200ft
Distance per action: Flying:100ft Hopping:0ft

Name:Robin
Total Distance:22ft
Distance per action: Flying:20ft Hopping:1ft

Name:Crow
Total Distance:62ft
Distance per action: Flying:30ft Hopping:1ft

C:\Users\Masha\source\repos\birdTest\birdTest\bin\Debug\netcoreapp3.1\birdTest.exe (process 6844) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Results:

Penguin hops both worlds, but does not fly in either.

Hawk flies both in both worlds, but does not hop in either.

Robin does not fly in world 2, but hops and flies in world 1.

Crow hops and flies in both worlds.

Test 4: Test for Crow and the wind speed is 26

```
World one = new World(0, 0); //default with no changes
World two = new World(26, 0); //new world with changes
```

```
Microsoft Visual Studio Debug Console

Total Distance hopped after the new world for each bird:

Name:Penguin
Total Distance:4ft
Distance per action: Flying:0ft Hopping:2ft

Name:Hawk
Total Distance:200ft
Distance per action: Flying:100ft Hopping:0ft

Name:Robin
Total Distance:22ft
Distance per action: Flying:20ft Hopping:1ft

Name:Crow
Total Distance:32ft
Distance per action: Flying:30ft Hopping:1ft

C:\Users\Masha\source\repos\birdTest\birdTest\bin\Debug\netcoreapp3.1\birdTest.exe (process 6280) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Results:

Penguin hops both worlds, but does not fly in either.

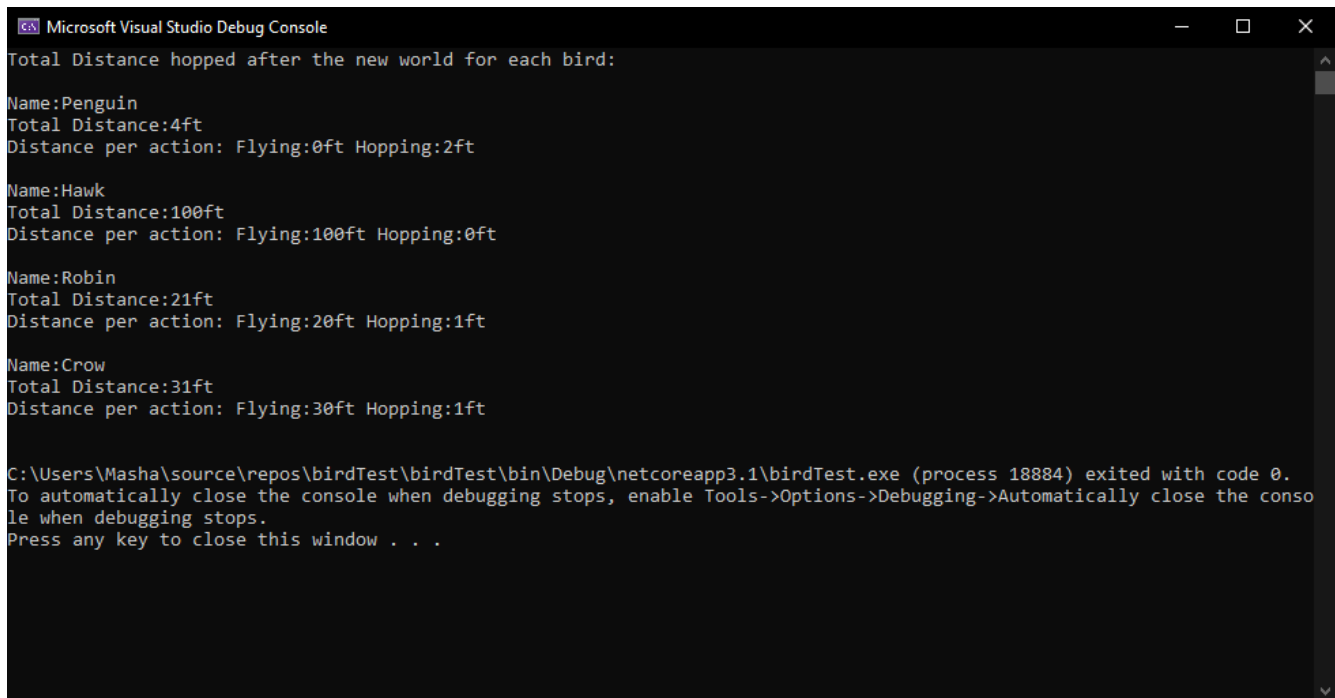
Hawk flies both in both worlds, but does not hop in either.

Robin does not fly in world 2, but hops and flies in world 1.

Crow does not fly in world 2, but hops and flies in world 1.

Test 5: Testing world temperature for all birds except Penguin

```
World one = new World(0, 0); //default with no changes
World two = new World(41, -10); //new world with changes
```



```
Microsoft Visual Studio Debug Console
Total Distance hopped after the new world for each bird:
Name:Penguin
Total Distance:4ft
Distance per action: Flying:0ft Hopping:2ft

Name:Hawk
Total Distance:100ft
Distance per action: Flying:100ft Hopping:0ft

Name:Robin
Total Distance:21ft
Distance per action: Flying:20ft Hopping:1ft

Name:Crow
Total Distance:31ft
Distance per action: Flying:30ft Hopping:1ft

C:\Users\Masha\source\repos\birdTest\birdTest\bin\Debug\netcoreapp3.1\birdTest.exe (process 18884) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

Results:

Penguin hops both worlds, but does not fly in either.

Hawk flies in world 1 only, but does not hop in either.

Robin flies and hops in only world 1.

Crow flies and hops in only world 1.

Part 2:

Answer to Questions:

A) The design approach is object oriented programming and using the proper data structures to make the code manageable and simple to use. I believe these were the best used mainly because we are working with a list and we want to keep the information stored nicely. I used object oriented programming because it works well with utilizing the different objects, in this case it was the world

objects and the bird classes, by putting them in their own classes and calling them out it keeps the information organized and easy to access. I also utilized lists with the foreach loop as it helped with making the code look neater. I originally had no loops, but the output loop, and used the methods fly() and hop() function to make the birds fly and hop, but then decided instead of having to do that, a loop can do it instead and it gives the user or another coder less things to have to add or do in the program.

I also added 3 constructors for the bird, a default constructor that takes all the parameters and an overloaded one that takes all parameters except for the distance traveled. If a new bird is created and already has a distance traveled, that distance can be taken as a parameter. If not, you can create a bird without putting in a distance parameter and the distance will initialize as zero which makes it easier for anyone using the program as it saves time when creating birds(objects) that haven't traveled a distance yet.

B) If you were to add a bird for hopping and flying there would be very little that needs changing. The person adding the bird create the object with the needed variables for that bird(name, flying distance, hopping distance, if bird can fly, if bird can hop) and then add it to the list. The program will do the rest and the person would not need to even go to any other classes, they can do everything in the main().