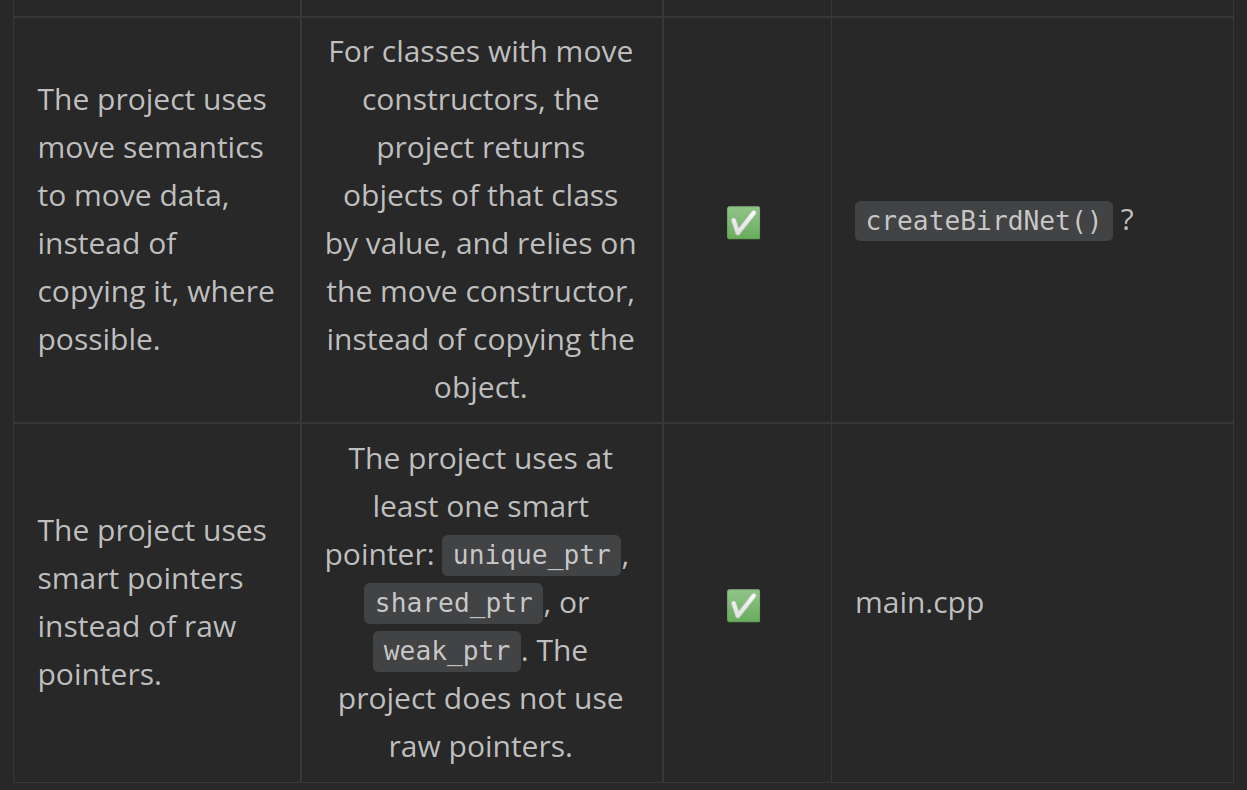
**Satisfait les spécifications**

**Conguratutation! You just graudate 🎓️🎓️**

Very impressive work Udacian!

**You did a great job creating a creative game with minimal modifications. Additionally, game information such as the player's life and the number of birds is also displayed well on the screen.**  
[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696589898/Screenshot_from_2023-10-06_19-58-10.png)

**It was nice going through your work. You implemented the following methods in your project well.**  
🎯 Using Object Oriented Programming concept to organize your code nicely  
🎯 Using MultiThread technique to run multiple objects at once

**Furthermore, you summarize the README really well 💪 Thank you for checking the rubric point what is met as green emote and function name.**  
[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696590204/Screenshot_from_2023-10-06_20-03-07.png)

**Some suggestions:**

* Try not to use the raw pointer type as much as possible  
  [](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696590344/Screenshot_from_2023-10-06_20-05-39.png)
* You can allow users to enter their names and save it in a text file with their scores, so you can get the top score.
* You can allow the user to select the initial speed (Slow - Fast).

**Additional Resources**

If you want to create a system that can create various games with a single C++ code, a slightly higher-level concept is needed.

* The first one is for that is [Entity-Component-System](https://www.codingwiththomas.com/blog/an-entity-component-system-from-scratch" \t "_blank)
* Additionally, you need to call code already written in C++ through another language in that case. You can use the [Scripting](https://eliasdaler.wordpress.com/2015/11/12/using-lua-with-cpp-in-practice-part-3/) method for this.

**README (All Rubric Points REQUIRED)**

**The README is included with the project and has instructions for building/running the project.**

**If any additional libraries are needed to run the project, these are indicated with cross-platform installation instructions.**

**You can submit your writeup as markdown or pdf.**

* The project contains clear instructions for **building and running** the project
* Also you can check [this link](https://www.makeareadme.com/" \t "_blank) for more information about writing a good readme file.

**The README describes the project you have built.**

**The README also indicates the file and class structure, along with the expected behavior or output of the program.**

* The description of the project was **so clear** 
* You can also learn more about how you can draw a diagram for your project from [this page](https://creately.com/blog/diagrams/class-diagram-tutorial/" \t "_blank)

**The README indicates which rubric points are addressed. The README also indicates where in the code (i.e. files and line numbers) that the rubric points are addressed.**

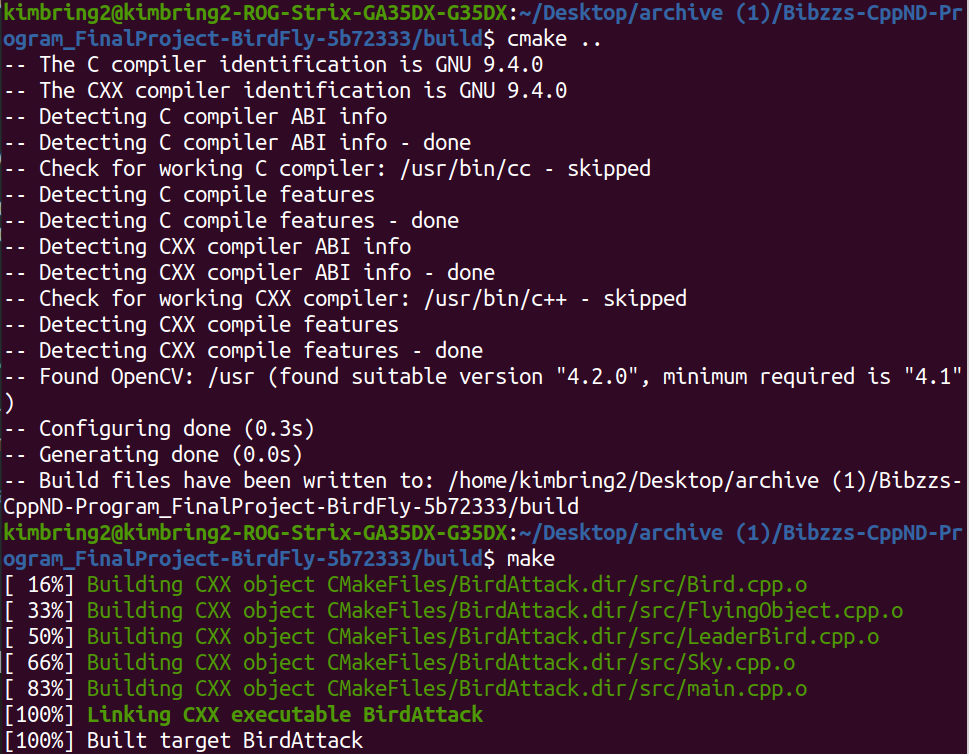
You already implemented **more than 5 rubric** points. Well done 

**Compiling and Testing (All Rubric Points REQUIRED)**

**The project code must compile and run without errors.**

**We strongly recommend using cmake and make, as provided in the starter repos. If you choose another build system, the code must compile on any reviewer platform.**

* Your code compiles without any errors 
* Here are some useful links for the Make file topic, you will need it entire life being a software engineer!
* [How to Build a CMake-Based Project](https://preshing.com/20170511/how-to-build-a-cmake-based-project/" \t "_blank)
* [Introduction to CMake by Example([http://derekmolloy.ie/hello-world-introductions-to-cmake/](http://derekmolloy.ie/hello-world-introductions-to-cmake/" \t "_blank))
* [Cmake tutorial([https://cmake.org/cmake/help/latest/guide/tutorial/index.html](https://cmake.org/cmake/help/latest/guide/tutorial/index.html" \t "_blank))

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696587938/Screenshot_from_2023-10-06_19-25-33.png)

**Loops, Functions, I/O**

**A variety of control structures are used in the project.**

**The project code is clearly organized into functions.**

* Control Structures are just a way to specify flow of control in programs.
* Any algorithm or program can be more clear and understood if they use self-contained modules called as logic or control structures.
* It basically analyzes and chooses in which direction a program flows based on certain parameters or conditions.
* There are three basic types of logic, or flow of control, known as:
  + Sequence logic, or sequential flow
  + Selection logic, or conditional flow
  + Iteration logic, or repetitive flow
* For more information, you can check [this link](https://www.geeksforgeeks.org/control-structures-in-programming-languages/)

**The project reads data from an external file or writes data to a file as part of the necessary operation of the program.**

You can allow users to enter their names and save them in a text file with their scores, so you can get the top score.

* You can capture the date too.
* Check [this link](https://www.tutorialspoint.com/cplusplus/cpp_date_time.htm) for more information about how you could do that

**The project accepts input from a user as part of the necessary operation of the program.**

You can allow the user to select the initial speed (Slow - Fast).

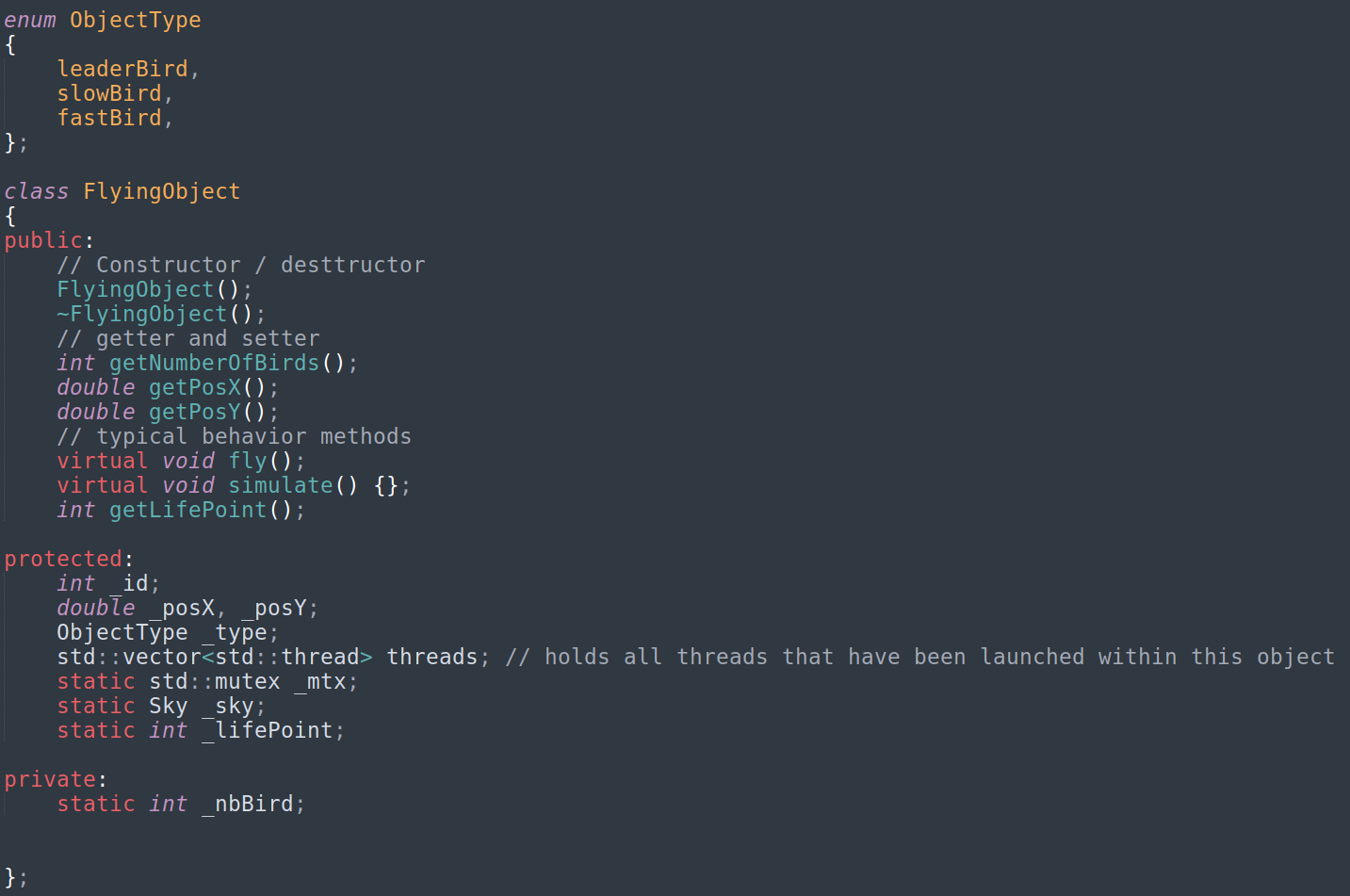
**Object Oriented Programming**

**The project code is organized into classes with class attributes to hold the data, and class methods to perform tasks.**

Well done for **structuring** your project into **classes**

**All class data members are explicitly specified as public, protected, or private.**

* **Public and private** access modifiers were used.
* You can check [this link](https://www.geeksforgeeks.org/access-modifiers-in-c/) for more information about their usages.

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696588111/Screenshot_from_2023-10-06_19-28-25.png)

**All class members that are set to argument values are initialized through member initialization lists.**

* This point was not addressed in the project
* The initialization list is used for  
  1) For initialization of non-static const data members  
  2) For initialization of reference members  
  3) For initialization of member objects which do not have default constructor  
  4) For initialization of base class members  
  5) When constructor’s parameter name is same as data member  
  6) For Performance reasons
* Check [this link](https://www.geeksforgeeks.org/when-do-we-use-initializer-list-in-c/" \t "_blank) to know more about **'When do we use Initializer List in C++'**?

**All class member functions document their effects, either through function names, comments, or formal documentation. Member functions do not change program state in undocumented ways.**

* Documenting projects is one of the most important topics in C++
* It helps others to understand your work and build on top of it
* Check [this link](https://www.bgsu.edu/arts-and-sciences/computer-science/cs-documentation/c-plus-plus-program-documentation-guidelines.html) to learn more about this topic

**Appropriate data and functions are grouped into classes. Member data that is subject to an invariant is hidden from the user. State is accessed via member functions.**

* **Getter** and **setter** methods can be used to access private members of classes
* double LeaderBird::getVy()
* {
* return \_spdY;
* }
* double LeaderBird::getVx()
* {
* return \_spdX;
* }
* [This link](https://www.w3schools.com/cpp/cpp_encapsulation.asp) talks about Encapsulation
* Feel free to check it

**Inheritance hierarchies are logical. Composition is used instead of inheritance when appropriate. Abstract classes are composed of pure virtual functions. Override functions are specified.**

* Well done for using the class inheritance  
  [](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696588679/Screenshot_from_2023-10-06_19-37-53.png)
* Check [this link](https://www.geeksforgeeks.org/inheritance-in-c/" \l "Why%20and%20when%20to%20use%20inheritance?" \t "_blank) for more information about **Why and when to use inheritance?**

**One function is overloaded with different signatures for the same function name.**

Excellent! You overload the fly functions well.

void FlyingObject::fly()

{

}

void LeaderBird::fly(double TIMESTEP)

{

std::lock\_guard<std::mutex> lock(\_mtx);

\_prevPosX = \_posX;

\_prevPosY = \_posY;

\_posX = \_newPosX;

\_posY = \_newPosY;

\_spdX = (\_posX - \_prevPosX) / TIMESTEP;

\_spdY = (\_posY - \_prevPosY) / TIMESTEP;

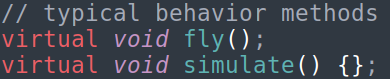
if (\_prevPosY != \_posY || \_prevPosX != \_posX) { \_budpated = true; }

//else { \_budpated = false; }

}

* C++ allows you to specify more than one definition for a function name or an operator in the same scope, which is called function overloading and operator overloading respectively.
* [This link](https://www.geeksforgeeks.org/operator-overloading-c/" \t "_blank) contains a brief introduction about that topic.

**One member function in an inherited class overrides a virtual base class member function.**

* **A virtual function** is a member function that you expect to be redefined in derived classes.  
  [](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696588918/Screenshot_from_2023-10-06_19-41-51.png)  
  When you refer to a derived class object using a pointer or a reference to the base class, you can call a virtual function for that object and execute the derived class's version of the function.

You can save [this link](https://www.geeksforgeeks.org/virtual-function-cpp/) in your notes to come back to it whenever you want to revise this topic quickly

**One function is declared with a template that allows it to accept a generic parameter.**

* This point was not addressed in the project.
* Check [this link](http://www.cplusplus.com/doc/oldtutorial/templates/) for more information about templates

**Memory Management**

**At least two variables are defined as references, or two functions use pass-by-reference in the project code.**

* This point was not addressed in the project
* In principle, this part should be excluded from the rubric point because you pass the pointer instead of the reference.
* Check [this link](https://www.educative.io/edpresso/pass-by-value-vs-pass-by-reference) for more information about the difference between the differences between pass by reference and pass by values

**At least one class that uses unmanaged dynamically allocated memory, along with any class that otherwise needs to modify state upon the termination of an object, uses a destructor.**

* Any new operator in your project must have a delete operator encounter to avoid memory leaks
* At the end of any class, check if you allocated a memory in the heap
* If **YES**, free this memory in the **class destructor** using the delete operator

**The project follows the Resource Acquisition Is Initialization pattern where appropriate, by allocating objects at compile-time, initializing objects when they are declared, and utilizing scope to ensure their automatic destruction.**

If you still don't know what is the meaning of **RAII**, check [this discussion](https://stackoverflow.com/questions/2321511/what-is-meant-by-resource-acquisition-is-initialization-raii)

**For all classes, if any one of the copy constructor, copy assignment operator, move constructor, move assignment operator, and destructor are defined, then all of these functions are defined.**

**Quick revision for the the difference between the shallow and deep copying**

* There are two different types of copying
* The first is the shallow copying, which is commonly used in the move operator method.  
  EX: \_image = source.\_image;
* The second is a deep copying, which is commonly used for copying the memory in the heap in the copy operator method.  
  Ex: \_image = new wxBitmap(\*source.\_image);

**For classes with move constructors, the project returns objects of that class by value, and relies on the move constructor, instead of copying the object.**

* Move semantics is about transferring resources rather than copying them when nobody needs the source value anymore.
* Check [this link](https://www.geeksforgeeks.org/move-constructors-in-c-with-examples/) to learn more about **Why Move Constructors are used**?

**The project uses at least one smart pointer: unique\_ptr, shared\_ptr, or weak\_ptr. The project does not use raw pointers.**

* A smart pointer is a class that wraps a 'raw' (or 'bare') C++ pointer, to manage the lifetime of the object being pointed to.
* Smart pointers should be preferred over raw pointers.

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696589587/Screenshot_from_2023-10-06_19-53-00.png)  
[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696589587/Screenshot_from_2023-10-06_19-52-44.png)

**Concurrency**

**The project uses multiple threads in the execution.**

Excellent! You use the multithread well to run the multiple bird object together

void Bird::fly(double timestep, LeaderBird\* leadBird)

{

while (leadBird->getStatus())

{

std::cout << "\_id: " << \_id << std::endl;

// sleep at every iteration to reduce CPU usage

std::this\_thread::sleep\_for(std::chrono::milliseconds((int)(timestep \* 1000)));

if (leadBird->getUpdate())

{

std::unique\_lock<std::mutex> lock(\_mtx); // ensure that the x,y or speed of leadbird are not change during the controller calculation.

// std::cout << " calculation of bird # " << this->\_birdID << std::endl; // DEBUG INFO

double deltaY = leadBird->getPosY() - \_posY;

double deltaX = leadBird->getPosX() - \_posX;

double deltaVy = leadBird->getVy() - \_vY;

double deltaVx = leadBird->getVx() - \_vX;

lock.unlock();

\_aY = 0.7 \* deltaY + 0.9 \* deltaVy;

\_aX = 0.7 \* deltaX + 0.9 \* deltaVx;

double noiseVX = randomInRange(-0.020, 0.020) \* deltaVx;

double noiseVY = randomInRange(-0.05, 0.05) \* deltaVy;

\_vY = \_vY + \_aY \* timestep + noiseVY;

\_vX = \_vX + \_aX \* timestep + noiseVX;

double noiseX = randomInRange(-0.05, 0.05);

double noiseY = randomInRange(-0.1, 0.1);

\_posY = 0.5 \* \_aY \* timestep \* timestep + \_vY \* timestep + \_posY + noiseY;

\_posX = 0.5 \* \_aX \* timestep \* timestep + \_vX \* timestep + \_posX + noiseX;

}

else

{

if (\_posY > \_sky.getHeight())

{

\_vY = 0;

\_aY = 0;

}

else

{

\_vY = \_vY + \_aY \* timestep;

\_posY = 0.5 \* \_aY \* timestep \* timestep + \_vY \* timestep + \_posY;

}

}

\_hit = targetReached(leadBird);

}

}

**A promise and future is used to pass data from a worker thread to a parent thread in the project code.**

* This point was not addressed in the project.
* std::thread is the thread class that represents a single thread in C++.
* To start a thread we simply need to create a new thread object and pass the executing code to be called (i.e, a callable object) into the constructor of the object.
* Once the object is created a new thread is launched which will execute the code specified in callable.
* Check [these tutorials](https://www.tutorialspoint.com/cplusplus/cpp_multithreading.htm), it contains some useful examples

**A mutex or lock (e.g. std::lock\_guard or `std::unique\_lock) is used to protect data that is shared across multiple threads in the project code.**

Excellent! You use mutex and lock functions to ensure that only one entity can access a resource at a time

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/32511/1696589711/Screenshot_from_2023-10-06_19-55-03.png)

* You can check [this link](http://jakascorner.com/blog/2016/02/lock_guard-and-unique_lock.html) to learn more about the difference between std::lock\_guard and std::unique\_lock

**A std::condition\_variable is used in the project code to synchronize thread execution.**

The condition\_variable class is a synchronization primitive that can be used to block a thread, or multiple threads at the same time, until another thread both modifies a shared variable (the condition), and notifies the condition\_variable.