# C++ SV Project

Jeremy Mion (261178)

December 11th 2018

## Q.1

### Q.1.1

The only logical thing to di is to create a private methode. In my case “clamp”

### Q.1.2

The reason that the default = and copy constructor are sufficient is that they both do a surface copy of the elements. Codding them by default allows the user of our class to know that they can safely use these constrcutors.

### Q.1.3

A double loop will allow for a more concise piece of code. There is no point in describing it in gory details in this file. See the actual code for implementation details.

### Q.1.4

Taking a const reference is almost always better. The only case where we decided to not use a const ref, is in the constructor since we want a copy of the vector to operate on it.

### Q.1.5

All the methodes that do not affect the Object should be constant. This is extremely important since C++ is strict on how it manages const functions. To see which ones where declared as const see the CircularCollider.hpp file.

### Q.1.6

We are simply adding another way to call : we are not writing code that has any functionality. This is important because it allows use to ensure that all of the functions that are designed to have the same behavior behave in the same way. isCircularColliderInside(other) isColliding(other); isPointInside(point);

### Q.1.7

I chose an internal definition for most of the functions because there equivalent in non operator function calls are also internal.

### Q.1.8

For which functions are receiving parameters by reference see .hpp. In all cases where we do not need a copy of the object to operate on it is better to use a const reference since it will improve performance.

### Q.1.9

See CircularCollider.hpp for the list on functions that we decided to declare as const.

## Q.2

### Q.2.1

The draw method should be const. The other ones will make modifications to the environment, therefor they cannot be constant.

### Q.2.2

Use the delete of the operator= and the copy constructor.

```
/*!
 * Disabeling copy constructor since this is a large object that should not be copied.
 */
Environment(const Environment&) = delete;
Environment& operator=(const Environment& env) = delete;
```

### Q.2.3

This means that when the Environment is destroyed it needs to destroy all the Animals.

### Q.2.4

**Warning to check with others.**

The definition of the use of the different keys is done in the EnvTest

### Q.2.5

Well we clearly want one method that will calculate the force that is being applied to the automaton. This method called attractionForce will return the force that the automaton is experiencing. This will allow classes inheriting of ChassingAutomaton to simply redefine the attractionForce and have the wanted behavior. The methode prototypes are as follows:

```
/*!
 * Calculates the attraction force that the automaton is experiencing
 * @return force that are being applied to the robot.
 */
Vec2d attractionForce() const ;

/*!
 * Makes the automaton moved based of of the force that it is experiencing.
```

```
     * @param force that the robot is experiencing
     * @param dt time that has passed since previous update
     */
    void updateMovementVariables(const Vec2d& force, const sf::Time dt );
```

## Q.2.6

We will for now declare a default value in the constructor of ChassingAutomaton. We have added it to the consctructor with a default value. We will adapt how this enum is set, once we have a better grasp of the context in which it would need to be changed. The enum names are of course capitalized to respect the convention that constants are written in capital letters. The main issue with using enums is of course that we cannot assign floating point values to the elements of an enum. Therefor we had multiple possibilities that where present. Store the enum as integer values by multiplying by a factor 10. This is far from ideal since it means that there is information that is encoded into the enum but that can be misinterpreted. The solution that we decided to apply in this case is to use a private methode to resolve the enum value to it's floating point equivalent.

## Q.2.7

The animal is a seperate entity. Therefor we can imagine that classes that extend the Animal class will want to influence the rotation of the animal. There is no reason why an external class such as the environment should be able to influence the rotation of the automaton. For the method setPosition does not exist since I have not had any use for it so far .

## QUESTIONS

- Why do we define a draw of a target when the environment does it for use?
- What is the methode setPosition of CircularCollder ref. Q2.7?