Hermmy Wang

Professor Smallberg

CS 32 Lecture 3

1 March 2018

Report

1. Description of complex member functions in Actor, NachenBlaster, Star, Explosion, Projectile, Goodie, and Alien.

class Actor:

* doSomething(): I chose to define it as a nonvirtual function in Actor class because every actor needs to check if it is alive. Inside this function, I will call a pure virtual function doSomethingDiff() if the actor is alive.
* doSomethingDiff(): This is a pure virtual function in Actor class. Each type of actor is able to do something , and they do it in different ways.
* sufferDamage(double d): This is a virtual function in Actor class. Aliens and Nachenblaster are able to suffer damage, and others simply die. Therefore, Nachenblaster and Aliens will deal with damage in their own ways, and others have a default version of sufferDamage in Actor class, which is set them dead.
* collideNB(): This is a nonvirtual function because every actor in Actor class collides with the NachenBlaster in the same way (except for the Nachenblaster can’t collide with itself, but it never calls this function).
* getDamagePoints(): This is a virtual function because each type of actors have different damage points (e.g. a cabbage has 2 and a torpedo has 8). For stars and explosion which do not cause damage, the base version in Actor class is to return 0.
* alienShip(): This is a virtual function in Actor class. The base version returns false, which can be applied to every type of actors except for alien ships. Alien’s version of alienShip() returns true, so StudentWorld can identity it is an alien ship when it deletes one.
* Other accessor functions, mutator functions, and const functions are nonvirtual because every actor behaves in the same way.
* class NachenBlaster:
  + doSomethingDiff(): This is a virtual function in NachenBlaster class. It is inherited from its base class.
    - Pseudocode:

Get key from Student World

If user presses Space and enough cabbages

create cabbages

activate them

play sound

decrement the number of cabbage energy points

If user presses Tab and enough torpedoes

create a torpedo

activate it

play sound

decrement the number of torpedoes

If user presses direction

move if it is not offscreen.

Increase the number of cabbage if it is not full.

* + sufferDamage(double d): This is a virtual function in NachenBlaster class, inherited from Actor class.
    - Pseudocode

Decrease hit points by d

Set it dead if hit points are below 0

* class Star:
  + doSomethingDiff(): This is a virtual function in Star class. It is inherited from its base Actor class.
    - Pseudocode:

Move 1 px to the left

If it is offscreen

Set it dead

* class Explosion:
  + doSomethingDiff(): This is a virtual function in Explosion class. It is inherited from its base Actor class.
    - Pseudocode:

Increase size by 1.5 every tick.

Increment the number of ticks.

If the number of ticks is greater than 4

Set it dead

* class Projectile:
  + doSomethingDiff(): This is a virtual function because each projectile needs to check if it is offscreen, if it collides with anything, if it continues to move, and if it collides with anything again.
    - Pseudocode:

If offscreen

Set it dead

Else

Check collsion

If not dead

Move the projectile

Check collision again

* + checkCollision(): It is a pure virtual function because each type of projectile is able to collide with different actors, and causes different amount of damage to the victim.
    - Pseudocode:

If collide with an alien

Let alien suffer damage

Set itself dead

If collide with the NachenBlaster

Let the NachenBlaster suffer damage

Set itself dead

* + moveProjectile(): It is a pure virtual function because each type projectile moves in different speed and direction.
* class Goodie:
  + doSomethingDiff(): It is a virtual function because it is inherited from its base Actor class. Every goodie needs to check if it is offscreen, if it is collected by the player, if it can continue to move, and if it collides with NachenBlaster again after moving. (Similar to Projectile’s doSomethingDiff).
  + collisionReaction(): It is a nonvirtual function because Goodie’s derived classes do not call it, and every goodie reacts to the collision in the same pattern.
  + bonus(): It is a pure virtual function in Goodie because each type of goodie gives the player different bonus.
* class Alien:
  + doSomethingDiff(): It is a virtual function because it is inherited from its base Actor class. Every alien ship does something in the same pattern.
    - Pseudocode

If offscreen

Set it dead

Else

Check collision

If not dead

Check if it needs new fight plan

Change flight plan

Check if the player is in the sight

Possibly Fire a projectile

Continue moving

Check collsion again

* + checkCollision(int damage, int score): It is a nonvirtual function in Alien class that checks if an alien collides with the NachenBlaster. Every alien ship checks collision with the player in the same way.
    - Pseudocode

If it collides with the NachenBlaster

NachenBlaster suffer some damage

Fatal collision for the alien ship

Possibly drop a goodie

* + fatalCollsion(): It is a nonvirtual in alien class because all aliens ship deals with fatal collision in the same fashion.
    - Pseudocode:

Increase the player’s score

Set it dead

Increment the player’s number of destroyed aliens

Play sound

Create and activate an explosion

* + sufferDamage(): It is a virtual function because it inherits from its base Actor class. It decreases a certain amount of hitpoints from the alien ship and makes it do something.
    - Pseudocode:

Suffer some damage

If the damage causes it to die

Fatal collision for the alien ship

Possibly drop a goodie

Else

Play sound

* + changeDirection(): It is a nonvirtual function because none of the derived classes need to call it, and every alien changes direction partially in this fashion.
    - Pseudocode:

If reaches top

Set direction to downleft

If reaches bottom

Set direction to upleft

Call differentiated version of change direction

* + changeDirDiff(): It is a virtual function in Alien class because Smallgons and Smoregons deal with changing direction in the default way, where Snagglegon has a different way and needs to override this function.
    - Pseudocode:

If flight length reaches 0

Randomly change to a different direction

Randomly choose a new flight length

* + move(): It is a nonvirtual function in Alien class. Every alien moves in the same way according to its different travel direction.
  + moveDiff():It is a virtual function in Alien class because Smallgons and Smoregons deal with changing flight length in the default way, where Snagglegon does not have a flight length.
  + dropGoodie(): It is a pure virtual function in alien class. Every alien needs to drop a goodie, but they drop it in different patterns. Smallgons are not able to drop; Smoregons and Snagglegons drop different goodies.
  + fire(): This is pure virtual function in alien class because every type of alien is able to fire. They fire different projectiles with different possibilities.

class StudentWorld:

* init(): It is a virtual function inherited from GameWorld.
  + Pseudocode:

Repeatedly 30 times:

Create a new star at a random location

Push the star into the actor vector

Create a new NachenBlaster

Initialize data members

* move(): It is a virtual function inherited from GameWorld.
  + Pseudocode:

For every actor in the actor vector

If the actor is not dead

Let it do something

If the player died

Decrement a life

Return player died

Else

If the level is finished

Play sound

Return finished level

Else

Erase the dead actor from the actor vector.

If it is an alien

Decrement the counter from alien ships

Delete the dead actor

Let the player do something

Update game text

Possibly insert stars

Possibly insert Aliens

Return continue game

* cleanUp(): It is a virtual function inherited from GameWorld.
  + Pseudocode

If the actor collection is not empty

Push the player into the collection

Repeatedly:

Erase every actor in the collection

* animate(Actor\* obj): It is a nonvirtual function because Student World does not have a derived class. It allows the actor being passed in to do something.
* collide(Actor\* obj): It is a nonvirtual function because Student World does not have a derived class. It checks if the actor being passed in, which should be a player projectile, collides with an alien ship. If it does, return a pointer to that actor, so the player projectile can let the alien ship suffer damage.

1. I did not fail any functionality.
2. The specification did not clarify