1.

-studentworld

Actor\* getOneCollidingAlien(const Actor\* a) const;

It returns an pointer to an alien if the actor a points to collides with any alien in the current tick

NachenBlaster\* getCollidingPlayer(const Actor\* a) const;

It returns the pointer to the nachenblaster of the actor a points to collides with the nachenblaster in the current tick

bool playerInLineOfFire(const Actor\* a) const

it returns whether the nachenblaster is on the left of the actor a points to and has a y coordinate within [4,-4] of that of the actor a points to

void addActor(Actor\* a);

it adds and actor to the studentworld

void recordAlienDestroyed()

it keeps count of the number of aliens destroyed to see how many more aliens the nachenblaster need to destroy in order to win the current level

-Actor

StudentWorld\* getWorld();

It returns a pointer of the studentworld the actor belongs to

virtual void doSomething() = 0;

it’s what an actor do during each tick. It’s pure virtual because different actors do different things

virtual string type() = 0;

It returns what thing an actor is (nachenblaster, cabbage…). It’s pure virtue because different kinds of actors return different things

bool isdead();

it returns if the actor is dead

void die();

it kills the actor

virtual void sufferDamage(double amt, int cause)

It makes the actor suffer amt amount of damage caused by cause. It’s defined here instead of in the damageableobject class is because the getcollidingalien function in the studentworld function returns a pointer to an actor, not to an damagebleobject, and that function would later call the sufferdamage function. I didn’t make it virtue so that I don’t need to redefine it in subclasses that can’t get damage like star or goodie

-Star

virtual void doSomething();

it does things that star is supposed to do. Specifically, it moves itself 6 pixel to the left

string type()

it return “Star”, type of this actor

-Explosion

string type()

it returns “Explosion”, type of the actor

virtual void doSomething();

specific behavior of explsion

-DamageableObject

double hitPoints() const

return the number of hitpoints the actor have left

void increaseHitPoints(double amt)

increase hitpoints by amt

virtual void sufferDamage(double amt, int cause)

suffer a damage. It’s virtue because each kinds of damageableobjects reacts differently when suffering a damage. Didn’t make it pure virtue for the same reason as the sufferdamage function in actor

-NachenBlaster

virtual void doSomething();

do specific thing (take keys and react accordingly etc. ) that NachenBlaster dpes

string type()

return type

void increaseTorpedoes(int amt)

increase number of torpedoes by amt

int healthPct() const

return health percent

int cabbagePct() const

return cabbage percent

int numTorpedoes() const

return number of torpedoes

void sufferDamage(double amt, int cause);

suffer damage of amt amount from cause cause in a specific nachenblaster way

-Alien

virtual void sufferDamage(double amt, int cause);

suffer damage in a specific alien way. Didn’t make it pure virtual because all aliens suffer damage in the same way

void doSomething();

do the thing that are common for all aliens

virtual bool dospecialthing() = 0;

do specific thing that is different for different kinds of aliens. It’s pure virtue because it’s different for different aliens

void move()

move in the way an alien should

void setDeltaY(double dy)

set the direction of movement of the alien to dy

void setSpeed(double speed)

set the speed of the alien to speed

bool damageCollidingPlayer(double amt);

returns if the alien collide with a nachenblaster. If it does, make the nachenblaster suffer the damage and kill the alien

-Smallgon

bool dospecialthing();

do things that only smallgon should do, and returns true if fires a turnip

string type()

return type name

-Smoregon

bool dospecialthing();

do the specific thing that only smoregon would do, and return true if it shoots a turnip

string type()

return type name

-Snagglegon

string type()

return type name

bool dospecialthing();

do the specific thing that only a snagglegon should do. Returns true if it fires a torpedo

-Projectile

void doSomething();

do things that all projectile share in common. It’s not virtue because it’s the same for everybody

virtual bool dorealthing() = 0;

do specific thing each kind of projectile should do. It’s pure virtue because it’s different for different kinds of projectiles

-Cabbage

bool dorealthing();

do things that only cabbage should do and return true if it collide with an alien

string type()

return type name

-Turnip

bool dorealthing();

do specific things that only turnip should do. Return true if it collide with an alien

string Type()

return type name

-PlayerLaunchedTorpedo

bool dorealthing();

do thing that only playerlaunchedtorpedo would do, and return true if it collides with an alien

string type()

return type name

-AlienLaunchedTorpedo

bool dorealthing();

do things that only alienlauchnedtorpedo would do, and return true if it collides with a nachenblaster

string type()

return type name

-Goodie

void doSomething();

do things that all goodies do

virtual void doSpecialthing(NachenBlaster\* ptr) = 0;

do things that specific goodies should do when interacting with a nachenblaster. It’s pure virtual because it is different for different goodies

-ExtralifeGoodie

String Type()

Return type name

void doSpecialthing(NachenBlaster\* ptr)

do specific thing that only extralifegoodie would do when colliding with nachenblaster ptr point to

-RepairGoodie

String Type()

Return type name

void doSpecialthing(NachenBlaster\* ptr)

do specific thing that only repairgoodie would do when colliding with nachenblaster ptr point to

-TorpedoGoodie

String Type()

Return type name

void doSpecialthing(NachenBlaster\* ptr)

do specific thing that only torpedo goodie would do when colliding with nachenblaster ptr point to

2. didn’t find any

3.

It’s not clear if actors newly created by other actor’s dosomething, for example, a cabbage created by Smallgon’s dosomething ,should dosomething itself during the current tick. I assume that it would.

4.

StudentWorld

First I used the sanity check to see if there is any memory leak or problems of such. Then I made it print how many more aliens should nachenblaster destroy before going to the next level and then play the game to test if it goes to the next level as requested by the spec. Then I play multiple times of the game to see if the line at the top follows what the spec asked. Then I play the game byte by byte to see if the number of aliens on the screen follow the request of the spec.

Explosion

Crash the nachenblaster to an alien and play it byte by byte to see if the explosion lasts for exactly four ticks

Nachenblaster

First see if it’s initialize in the position specified by the spec. Them input different keys to see if it reacts accordingly. Then check if it could go out of the screen.

Projectile

Fire cabbage and then play byte by byte to see if it is created at the position specified by the spec and rotates). Then do the same to torpedo and turnip. Also, check if they create damage as specified by the spec to other actors.

Goodie

Change the init function in the studenworld class and create some goodies at the beginning of the game. Check if it only goes to down and left. Then get those goodie and see if Nachenblaster get what the goodies provide.

Alien

Check if the alien fly the way specified by the spec. Then see how many cabbage does it take to kill each alien. Then check if they drop goodies at all. Crash Nachenblaster to them and see if they die.