Fiona Tang
604402825

Project 4 Report

<u>StudentWorld Class</u>

- Derived from GameWorld

struct Point

- Used for evaluating exit paths as it is a convenient way to store and access paired coordinates (x, y)

enum ActorType

- Defined a type that would reference a particular Actor class and could be passed as a function parameter to elicit class specific behaviors

virtual ~StudentWorld()

- Destructor
- All destructors should be virtual to follow OOP conventions

Iceman getIceman()*

- Accessor function to m_iceman

std::vector<Actor> getActors()*

- Accessor function to vector of actors that exist in the world, excluding iceman and ice

int getBarrels() const

- Determines number of barrels are left in level
- Can determine if player completed a level

int getProtesters() const

- Determines number of protesters that exist in world
- Can Determine if maximum number of protesters for a level is reached

bool isBoundary(ActorType type, int x, int y) const

- Determines if the coordinates of an Actor with a 4x4 grid will overlap restricted areas such as:
    - The playable window boundary
    - Boulders have additional boundaries of where it can be placed
    - Gold, Barrels, and Boulders can not be placed where there is no ice initially
- Prevents moving or placing actors where they shouldn't be

bool isBoulder(Actor a, int x, int y) const*

- Determines if the coordinates of an Actor with a 4x4 grid will overlap the grid of a boulder

- The implementation of checking the 4x4 area of both Boulder and Actor was far too slow, so instead the Euclidian distance between the coordinates of both objects cannot be less than 3
- Prevents moving or placing actors where they shouldn't be

bool isIce(int x, int y) const
- Determines if ice is present at coordinate (x ,y)
- Prevents moving or placing actors where they shouldn't be
- Also helps determine if iceman needs to dig ice

bool isIceGrid(int x, int y) const
- Determines if ice is present at a 4x4 grid starting at coordinate (x ,y)
- Prevents moving or placing actors where they shouldn't be

void setDisplayText()
- Configures the text that should display at the top of the screen

bool clearIce(int x, int y)
- Deletes ice object at coordinate (x ,y)
- Useful for when iceman is digging

double distance(int x1, int y1, int x2, int y2) const
- Calculates distance between two points

bool euclidian6(int x, int y) const
- Determines if two objects are less than a distance of 6 from each other
- Prevents placing new object too close to an existing one

bool wiRadIceman(Actor a, double radius) const*
- Determines if an Actor is within a specified distance from iceman
- Certain actions occur if Actor is close enough to iceman:
    - Protester annoys or follows
    - Goodie is picked up
    - Boulder bonks

bool isFacingIceman(Actor a) const*
- Determines if Actor is facing in the direction of iceman
- Certain actions occur if Actor is facing to iceman:
    - Protester annoys

GraphObject::Direction lineIceman(Actor a) const*
- Determines if Actor is inline horizontally or vertically with iceman and in which direction iceman is

- Certain actions occur if Actor is close enough to iceman
    - Protester follows

*bool wiRadProtester(Actor *a, Actor* protester, double radius) const*

- Determines if an Actor is within a specified distance from Protester
- Certain actions occur if Actor is close enough to Protester:
    - Squirt of water stuns and annoys protester
    - Boulder bonks
    - Gold is picked up

bool isFacingProtester(Actor protester, Actor* a) const*

- Determines if Actor is facing in the direction of a Protester
- Certain actions occur if Actor is close enough to Protester:
    - Squirt of water stuns and annoys protester

int chanceOfGoodie(int chance)

- Determines whether a new sonar kit or water pool should be generated during tick based on specified probabilities

int chanceOfProtester(int chance)

- Determines whether a new Regular or Hardcore Protester should be during tick generated based on specified probabilities

void insertRandom(int amt, ActorType type)

- Inserts new object of type boulder, gold, barrel, or water to the world at a random coordinate, but acceptable location
- Function takes into account the different constraints for each type

void addActor(Actor add)*

- Adds an actor to the m_actors vector

void addObjIceman(ActorType type)

- Adds objects pickable by iceman to iceman and commits item specific actions such as:
    - Play appropriate sound
    - Add appropriate points to score

void destroyAll()

- Deallocates all dynamic memory
- Useful for destructor and cleanup methods

std::vector<Point> pathToExit(Protester pro, int inX, int inY)*

- Determines quickest path to exit from initial coordinates using:
    - A queue that pushes open and undiscovered coordinates

    - 2D array that tracks which coordinates have already been discovered
    - Returns a vector of what ordered coordinates for path

virtual int init()

- Initializes world:
    - Dynamically allocates iceman, ice, boulders, gold, barrels and a protester

virtual int move()

- Updates game world during each tick by:
    - Updating display text
    - Calling iceman doSomething
        - Digging/removing ice if iceman moves to new coordinate
    - Calling all other Actor doSomething
    - Deletes dead actors and makes it nonvisible
    - Checks if new actors actors should be added and adding if so:
        - Water pool or Sonar kit
        - Regular or Hardcore Protester
    - Checks the status of iceman
        - If dead, a life is lost
        - If no barrels left, level is finished
        - Otherwise game level continues

virtual void cleanUp()

- Deallocates all dynamic memory

Actor Class

- Derived from GraphObject
- Base class for all game objects

Actor(StudentWorld world, int imageID, int startX, int startY, Direction dir = right, double size = 1.0, unsigned int depth = 0)*

- Initializes actor to be initially
    - Visible
    - Alive
    - Permanent
    - Able to be picked up by iceman – majority of game objects are
    - Ticks since existence to 0

virtual ~Actor()

- Destructor
- All destructors should be virtual to follow OOP conventions
- Makes Actor:
    - Nonvisible

  - Dead
  - Cannot be picked up by iceman

StudentWorld getWorld() const*

- Accessor function to student world

bool isAlive() const

- Determines if Actor is alive

bool isPermanent() const

- Determines if Actor is permanent

bool isPickableIceman() const

- Determines if Actor can be picked up by iceman if close enough

int getTicks() const

- Get ticks that an item has existed If set to temporary

virtual bool isStunned() const

- Determines if Actor is stunned
- Used by protesters if hit by squirt, must be accessible by actor array in student world

virtual bool isBonked() const

- Determines if Actor is bonked by boulder
- Used by persons class to determine if killed by boulder, important for points allocated for how protesters that are killed

virtual bool isFixated() const

- Determines if Actor is fixated by gold
- Used by Hardcore protesters that pick up gold

void setDead()

- Sets status from alive to dead

void setTemp()

- Make Actors temporary

void setPickableIceman(bool pickable)

- Make Actors able to pick up by iceman

virtual void setStun(bool stun)

- Make Actors stunned
- Used by protesters if hit by squirt, must be accessible by actor array in student world

virtual void setBonked(bool bonk)

- Sets Actor status to bonked by boulder
- Used by persons class to determine if killed by boulder, important for points allocated for how protesters that are killed

virtual void setFixated(bool fix)

- Set Actor status to fixated by gold
- Used by Hardcore protesters that pick up gold

virtual void setLeave()

- Set status to leaving
- Used by protesters

void addTick()

- Add ticks

virtual void isAnnoyed(int annoy)

- Virtual function to annoy persons and reduce hp
- Not pure virtual because must be accessed by actor vector of world

virtual void doSomething() = 0;

- Pure virtual function for each derived class to commit actions each tick

<u>Ice Class</u>

Ice(StudentWorld world, int x, int y)*

- Constructor
- Set pickable by iceman to false

virtual void doSomething()

- Ice does not do anything

Boulder(StudentWorld world, int x, int y)*

- Initialize item to not pickable by iceman
- Must start as stable and not falling

bool isStable() const

- Determines if Boulder is stable
- Used to see if Boulder is waiting to fall

bool isFalling() const

- Determines if Boulder is falling
- Used to determine if boulder should be moved

void setUnstable()

- Set from stable to unstable

void setFalling(bool fall)

- Set from not falling to fall

void fall(int x, int y)

- *Makes boulder fall down one unit until it hits ice or another boulder*

bool personUnder()

- Determines if a person is under a boulder as it falls
- If iceman is under live is lost
- If protester is under, protester is fully annoyed and is set to leaving

virtual void doSomething()

- Commits Boulder actions at every tick. Boulder may:
  - Become unstable
  - Set to falling
  - Fall

Gold Class

Gold(StudentWorld* world, int x, int y)

- Gold is initially invisible and not pickable by protester

bool isPickableProtester() const

- Determines if gold is pickable by protester
- Happens when gold is dropped by iceman

void droppedGold()

- Set gold member variables to properties of being dropped by iceman
  - Not pickable by iceman
  - Pickable by protester
  - Temporary

virtual void doSomething()

- Commits Boulder actions at every tick. Boulder may:
  - Add tick if not temporary state
  - Reveal if less than distance of 4 from iceman
  - Get picked up by iceman and set to dead if distance is less than 3 from iceman
  - Set to dead if ticks are more than 100 when temporary
  - If dropped by iceman, allow to be picked up by
    - Regular protester – play sound, set to leave and add 25 points

- Hardcore protester – play sound, set to fixated and add 50 points

Barrel Class

Barrel(StudentWorld world, int x, int y)*

- Initially not visible

virtual void doSomething()

- Reveal if distance is less than 4 from iceman
- Get picked up by iceman if distance is less than 3 from iceman

Sonar Kit Class

SonarKit(StudentWorld world)*

- Set to temporary

virtual void doSomething()

- Get picked up by iceman if distance is less than 3 from iceman and add charge
- Set to dead if existed for over specified ticks

Water Pool Class

WaterPool(StudentWorld world, int x, int y)*

- Set to temporary

virtual void doSomething()

- Get picked up by iceman if distance is less than 3 from iceman and add 5 squirts
- Set to dead if existed for over specified ticks

Squirt Class

virtual void doSomething()

- Move in initial direction with every tick for 4 points or until it hits a boundary, ice, boulder, or protester

Person Class

- Base class for all "live" objects: iceman, protesters

double getHP() const

- Return health points

void setHP(double newHP)

- *Sets health points*

virtual void isAnnoyed(int annoy)

- When annoyed, reduce hp for appropriate amount
- Set to 0 if zero is passed as parameter

virtual void doSomething() = 0;

- Pure virtual function as Person class should not be able to call function

<u>Iceman Class</u>

int getGold() const

- Access number of gold iceman has
- For checking if a nugget can be dropped

int getSquirts() const

- Access number of squirts iceman has
- For checking if iceman can squirt

int getCharge() const;

- Access number of charges iceman has
- For checking if a sonar can be used

void addGold()

- Increment gold

void addSquirts()

- *Increment squirts*

void decSquirt()

- *Decrement squirts*

void addCharge()

- *Increment charge*

void decCharge()

- *Decrement squirts*

void dropGold()

- Have iceman drop gold, add actor to world, decrease nugget

void squirt()

- Have iceman squirt, initiate moving direction add actor to world, decrease squirts *void useSonar()*

- Reveal nearby items, decrease sonar

virtual void doSomething()

- Get key inputs
    - Escape key – set iceman to dead
    - Direction key - Move in direction by one as long as its not a boundary or boulder
    - Tab key – drop gold
    - Space key – squirt
    - Z key – use sonar
- Set to dead if hp is equal or less than 0

Protester Class

- Base class for regular and hardcore protesters

bool isLeaving() const

- Determines if protester to leaving

virtual bool isBonked() const

- Checks if bonked
- Virtual so accessible by actor ptr

virtual bool isStunned() const

- Checks if stunned
- Virtual so accessible by actor ptr

virtual int getStunTicks() const

- Return number of ticks passed while stunned

int getTicksToMove() const

- Return number of ticks to next move

getTicksToTurn() const

- Return number of ticks next turn

int getRestTicks() const

- Return number of ticks since last action

int getTurnTicks() const

- Return number of ticks since last turn

int getShoutTicks() const

- Return number of ticks since last shout

Fiona Tang
604402825

int getPerpendicularTicks() const

- Return number of ticks since last turn due to encountering a intersection

virtual void setLeave()

- Set to leaving
- Virtual so accessible by actor ptr

virtual void setBonked(bool bonk)

- Set to bonked
- Virtual so accessible by actor ptr

virtual void setStun(bool stun)

- Set to stunned
- Virtual so accessible by actor ptr

virtual void setStunTicks(int ticks)

- Set number of ticks passed while stunned

void setTicksToMove()

- Set number of ticks to next move

void setTicksToTurn()

- Set number of ticks next turn

void setRestTicks(int ticks)

- Set number of ticks since last action

void setTurnTicks(int ticks)

- Set number of ticks since last turn

void resetShoutTicks()

- Reset number of ticks since last shout to 0

void resetPerpendicularTicks();

- Reset number of ticks since last turn due to encountering a intersection to 0

void addNonRestTicks()

- Increment nonrest ticks: shout and perpendicular ticks

bool isBlocked(int x, int y, Direction dir)

- Determines if space is placed by ice, boulder, or bouldary

Fiona Tang
604402825

Direction randomDirection()

- Select random direction

bool intersectionTurn()

- Figures if intersection turn is viable, change if so

bool inlineIceman()

- Determines if iceman is inline vertically or horizontally with protester

void move(int inX, int inY, Direction moveDir)

- Move after changing boundaries

void annoy()

- Annoy iceman

virtual void doSomething() = 0;

- *Pure virtual as should only be called by derived classes*

<u>Regular Protester Class</u>

virtual void doSomething();

- If protester is less than or equal to 0, set to dead and increase player score based on
  - If killed by boulder
  - If killed by squirts
- Don't move if stunned
- If gold is picked up, leave
- Shout at iceman if facing and close enough
- Check if various ticks should be incremented, reset and appropriate action commited
- If is leaving, get path vector and move one step in path
  - Recall path function in case iceman opens up new, faster path
- If inline with iceman, follow

virtual void doSomething();

- If protester is less than or equal to 0, set to dead and increase player score based on
  - If killed by boulder
  - If killed by squirts
- Don't move if stunned
- If gold is picked up, become fixated
- Shout at iceman if facing and close enough
- Check if various ticks should be incremented, reset and appropriate action commited
- If is leaving, get path vector and move one step in path
  - Recall path function in case iceman opens up new, faster path
- If inline with iceman, follow

Fiona Tang
604402825

Bugs and Unfinished Implementations

- pathToExit function has bugs. Cannot determine a correct path if at some intersections
- Hardcore protester ability to track iceman is not implemented

Test Cases

- Correct number of barrels and gold are initialized in world and hidden and only in ice
- Correct number of boulders are initialized in world and not in ice or boundaries
- Water pool appears in non ice areas and disappears after certain amount of ticks
- Sonar kit appears at 0,60 and disappears after certain amount of ticks
- Barrels, gold, water pool, sonar kit get added to iceman inventory correctly
- Iceman dies if bonked by boulder or annoyed enough
- Protester leaves if bonked by boulder or annoyed enough
- Gold nugget dropped by iceman can be picked up by protester or disappears after certain number of ticks
- Regular protester leaves If picks up a gold nugget
- Hardcore protester becomes fixated if picks up a nugget for certain number of ticks
- Protesters will follow iceman if close enough
- Protesters don’t move every tick
- Protesters only shout at iceman is close enough and if less than 15 ticks from last shout