Humza Ahmed

CS 162

Final Project Design Plan

Main

Description
A slowly revealed text based story/game where a time traveler that wakes up in purgatory finds out that he killed his own grandfather if he doesn't solve the mystery in time he fades away

Create Game object

Loop (!win && !dead && timeLeft > 0)
    game.Gameplay()

Space Class (Abstract with pure vitual functions
    Space *top, *right, *left = null, *bottom
    Struct Stack Inventory bag
    int timeLeft = 20 (minutes)

    int getTimeLeft()
    Space()
        top =

Purgatory
    bool feltOnce

    lookAround()
        if (flashlight)
            You shine your flashlight in the darkness…
            see book on floor and pick it up
            Text "Diary of Joseph Lynch…
            add book to inventory
        else

```
                    it is pitch black, you can't see a thing


        feelAround()
                if (feltOnce)
                        you enter the the chasm
                else
                        you feel around on you

        move()
                if flashlight is in inventory
                        Options(Look Around, feelAround)


                else if (feltOnce)
                        Options (Look Around, feelAround)
                        cout different text
                else
                        options (Look around, feelAround)
```

Future

```
        lookAround()
                if (flashlight)
                        You don't see anything of importance
                else
                        You see a very sleek rod with a red circle on it
                        You slowly start to remember that this is called a flashlight
                        add flashlight to pack

        visitScientist()
                scientist tells you he built time machine for you
                didn't know what you were going to do with it.
                he has since built another
                Options (Knock out scientist and get in, leave)
                if (knock out)
                        sent too far to the past!
                else
                        return
```

move()
    if flashlight and book in inventory
        Options (Look Around (nothing), Go back in the hole in the tree, Visit Scientist)
        allow user to visit scientist
        scientist creates time machine to send Joseph Back in time
        goes too far back!
        time
    if flashlight in inventory
        Options (look around (nothing), Go back in the hole)
    else
        options (look around (find flashlight), go back in the hole)

Past
    bool tusk, gun;
    Takes place during the ice age. Have to kill a mammoth
    lookaround()
        std::cout story
        options to kill or run
        if (gun && kiil)
            kill mammoth
        else if (!gun && kill)
            die
        else
            try to run and die

    changeSpace(*ptr)
        allow user to fix time machine and head back to future or allow them to freeze and go to present

        ptr = past or ptr = present

    move()
        story continues
        if (tusk && gun)
            allow user to changeSpace
        else
            lookAround or grabGun

Present
    end of story
    move()
        if (?)

```
                    give user final options
                    one wins the game one kills you
            else
                    breakFree from ice or sitAndWait

Game
        bool win, dead;
        Space *past, *present, *future, *purgatory;

        bool getDead();
        bool getWon();

        Game()
                create space objects.
                set right, top, bottom appropriately
                player = purgatory

        ~Game()
                delete allocated mem

        gameplay()
                player->move()
                player->changeSpace(*ptr)
```

Testing Plan

Test Case	Input Values	Driver	Expected Outcomes	Observed Outcomes
Menu options	integer 1,2,3,		Correctly selected option	correctly selected option
invalid menu input	1.5, 1abc, a, @		Error message shown	error message shown
purgatory	move()		displays correct message based on inventory and calls correct function	displays correct message based on inventory and calls correct function
purgatory	lookAround()		displays correctly if lightOn	displays correctly if lightOn
purgatory	changeSpace()		change space to future	did not change at first
future	move()		displays correct message based on inventory and calls correct function	displays correct message based on inventory and calls correct function
future	changeSpace()		change space to purgatory or past	change space to purgatory or past
future	lookAround		display correctly if (flashlight == true)	display correctly if (flashlight == true)
past	move()		displays correct message based on inventory and calls correct function	displays correct message based on inventory and calls correct function
past	changeSpace()		change space to future or present	change space to future or present
past	lookAround()		display correctly if (gun == true)	display correctly if (gun == true)
past	lookAround()		die if encountering mammoth w/o gun	die if encountering mammoth w/o gun
present	move()		displays correct message based on inventory and calls correct function	displays correct message based on inventory and calls correct function
present	move()		dies/wins depending on choices made	dies/wins depending on choices made
game	gameplay()		player ptr switched to different spaces correctly	did not work at first. made player ptr return space and created an if statement to switch

game	timeLeft		timeLeft correctly decremented	timeLeft correctly decremented
game	~game()		deallocate memory	deallocate mem
game	game()		set top/right/bottom for space *ptrs appropriately	set top/right/bottom for space *ptrs appropriately
main	loop		exit loop if timeLeft == 0, gameWon or died	exit loop if timeLeft == 0, gameWon or died
memory leaks	valgrind		no mem leaks	no mem leaks

Reflection

I think the thing that I struggled with the most for this assignment was understanding the actual requirements themselves. Looking back on it, the requirements were intentionally vague so that each project could be unique, but they really ended up confusing me at first. After spending some time on Piazza and Slack I was able to put together what was actually required in my head and started attacking the problem. I ended up choosing to make my theme revolve around time travel paradoxes, because it seemed to be a great way to represent the nature of loops and recursion that occur in CS.

One of the biggest issues I had was figuring out how to actually change spaces in the program. At first I tried using player->setRight player->setBottom etc, but that would not work. I later had the move function return an integer and if that integer was a 2 it would then call a changeSpace function for that object. Then depending on what kind of object it was I would switch the player object to point to the correct object.

Another issue I ran into was the container. I couldn't for the life of me figure out how to make the container part of the Space class and also increment appropriately for wherever the player ptr would point. I later decided to just create an inventory class and link it to the Game class which helped immensely.

Overall I think this was a very fun project to write and it allowed me to really link my creative side with the logical side in a way that I have never done before. I'm very proud of the game and happy with how well it turned out. 162 taught me a lot in the past 8 week and now more than ever I look forward to continuing my education here to see what other classes have in store for me!