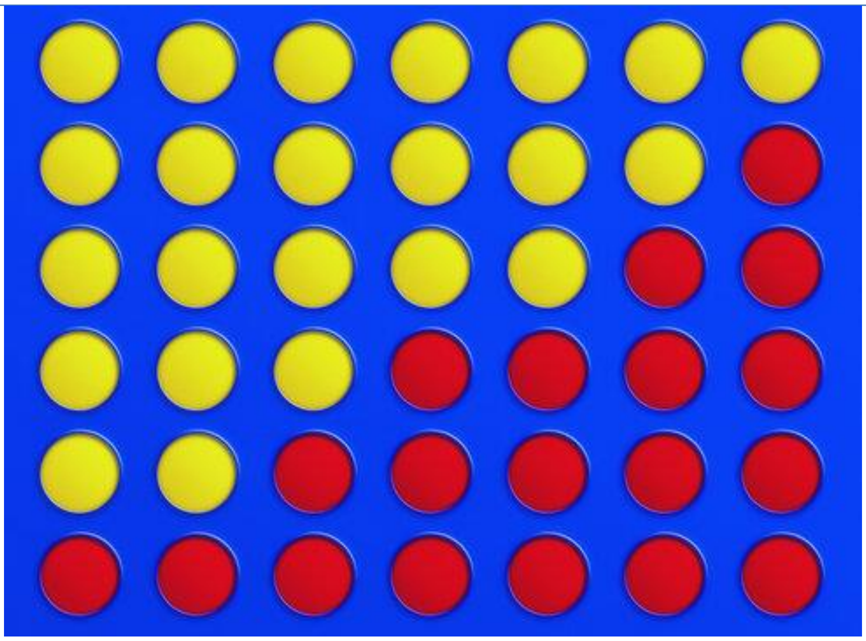
CPSC 2720 FALL 2024

[Major Project]



Group 4

"Connect 4"

John S. Anvik, Sofiah Plarisan, Gurmannat Sandhu, and Solomon Haile

Nov 6, 2024

SOFTWARE DESIGN

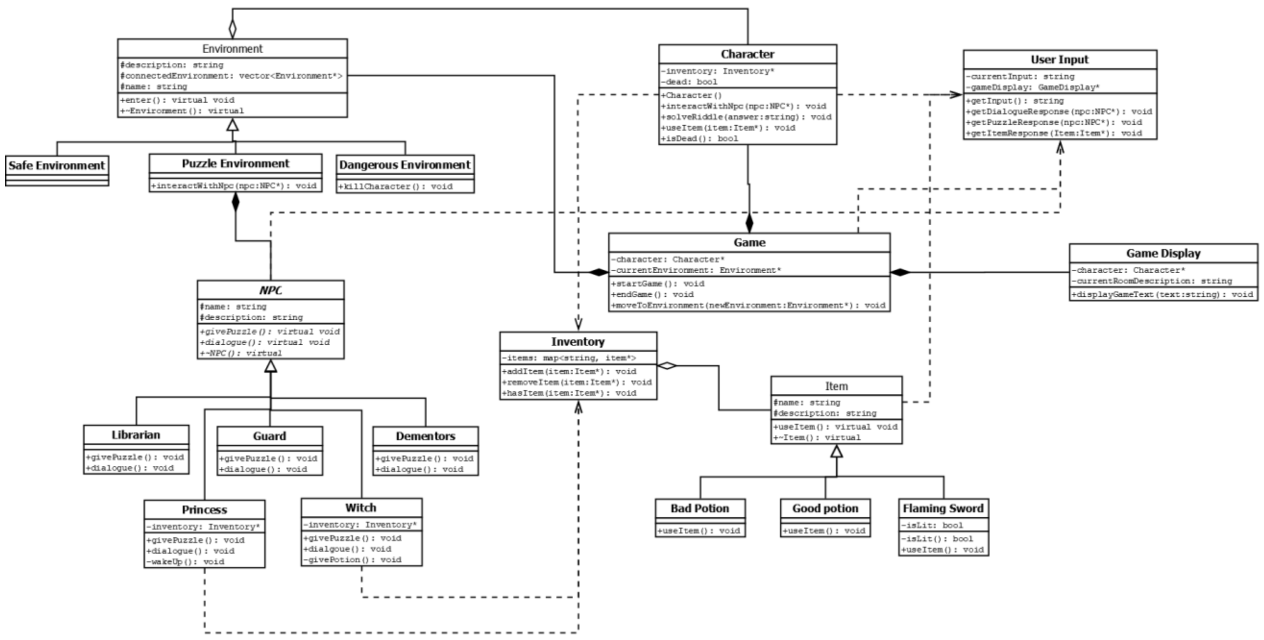
DESIGN – CLASS DIAGRAMS

CLASS DESCRIPTIONS

Class Name	Method Name	Description
Environment	enter(): virtual void	Will help the player enter a specific environment
PuzzleEnvironment	interactWithNpc(NPC*): void	Interacts with NPC specified in parameter
DangerousEnvironment	killCharacter(): void	Kills player character upon entering that environment
NPC	givePuzzle(): virtual void	Virtual method – each NPC subclass implements different puzzle behaviour
NPC	dialogue(): virtual void	Virtual method – each NPC subclass implements different dialogue behaviour
Librarian	givePuzzle(): void	Libararian gives a unique specified puzzle to player on that floor
Librarian	dialogue(): void	The dialogue that will be spoken to the player about what to do
Guard	givePuzzle(): void	NPC Guard gives the puzzle to player on their floor
Guard	dialogue(): void	Guard interacts with player by using dialogue
Dementors	givePuzzle(): void	Dementor gives a unique specified puzzle to player
Dementors	dialogue(): void	interact with player by using dialogue
Witch	givePuzzle(): void	gives a unique specified puzzle to player
Witch	dialogue(): void	Interact with player by using dialogue
Witch	givePotion(): void	If the player solves the puzzle by the witch, then the witch gives a good/bad potion to the player depending on if the solved puzzle was correct or incorrect (the player is unaware if it's good or bad)
Princess	dialogue(): void	interact with player by using dialogue
Princess	wakeUp(): void	The princess wakes up, causing the player to win the game
Princess	givePuzzle(): void	gives a unique specified puzzle to player
Inventory	addItem(Item*): void	Adds item pointed to by pointer parameter to an inventory

Inventory	removeItem(Item*): void	Removes item pointed to by pointer parameter from an inventory
Inventory	hasItem(Item*): bool	Returns true if item specified in parameter is found in inventory
Character	interactWithNpc (NPC*): void	Player character interacts with NPC (witch, librarian, etc.)
Character	solveRiddle(string): void	Checks if string parameter is answer to the riddle before the player
Character	useItem(Item*): void	Uses an item (specified by the parameter) in the character's inventory
Character	isDead(): bool	Returns true if player character is dead.
Game	startGame(): void	Starts the game
Game	endGame(): void	Ends the game
Game	moveToEnvironment(Environment*) : void	Moves player character to a different environment, specified by the parameter
Item	useItem(): virtual void	Virtual method – Items' subclasses will determine behaviour when that item is used
BadPotion	useItem(): void	The bad potion given by the witch to the player for the princess which when used at the end of the game kills the princess and the player loses
GoodPotion	useItem(): void	The good potion that saves the princess, when used player wins the game
FlamingSword	isLit(): bool	Returns true if flaming sword is ignited
FlamingSword	useItem(): void	Ignites the flaming sword.
UserInput	getDialogueResponse(NPC*): void	Takes user input to reply to a given NPC's dialogue.
UserInput	getPuzzleResponse(NPC*): void	Takes user input to reply to a puzzle.
UserInput	getItemResponse(NPC*): void	Takes user input to use an item.
UserInput	getInput(): string	Returns a string listing valid input options for the game's current state
GameDisplay	displayGameText(string): void	Outputs text to terminal

UML diagram


Environment
#description: string
#connectedEnvironment: vector<Environment*>
#name: string
+enter(): virtual void
+~Environment(): virtual
Safe Environment
Puzzle Environment
+interactWithNpc(npc:NPC*): void
Dangerous Environment
+killCharacter(): void
NPC
#name: string
#description: string
+givePuzzle(): virtual void
+dialogue(): virtual void
+~NPC(): virtual
Librarian
+givePuzzle(): void
+dialogue(): void
Guard
+givePuzzle(): void
+dialogue(): void
Dementors
+givePuzzle(): void
+dialogue(): void
Princess
-inventory: Inventory*
+givePuzzle(): void
+dialogue(): void
-wakeUp(): void
Witch
-inventory: Inventory*
+givePuzzle(): void
+dialgoue(): void
-givePotion(): void
Character
-inventory: Inventory*
-dead: bool
+Character()
+interactWithNpc(npc:NPC*): void
+solveRiddle(answer:string): void
+useItem(item:Item*): void
+isDead(): bool
User Input
-currentInput: string
-gameDisplay: GameDisplay*
+getInput(): string
+getDialogueResponse(npc:NPC*): void
+getPuzzleResponse(npc:NPC*): void
+getItemResponse(Item:Item*): void
Game
-character: Character*
-currentEnvironment: Environment*
+startGame(): void
+endGame(): void
+moveToEnvironment(newEnvironment:Environment*): void
Game Display
-character: Character*
-currentRoomDescription: string
+displayGameText(text:string): void
Inventory
-items: map<string, item*>
+addItem(item:Item*): void
+removeItem(item:Item*): void
+hasItem(item:Item*): void
Item
#name: string
#description: string
+useItem(): virtual void
+~Item(): virtual
Bad Potion
+useItem(): void
Good potion
+useItem(): void
Flaming Sword
-isLit: bool
-isLit(): bool
+useItem(): void