Artifact under test: Item class

Artifact specifications: contains information about an item

Control inputs and conditions: When creating item, must have an ID, name, and an image

Output results and conditions: Able to get item ID, name, or image

Spec ID	**Inputs/Conditions**	**Success Criteria**
itemClass_01	Class creates an item with information specified by a different class	Item is created with information specified by a different class
itemClass_02	Program needs the item image and calls getImage method	Class returns the correct image to represent the item
itemClass_03	When Player uses/collects a specific item	Class should increment or decrement a value that represents amount in possession

Artifact under test: PlayerBag.addItem(itemID) (same tests were done for removing an item)

Artifact specifications: adds an item to the inventory

Control inputs and conditions: itemID

Output results and conditions: Item is now in inventory

Spec ID	**Inputs/Conditions**	**Success Criteria**
addItem_01	If bag is full	Does not add item
addItem_02	itemID does not exist	Does not add item
addItem_03	itemID results in out of array bounds	Does not add item
addItem_04	Correct itemID	Item with specific itemID is added to inventory

Artifact under test: InventoryUI.addItem(itemID)

Artifact specifications: Uses PlayerBag class to add item to inventory

Control inputs and conditions: itemID

Output results and conditions: Shows image of item in inventory to the player

Spec ID	**Inputs/Conditions**	**Success Criteria**
InventoryUI_display_01	If bag is full	Display a message saying the bag is full
InventoryUI_display_02	Player collects item	Creates a button with image of item in the player’s inventory

Artifact under test: InventoryUI.getClickedButton(itemID, itemButton)

Artifact specifications: Gets which button got pressed in inventory

Control inputs and conditions: itemID, itemButton

Output results and conditions: Drops or uses item associated with the button pressed

Spec ID	**Inputs/Conditions**	**Success Criteria**
InventoryUI_button_01	Pressing a button representing an item	Item is removed from inventory and the button is destroyed.
InventoryUI_button_02	Pressing a button of item that can be dropped	Item is created in front of the player with correct information i.e. correct image
InventoryUI_button_03	The item is the water bottle and player’s health is full	Do nothing
InventoryUI_button_04	The item is the water bottle and player’s health is not full	Player health is set to full

Artifact under test: PlayerHealth Script

Artifact specifications: Keeps track of player health

Control inputs and conditions: Needs to be attached to player object

Output results and conditions: Shows player current health as an image

Spec ID	**Inputs/Conditions**	**Success Criteria**
PlayerHealth_01	Player starts first stage	An image appears on screen showing player’s health
PlayerHealth_02	Player gets hit by an enemy	Player loses health and is reflected in the image
PlayerHealth_03	After player is hit	Player should flash for a period of time where player can’t take anymore damage
PlayerHealth_04	When player uses a water bottle	Player gains health and is reflected in the image
PlayerHealth_05	When player has no more health	Game should show next scene where a game over message is shown

Artifact under test: GameOver Scene

Artifact specifications: Is shown when player loses

Control inputs and conditions: Player has no more health

Output results and conditions: Player is given a couple of options to choose from

Spec ID	**Inputs/Conditions**	**Success Criteria**
GameOver_01	Player presses “Try Again” button	Game restarts at the beginning
GameOver_02	Player presses “Quit”	Game exits

Artifact under test: Enemy Movement

Artifact specifications: Enemy State Behaviour

Control inputs and conditions: Player is nearby or not in range

Output results and conditions: Enemy chases the player or patrols

Spec ID	**Inputs/Conditions**	**Success Criteria**
Enemy_01	Adding sprites and animations	Enemy should visibly have correct sprite and animation when walking in certain directions
Enemy_02	Enemy is in patrol state	Enemy walks in a predetermined pattern
Enemy_03	Player is in range of enemy	Enemy is in Chase state
Enemy_04	Enemy is in chase state	Enemy moves to player’s last position
Enemy_05	Player moves out of range and enemy was in chase state	Enemy is in patrol state