**Name:** Application

**Description:** Runs a simple side scroller game.

**Requirements:**

The program should run a simple side scrolling game that features a player, non-moving objects and moving objects. The game should support save, pause and exit operations. Some statistics must be tracked.

**Classes:**

1. Application
   * Instance variables:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Purpose** |
| in | BufferedReader | Reads console input |
| game | Game | Holds current game state |
| frame | JFrame | Frame to display the game |

* + Methods:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Parameters** | **Return** | **Description** | **Pre/post-conditions** | **Interaction** | **I/O** |
| main | String[] arg | void | The main function of the application. | None. | Creates a Game object and invokes its play() method. | Sends prompts to user via System.out, reads options via System.in |

1. Game extends JPanel
   * Instance variables:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Purpose** |
| p | Player | The player’s character. |
| obstacle | Obstacle[][] | holds the obstacles in the world |
| life | Life[][] | holds the lives in the world |
| missile | Missile[][] | holds the missiles in the world |
| sentry | Sentry[][] | Holds the sentries in the world |
| lightning | Lightning[][] | holds the lightning bolts in the world |
| map | String[] | Holds a rendering of the area currently visible to the player. |
| score | int | Holds the players current score. |
| t | Timer | Coordinates actions by firing at regular intervals. |

* + Methods:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Parameters** | **Return** | **Description** | **Pre/post-conditions** | **Interaction** | **I/O** |
| play | BufferedReader in | boolean | This function handles the operations performed on r rightevery turn. | A valid game object must be created. | Uses the methods provided by the Player, Obstacle and Missile objects to manipulate them. | Accepts game controls from the console. Prints the map, score, and instructions to console. |
| render() | -/- | boolean | Renders the visible area using the locations stored in the game objects. | The game must be initialized. After execution, the map[][] field holds a representation of the map. | None | None. |
| printMap | -/- | void | Prints the map. | Map must be refreshed. | None | Prints the map to console. |
| actionPerformed() | ActionEvent ae | void | Advances the game by one cycle every time an action event is called. | t must generate an ActionEvent | The objects should advance by one cycle. | None |

1. Game object classes:
   * Instance variables:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Purpose** |
| sym | String | Holds the symbol of the object |
| r, c | int | Holds coordinates of the object |

* + Methods:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Parameters** | **Return** | **Description** | **Pre/post-conditions** | **Interaction** | **I/O** |
| move() | -/- | void | Moves the object | None | Changes the coordinates of the object | None |

**Game manual:**

W moves the player up

S moves the player down

D moves the player right

Lives add a life.

Colliding into any other object removes a life.

The game ends when all lives are over.