Class Launcher

Java.lang.Object

The launcher class is where the main method sits to initialize the game. First, the dungeon is created and is called dungeon1. The user inputs either 1 or 2 to move the hero across the dungeon. The game will keep running until the method of gameEnd() returns a 1, which will then terminate the game.

Fields:

Methods:

Class Beings

Java.lang.Object

The beings class is an abstract class that houses the health of all beings in the dungeon.

Fields:

public int health;

This field initializes the health system which allows the hero and monster to have health stored/accessed/modified.

Methods:

Class Monster

Java.lang.Being

Java.lang.Monster

This class is where the monster's health and attack are modified. Accessing the methods listed below will adjust the health of the monster accordingly.

Fields:

public int health = 10;

This field sets the monster's health to 10.

public int attack = 2;

This field sets the monster's attack to 2.

Methods:

public int getHealth()

This method returns the health of the monster

public int adjustHealth(Hero h)

This method takes in the hero object as a parameter to be able to adjust the monster's health. Then the method takes the current health of the monster subtracted by the attack of the hero and then returns the new health

Class Hero

Java.lang.Being

Java.lang.Hero

This class is where the hero's health and attack are modified. Accessing the methods listed below will adjust the health of the monster accordingly.

Fields:

public int health = 10;

This field sets the monster's health to 10.

public int attack = 5;

This field sets the monster's attack to 5.

Methods:

public int getHealth()

This method returns the health of the monster

public int adjustHealth(Monster m)

This method takes in the monster object as a parameter to be able to adjust the hero's health. Then the method takes the current health of the hero subtracted by the attack of the monster and then returns the new health

Class Dungeon

Java.lang.Dungeon

The dungeon class is where most of the work is done in the game. The hero and monster are both created and placed inside of the dungeon. The

Fields:

private int hLocation = 0;

This value is used to keep track of the hero through the indexes of the array

private int hLocationRemover = hLocation;

This value is used to remove the hero from the previous index after the hero moved

private int mLocation = 9;

This value is used to keep track of the monster through the indexes of the array

private int mLocationRemover = mLocation;

This value is used to remove the monster from the previous index after the monster moved.

boolean mDead = false;

This Boolean is used to check for if the monster is dead or alive

boolean hDead = false;

This Boolean is used to check for if the hero is dead or alive

Methods:

public void createDungeon()

This method places the hero at the beginning of the array (index 0) and the monster at the end of the array (Index 9)

public void dungeonToString(Dungeon d)

This method runs through the object array after turn and will display the new positions of the objects inside of it. A ‘H’ represents the hero, a ‘M’ represents a monster and ‘-‘ represents empty spots in the dungeon.

public void moveBeings(int input)

This method takes in the user input (either a 1 or 2) and makes several calls to other methods to move the beings in the array. If 1 is inputted and the hero is not at the index of 0, both the hero and monster are moved to the left. If the hero is at the 0 index, only the monster moves left. If 2 is in input, the hero will move right while the monster moves left. Once the hero and monster are next to each other, the adjustHealth() methods are called and the beings continue to fight it out until one of their health are 0. After each “battle” both the health of the monster and hero will be outputted. If either the hero or monster health hits 0, mDead or hDead will turn true which means they die.

public void moveHeroRight()

This method moves the hero right one index (n+1) and then removes the hero from the previous index(n).

public void moveHeroLeft()

This method moves the hero left once index and removes the hero from the previous index.

public void moveMonsterLeft()

This method moves the monster to the left and then removes the monster from its previous index

public int gameEnd()

This method checks if the hero hits the end of the dungeon (index 9) or if the monster hits the end of the dungeon (index 0). If either of those conditions are met, the method returns a 1 which will force the game to end. If neither of those conditions are met, the method returns a 2 and the game continues.