You have decided to write your own game as a basic console application prototype to demonstrate how two types of game characters would move. Every game character on screen will be represented using a single-character string, e.g. #.

Your play area should use an x, y coordinate system. 0, 0 represents the top-left corner of your play area (console window). Your chosen maximum height and width of the play area will then represent the bottom-right corner. The y's will be individual lines on the console window and the x's will be individual character spaces in a specific line. Remember to limit your play area to a size that will fit on the console window and still have space for some text below the play area (see the rest of the specifications for what the text may be).

To serve as a basis for all types of game characters, you have decided to create a basic class called GameCharacter. This class should contain the following functionality:

- A public const int variable to contain the maximum width of the play area (you may choose this value yourself).
- A public const int variable to contain the maximum height of the play area (you may choose this value yourself).
- An int field to store the current X-coordinate of the GameCharacter. This value should be available to any derived classes, but not to any outside classes or programs.
- An int field to store the current Y-coordinate of the GameCharacter. This value should be available to any derived classes, but not to any outside classes or programs.
- A string field to store the symbol used to display the GameCharacter. This value should not be available to any derived classes, outside classes or programs. This can be any single character, e.g. #.
- A constructor, which receives the GameCharacter's initial coordinates and display symbol.
- A method Move, which is to be overloaded by derived classes. This method should receive no parameters and return no values.
- A method Draw, which receives two integer values, representing a set of coordinates. If the current object's coordinates match the received coordinates, draw the GameCharacter's symbol to screen, using Console.Write(). This method should return true if the coordinates match and false if it does not.
- An overload of the ToString() method which returns the GameCharacter's display symbol, its x and y coordinates and the type of the current object, using the object class' GetType method. The details should be returned in a string, similar to the one shown below:

  # 10, 10 object type name

Create a class called HorizontalMover, which inherits from class GameCharacter. This class should contain the following functionality:

- A private bool field, MoveRight, which is set to true if the HorizontalMover is moving to the right.
- A constructor, which receives two integer variables, to serve as its initial x and y coordinates, and a symbol to display itself.
- An overload of the base class' Move method. This method must move the character either to the left or the right, depending on the class' MoveRight field. The character should move in one direction until it reaches the limit of the play area.

Create a class called VerticalMover, which inherits from class GameCharacter. This class should contain the following functionality:

- A private bool field, MoveDown, which is set to true if the VerticallMover is moving to the bottom.
- A constructor, which receives two integer variables, to serve as its initial x and y coordinates, and a symbol to display itself.
- An overload of the base class' Move method. This method must move the character either to the top or bottom, depending on the class' MoveDown field. The character should move in one direction until it reaches the limit of the play area.

In the program's main method:

- Create an array of 4 GameCharacters, containing two HorizontalMovers and two VerticalMovers.
- The game prototype should consist of a loop, which will continue until the user enters 'e'.
- In each iteration of the game loop, do the following:
    - Clear the console window.
    - Draw every GameCharacter at its correct position in the play area.
    - Display the symbol, coordinates and object type of each GameCharacter below the play area.
    - Move every GameCharacter. This move will only reflect (i.e. be Drawn) on the next iteration of the loop.
    - Gather user input