Escape floor

Create a C# Console Application Game. The Game is simple: Your character is situated at the very bottom of the Console. Floors start coming from the bottom and your character has to escape through the holes from each floor.

The Character

1. The character should look like this:

                \_O\_

                   |

                  / \

1. The character moves left or right and he tries to escape through a hole in the floor.
2. The character starts at the very bottom of the page and moves up if he is on a floor.
3. If he goes through the hole he moves down until he reaches another floor.
4. He has to move faster than the floors.
5. The Game is Over when the character hits the top wall.

Floors

1. Each floor is a single line comprised of “\*” with a hole in the middle. Something like this:

           “\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*          \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*”

1. Each floor is randomly generated (meaning that each hole is at a random position).
2. Floors move up and the character stays on a floor trying to escape through the hole.
3. They appear from the very bottom of the Console and move up until they reach the top. After they reach the top, they are destroyed.

Difficulties

There should be three difficulties: easy, medium and hard.

1. Easy - slow floors, large holes.
2. Medium - faster floors, smaller holes
3. Hard - really fast floors, really small holes.

Points

Each floor passed generates 20 points. Each loop step should generate +1 point.

Power-Ups and Power-Downs

You should implement three power-ups/power-downs. They are generated at a random position on some floors.

1. Red “@”. Increases the speed of the floors for 7 seconds
2. Yellow “&” Decreases the speed of the floors for 5 seconds.
3. Green “$”. Removes the next **3-6(depending on balance)** floors from spawning. Your character should fall through the hole and reach either the bottom of the Console or the next available floor.

1.)  should be spawn relatively often (e.g. 10% of the floors)

2.)  should be slightly rarer. (5%)

3.) should be very rare. (3%)

Balance the game to your liking.

Evaluation:

1. Player drawing and player movement and falling - 20 points
2. Floors generation, drawing, moving up and destroying - 20 points
3. Difficulties - 20 points
4. UI - points drawing and correctly evaluated, difficulty drawing - 10 points
5. PowerUps - 20 points
6. Balancing - 10 points
7. Additional functionality bonus - 20 points. If you have enough time, you can add additional functionality to the game to win bonus points. You should not break the current mechanics of the game (e.g. making the player shoot would make no sense). Some examples: add an Intro State, Main Menu State, Game Over Screen, Replay, add more power-ups. If you add any functionality, be sure to explain it at the top of the .cs file as a comment.

Deductions:

There will be deductions for bad quality of the code! You need to name your variables properly, use methods and so on. Program.cs -> -30points.

**Make sure you set the console width and height!!!**

