using System;

using System.Collections.Generic;

using System.Text;

public class Player

{

private string name;

private int endurance;

private int sprint;

private int dribble;

private int passing;

private int shooting;

public Player(string name, int endurance, int sprint, int dribble, int passing, int shooting)

{

this.Name = name;

this.Endurance = endurance;

this.Sprint = sprint;

this.Dribble = dribble;

this.Passing = passing;

this.Shooting = shooting;

}

internal string Name

{

get

{

return this.name;

}

private set

{

if (String.IsNullOrEmpty(value) || String.IsNullOrWhiteSpace(value))

{

throw new ArgumentException("A name should not be empty.");

}

this.name = value;

}

}

private int Endurance

{

get

{

return this.endurance;

}

set

{

if (value < 0 || value > 100)

{

throw new ArgumentException($"Endurance should be between 0 and 100.");

}

this.endurance = value;

}

}

private int Sprint

{

get

{

return this.sprint;

}

set

{

if (value < 0 || value > 100)

{

throw new ArgumentException($"Sprint should be between 0 and 100.");

}

this.sprint = value;

}

}

private int Dribble

{

get

{

return this.dribble;

}

set

{

if (value < 0 || value > 100)

{

throw new ArgumentException($"Dribble should be between 0 and 100.");

}

this.dribble = value;

}

}

private int Passing

{

get

{

return this.passing;

}

set

{

if (value < 0 || value > 100)

{

throw new ArgumentException($"Passing should be between 0 and 100.");

}

this.passing = value;

}

}

private int Shooting

{

get

{

return this.shooting;

}

set

{

if (value < 0 || value > 100)

{

throw new ArgumentException($"Shooting should be between 0 and 100.");

}

this.shooting = value;

}

}

internal int Stats

{

get

{

return (int)Math.Round((this.Endurance + this.Dribble + this.Sprint + this.Passing + this.Shooting) / 5.0);

}

}

}