MATCOM

Grupo Weboo
Universidad de La Habana

Primer Examen de Programación

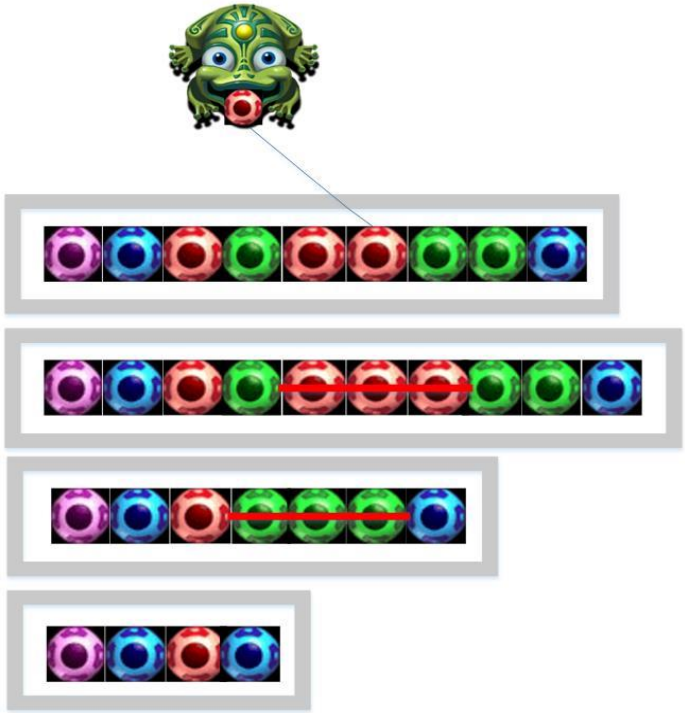
Curso 2019-2020

Zuma

NOTA: Si usted está leyendo este documento sin haber extraído el compactado que se le entregó, ciérrelo ahora, extraiga todos los archivos en el escritorio, y siga trabajando desde ahí. Es un error común trabajar en la solución dentro del compactado, lo cual provoca que los cambios no se guarden. Si usted comete este error y entrega una solución vacía, no tendrá oportunidad de reclamar.

Fito es un gran fan de los juegos de puzzle, y su preferido entre todos es Zuma. El juego consiste en que el jugador debe formar grupos de 3 o más bolas del mismo color alrededor de una pista antes de que las bolas lleguen a la Calavera. Cada vez que se forman grupos del mismo color estos se rompen y desaparecen. Para formar los grupos está la Rana, que lanza las bolas que ruedan por la pista.

Por ejemplo, en la figura, si la Rana lanza la bola roja en la posición 5, hace que se forme un grupo de 3 bolas rojas, por tanto, estas explotan y desaparecen. Siempre que quede un espacio vacío entre 2 líneas de bolas, estas se unen. En caso en el que vuelva a quedar algún grupo de colores iguales, explotan instantáneamente, lo que pasa en este caso con las bolas verdes.



Fito está teniendo problemas en un nivel particularmente difícil y necesita de tu ayuda. En este nivel, un poderoso guerrero malvado llamado Etzatlan le robó el control de la Rana, y va a disparar las **n** primeras bolas por él. Para decidir una estrategia ganadora, Fito necesita predecir el estado final de las bolas luego de los movimientos del guerrero.

Fito necesita tu ayuda para implemetar un programa que determine el estado en el que quedará la pista luego de los lanzamientos de Etzatlan.

Usted debe haber recibido junto a este documento una solución de Visual Studio con dos proyectos: una biblioteca de clases (Class Library) y una aplicación de consola (Console Application). Usted debe implementar el método Simula que se encuentra en la clase Zuma en el namespace Weboo.Examen. En la biblioteca de clases encontrará la siguiente definición:

```
namespace Weboo.Examen
{
    public class Zuma
    {
        public static int[] Simula(int[] colores, int[] posiciones, int[] pista)
        {
            //Borre la siguiente línea y escriba su código
            throw new NotImplementedException();
        }
    }
}
```

Los parámetros **colores** y **posiciones** son arrays de igual tamaño que representan los colores y las posiciones donde se lanzarán las bolas respectivamente. El parámetro **pista** representa los colores de las bolas en el estado inicial del juego. Usted debe implementar este método de tal manera que devuelva un array de enteros representando el estado final.

Aclaraciones:

- Si algún valor en **posiciones** es negativo, la bola se lanza a la posición más a la izquierda, y si es mayor que el tamaño del array, se lanza a la posición más a la derecha del mismo.
- Arrojar una bola en la posición **i** quiere decir que esta cae en la posición **i** y las demás bolas (incluida la i-ésima) se desplazan a la derecha.
- Ninguno de los arrays que se pasan como parámetro serán `null`.

NOTA: Todo el código de la solución debe estar en este proyecto (biblioteca de clases), pues es el único código que será evaluado. Usted puede adicionar todo el código que considere necesario, pero no puede cambiar los nombres del namespace, clase o método mostrados. De lo contrario, el probador automático fallará. En particular, es imprescindible que usted no cambie los nombres, tipo u orden de los parámetros del método **Simula**. Por supuesto, usted puede (y debe) adicionar todo el código que necesite.

Ejemplos:

```
//EJEMPLO 1
int[] col1 = {1,1,2,1};
int[] pos1 = {3,0,2,0};
int[] pista1 = {};
Zuma.Simula(col1, pos1, pista1);
//Respuesta correcta: {2}


//EJEMPLO 2
int[] col2 = {3};
```

```
int[] pos2 = {2};
int[] pista2 = {1,2,3,3,2,2,1,1};
Zuma.Simula (col2, pos2, pista2);
//Respuesta correcta: {}


//EJEMPLO 3
int[] col3 = {};
int[] pos3 = {};
int[] pista3 = {1,2,1,3,4};
Zuma.Simula (col3, pos3, pista3);
//Respuesta correcta: {1,2,1,3,4}


//EJEMPLO 4
int[] col4 = {4};
int[] pos4 = {2};
int[] pista4 = {1,4,4,3,3,3};
Zuma.Simula (col4, pos4, pista4);
//Respuesta correcta: {1,3,3,3}
```

NOTA: Los casos de prueba que aparecen en este proyecto son solamente de ejemplo. Que usted obtenga resultados correctos con estos casos no es garantía de que su solución sea correcta y de buenos resultados con otros ejemplos. De modo que usted debe probar con todos los casos que considere convenientes para comprobar la validez de su implementación.