

**Algoritmos y Estructura de Datos**

**Parcial 2**

**Búsquedas y Ordenamientos**

La empresa Mattel les ha pedido que creen una versión digital del sistema para calcular el puntaje de su juego UNO-Simple. En este juego cuenta con cartas de cuatro colores y solo numéricas (0-9). Los puntos de un jugador se calcular con base en el valor de número, y la carta 0, que se considera una carta castigo y vale 20 puntos.

El juego debe tener entre 3 y 6 jugadores (se debe definir al instanciar la clase principal del juego. En lo posible, la estructura que almacena los jugadores debería ser dinámica). Cada jugador tiene una estructura de datos en donde almacena las cartas que tiene (alguna estructura lineal vista en clase).

El sistema reparte las cartas en una simulación de partida final a cada y uno de los jugadores. Uno de los jugadores tiene 0 cartas, y el resto de las cartas deben repartirse, de manera aleatoria entre los jugadores.

Usted debe:

* Ejecutar sin errores y con funcionalidades mínimas (1.5)
* Completar la estructura de las clases y los métodos básicos. (0.5)
* Leer el número de jugadores (mínimo 3, máximo 6) y crear los jugadores. (0.25)
* Calcular el total de puntos de cada uno de los jugadores (1.0)
* Escribir por pantalla la lista de jugadores con sus cartas (0.25)
* Escribir por pantalla la lista de jugadores con sus puntajes (0.25)
* Ordenar los jugadores, de acuerdo con el puntaje, usando mecanismos dejados de tarea: Radix, Bucket o Merge Sort (0.5)
* Ordenar las tarjetas de cada jugador, primero por número y luego por color, usando mecanismos dejados de tarea: Radix, Bucket o Merge Sort (1.0)
* Escribir por pantalla la lista ordenada

Deben enviar el código fuente en C++ a la actividad en e-aulas, en el tiempo designado para el parcial.

**Salida del programa**

Ingrese el numero de jugadores:

7

Ingrese el numero de jugadores:

1

Ingrese el numero de jugadores:

4

0 -> (1,6) , (1,9) , (2,5) , (2,6) , (3,0) , (3,2) , (3,4) , (3,5) ,

1 -> (0,1) , (0,2) , (0,3) , (0,5) , (0,9) , (1,0) , (2,1) ,

2 -> La lista secundaria está vacía

3 -> (0,0) , (0,4) , (0,6) , (0,7) , (1,4) , (1,5) , (1,8) , (2,0) , (2,2) , (2,8) , (3,3) , (3,6) , (3,8) , (3,9) ,

Jugador 0 tiene 57 puntos

Jugador 1 tiene 41 puntos

Jugador 2 tiene 0 puntos

Jugador 3 tiene 110 puntos

Jugador 2 tiene 0 puntos

Jugador 1 tiene 41 puntos

Jugador 0 tiene 57 puntos

Jugador 3 tiene 110 puntos

0 -> (3,0) , (3,2) , (3,4) , (3,5) , (2,5) , (2,6) , (1,6) , (1,9) ,

1 -> (1,0) , (2,1) , (0,1) , (0,2) , (0,3) , (0,5) , (0,9) ,

2 -> La lista secundaria está vacía

3 -> (2,0) , (0,0) , (2,2) , (3,3) , (1,4) , (0,4) , (1,5) , (3,6) , (0,6) , (0,7) , (3,8) , (2,8) , (1,8) , (3,9) ,

\*\* Process exited - Return Code: 0 \*\*

**Función Main**

int main()

{

srand(1234);

int num\_jugadores = 0;

int num\_colores = 4;

int num\_tarjetas = 10;

//Lea el número de jugadores, hasta que cumpla con la condición

int winner = rand()%num\_jugadores;

Uno u = Uno(num\_jugadores);

//Añada num\_jugadores jugadores a la lista

int temp=0;

for(int i = 0; i<num\_colores; i++){

for(int j = 0; j< num\_tarjetas; j++){

temp = rand()%num\_jugadores;

if(temp != winner){

u.addTarjeta(temp, new Tarjeta(i,j));

}

}

}

u.print();

//Muestre el puntaje de los jugadores, en orden de id

u.sort(); //Ordena a jugadores y sus cartas internamente

cout<<endl;

u.print();

return 0;

}