Carrera Ingeniería y Licenciatura en Sistemas

Parcial 2 matutino de: Estructuras de datos y Algoritmos 2 **Código de materia: 1778**

Fecha: 9-12-21 **Hoja 1** de **2**

NOTAS IMPORTANTES
Duración: **2 horas**
Con material: **No**

Serán factores relevantes en la valoración de las soluciones desarrolladas, la claridad en la presentación, el estilo y la metodología de trabajo, acordes a los impartidos en el curso. Recordar la utilización de nombres autoexplicativos para funciones, variables, constantes, etc.

Ejercicio 1.

Ud. tiene un amigo que es dueño de una pastelería, ahora que se acercan las vacaciones, quiere hacer cupcakes y, para tentar a las familias que pasan por la vitrina, se le ocurrió ponerles frosting de distintos colores. **Con la condición de que dos cupcakes vecinos no pueden tener el mismo frosting.**

Cuando se puso a ver cuantos frostings tenía que hacer se empezó a marear, entonces le pidió a ud que le resuelva el problema, porque sabe que le gusta resolver este tipo de problemas. Le dice que no le molesta hacer frostings de más, ya que el costo de hacer varios colores es el mismo, pero tampoco quiere tener todos los cupcakes de colores distintos. Es decir, **quiere hacer pocos frostings, pero no necesariamente el mínimo**.

Se pide:
Hacer **un algoritmo greedy** que reciba los cupcakes en forma de grafo e imprima **cuántos colores** se utilizaron.

Firma de la función:

```
int colorear(Grafo g, int C)
```

Notas:

- *Los cupcakes se representan en el grafo con números del 1 al C.*
- *No debe implementar ningún método de grafo.*

Ejercicio 2. (12 puntos)

Considere un tablero de ajedrez de N*N, donde hay 1 caballo blanco, una cantidad X de peones blancos y una cantidad Y de peones negros.

Se le pide que indique una secuencia de la menor cantidad de pasos posibles, tal que el caballo coma a todos los peones negros, con la restriccion de que no puede pisar los lugares ocupados por peones blancos.

Recibira al tablero como una matriz de caracteres, donde ‘B’ es un peon blanco, ‘N’ es un peon negro, y ‘V’ es un espacio vacio.

Implemente con backtracking la función:

```
List<Pos>* comerPeones(int N, char** tablero, int Y, Pos posCaballo)

struct Pos {
  int x;
  int y;
};
```

Notas:

- *Si utiliza funciones auxiliares, deberá implementarlas completamente.*
- *No debe implementar ni especificar el TAD List.*