I200 - Algoritmos y Estructuras de Datos
2do Parcial.

```
Conversion de
Notas:
00-59 = INSUF
60-69 = 6
70-79 = 7
80-89 = 8
90-94 = 9
95-99 = 10
```

Nombre:	
Legajo:	

Instrucciones: Los ejercicios deberían poder completarse en el enunciado mismo, en caso de quedarse sin lugar puede usar alguna hoja auxiliar o el dorso.

Rúbrica:

Ej	1	2	3	4	5	6	7	8	9	10	Total
Pts	10	10	10	10	10	10	10	10	10	10	100

[1] Indicar V o F en cada una de las siguientes afirmaciones justificando brevemente su respuesta (por ejemplo indicando un contraejemplo):

a) La cantidad de Bits prendidos en todos los bitmaps de un **Hopscotch** hashing indica la cantidad de elementos existentes. _____
b) El **boosting** es un proceso basado en iteraciones que permite eliminar la sobreestimación de un Count-Min filter. _____
c) **DFS** es apropiado para encontrar Caminos mínimos en un **grafo con ciclos**. _____
d) Los resultados obtenidos luego de aplicar |V|+1 iteraciones de **Bellman-Ford** en un grafo con **ciclos negativos** son correctos y, además, se sabe que el grafo tiene este tipo de ciclos _____

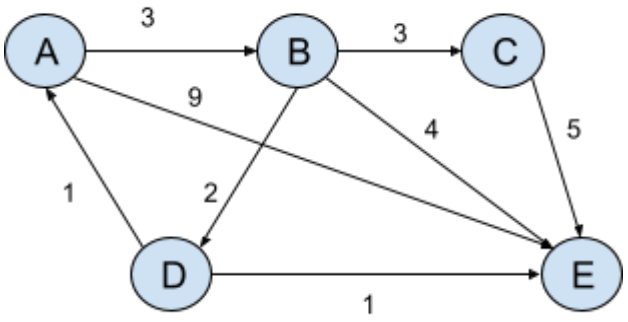
[2] Realizar las siguientes operaciones sobre la tabla de hashing usando el método del **hopscotch** con vecindario de 3 (**k=3**). Trate de indicar las operaciones sobre las mismas tablas tachando las claves que son reemplazadas.

	A	B	C	D	E	F
h()=	205	571	897	943	888	33

	Bitmap	Valor
0		
1		
2		
3		
4		

Operaciones: Insert(A), Insert(D), Insert(E), Insert(B), insert(F), Delete(A), Insert(C)

[3] Dado el siguiente grafo:



a) Calcular todos los caminos mínimos desde el nodo A usando el algoritmo de Dijkstra (construya una amplia tabla de trabajo e indique los cambios tachando los valores anteriores) .

Nodo	Visitado	Distancia	Desde
A			
B			
C			
D			
E			

b) Complete la tabla de Resultados:

Camino Mínimo desde A hasta…	Distancia	Camino
B		
C		
D		
E		

[4] Dado un **grafo no dirigido y no pesado** se pide programar la función ImprimirEquidistante(g,v1,v2,d) que imprima los nodos que se encuentran **a d aristas de v1 y a d aristas de v2**. Indique la **complejidad** del algoritmo que programó.

[5] Considere la serie matemática AnGi definida de la siguiente forma:

$$AnGi = \begin{cases} 1.5 & \text{para todo } i<3 \\ 2.5 & \text{para } i = 3 \\ 3 & \text{para } i=4 \\ 3.7 & \text{para } i=5 \\ AnGi[i-3] + 2*AnGi[i-2] - 5 * AnGi[i-5] & \text{para } i>5 \end{cases}$$

a) Programe una **función recursiva** que permita calcular cualquier término de la serie.

b) Sobre la base del algoritmo diseñado en (a) construya una solución de **programación Dinámica** aplicando la técnica de **Memoization**. Para esto suponga la existencia de una Tabla de Hashing ya creada y que no debe destruir, con la siguiente API:

```
void InsertElement(unsigned Key, float Value);
float GetValue(unsigned Key);
bool Exists(unsigned Key);
```

c) Construya una solución de programación Dinámica aplicando la técnica de Tabulation. Para esto suponga la existencia de un arreglo de máximo 6 elementos:

[6] Dado el siguiente vector:

12	67	32	11	72	38	25	1	6	33	16	5	36	20	9	4
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Usando bloques con **b=4** y **tablas dispersas** tanto para el **superbloque** como para los **bloques** individuales le pedimos que muestre la tabla del superbloque y las tablas individuales de cada bloque, estas últimas tiene que construirlas usando el **método de los cuatro rusos**. Una vez construidas las estructuras indique cómo resuelve las consultas RMQ(0,7). RMQ(6,14)

[7] Dado el siguiente bloque de 5 números enteros

10	65	42	9	30

a) Indicar el **mapa de Bits** correspondiente al Bloque

	10	65	42	9	30	
Bits						
Pila						

b) Encontrar un bloque **distinto**, **isomórfico** y armado con los **mismos números**.

Bits						
Pila						

c) Encontrar un bloque con los **mismo números** y que **no sea isomórfico**

Bits						
Pila						

[8] Para el diseño del firmware de un router se desea utilizar un **Filtro de Bloom** para poder determinar las IPs que han originado conexiones entrantes, con una **probabilidad de falso positivo máximo de 4 en 10.000** considerando que se registrarán **hasta 2.000.000** de conexiones entrantes.

a) ¿Cuántas **funciones de Hashing** deberían utilizarse?

b) ¿Cuántos **bits** deberían utilizarse en el filtro?

[9] Dibuje un **Grafo** que cumpla las siguientes condiciones:

- El grafo debe ser **conexo**, **dirigido** y **no-pesado**.
- El grafo no puede tener **loops**.
- El grafo debe tener **2 ciclos**.
- El grafo debe tener al menos **6 aristas**.
- El **diámetro** del grafo tiene que ser **4**.

Finalmente aplique DFS sobre el grafo construido indicando en qué orden se visitarán los nodos.

[10] A los efectos de contabilizar la cantidad de elementos distintos registrados en un stream de datos se utilizó un Filtro tipo Flajolet-Martin pero utilizando 10 funciones de hashing. Luego de unos minutos de funcionamiento los respectivos contadores eran:

L0	L1	L2	L3	L4	L5	L6	L7	L8	L9
7	11	60	58	20	14	46	23	11	36

Estime la cantidad de elementos utilizando las técnicas/métodos indicados:

Método	Estimación de Cantidad
Log-Log	
Super Log-Log	
Hiper Log-Log	