NOMBRE:_____APELLIDOS:_____

CUESTIONES (1:00 h) – No se puede utilizar ningún tipo de apuntes o libros

1) Considera las siguientes recurrencias y clasificalas según sean: No Lineales (NL), Lineales (L) o Lineales con coeficientes constantes (LCC) e indica si són Homogeneas (H) o No Homogeneas (NH). Indica de que orden son cuando sea posible. (5 ptos)

Recurrencia	Lineal	Homogenea	Orden
1) $T(n) = 2T(n-1) + n$			
2) $T(n) = \sqrt{T(n-1)}$			
3) $-T(n) = \dfrac{T(n+1) - 2T(n-1)}{2}$			
4) $T(n) = (n-1)T\left(\sqrt{n-2}\right) + 1/n$			
5) $2 = \sqrt{\dfrac{T(n)}{T(n-1)}}$			

2) Considera los siguientes algoritmos de ordenación: Insercción, Selección, Burbuja (o intercambio), Quicksort, Mergesort y Heapsort y responde a las siguientes preguntas (5 ptos)

a) ¿Qué algoritmos van generando un subvector correctamente ordenado (donde los elementos ya se encuentran en su posición final)? (1 pto)

b) ¿Qué algoritmos se basan en la selección del mínimo? (1 pto)

c) ¿Qué algoritmo de ordenación directa es <u>significativamente mejor</u> en comparaciones sobre el vector si el vector está inversamente ordenado? Justifica tu respuesta. (1 pto)

d) Considera el siguiente vector [4; 8; 9; 5; 6; 1; 7; 2]. Escribe como van ordenándose los elementos del vector si para ello se utiliza el algoritmo Mergesort. ¿Cuántas veces se llama al procedimiento de que mezcla dos subvectores?. Escribe en el siguiente cuadro como queda el vector después de cada llamada al procedimiento de mezclar (y sólo después de llamar a dicho procedimiento) (2 ptos)

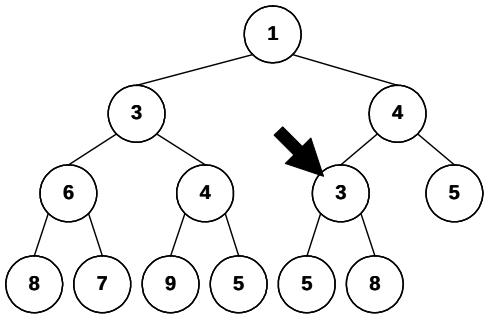
Nº llamadas:

INI	**4**	**8**	**9**	**5**	**6**	**1**	**7**	**2**
1								
2								
3								
4								
5								
6								
7								
8								
9								

11								
12								
13								
14								
15								
16								
17								
18								
19								

10									20								

3) Considera el siguiente árbol y contesta a las siguientes preguntas (5 ptos)



a) ¿Se trata de un montículo?. Justifica tu respuesta. (1 pto)

b) ¿Qué valores desde 1 hasta 100 puede tomar el nodo marcado con una flecha para que el árbol sea un montículo? (1 pto)

c) Dado un vector de n elementos no repetidos que representa un montículo. ¿Cuántas comparaciones son necesarias para determinar si se trata de un montículo de mínimo o de máximo? (1 pto)

d) Y si el vector puede tener elementos repetidos, ¿Cuántas comparaciones pueden llegar a ser necesarias para determinar si se trata de un montículo de mínimo o de máximo? (1 pto)

e) ¿Qué método es mejor para implementar un algoritmo para la construcción de montículos eficiente, el de "SUBIR" o el de "BAJAR"?. (1/2 pto)

f) ¿Cuál es el coste en el peor caso de "SUBIR" un elemento situado en la posición i? (1/2 pto)

4) Considera el siguiente algoritmo y responde a las siguientes preguntas (5 ptos):

```
ALGORITMO EXAMEN (i:N): N
GLOBAL: A: vector[1..n] de 1..n
        RES: N
INICIO: RES ← 0
        // 1º llamada EXAMEN(n)
METODO:
    Si (i>0) entonces
        Si A[i]=i entonces
            RES ← RES + A[i]!
        Fsi
        EXAMEN(i-1)
    Sino
        EXAMEN ← RES
    Fsi
FEXAMEN
```

a) ¿Cuándo se da el caso PEOR en asignaciones sobre CUALQUIER elemento?. (1 pto)

b) Escribe la ec. de recurrencia para el caso PEOR en asignaciones sobre CUALQUIER elemento. (1 pto)

c) ¿Serían iguales los costes en asignaciones sobre cualquier elemento en el caso MEJOR y PEOR?. Y Si se consideran únicamente comparaciones sobre el vector. ¿Serían iguales los costes del caso MEJOR y PEOR?.

d) Escribe la ecuación de recurrencia para el caso MEJOR en comparaciones sobre el vector (1 pto).

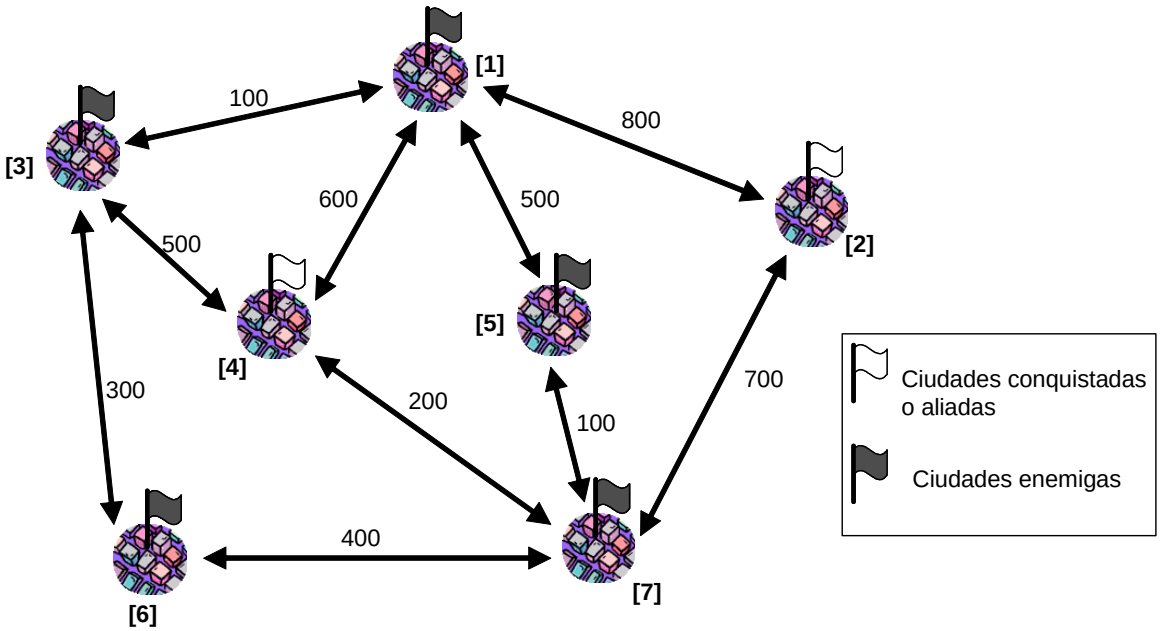
e) Cual es el coste espacial del algoritmo en el caso MEJOR y PEOR (1 pto)

NOMBRE:_____APELLIDOS:_____

PROBLEMAS (1:30) – Se pueden utilizar apuntes y libros

5) Dado el siguiente problema responde a las preguntas que se te plantean (10 ptos).

Estas en un juego de guerra, han herido a tu general, y te han puesto al mando del ejército. Tu objetivo es ocupar N ciudades con el menor número de bajas en tu ejército. En el momento de tomar el mando tus tropas ya han conquistado B ciudades y tu cuartel general te ha pasado un mapa (como el ejemplo de la figura) estimando las rutas por las que se puede atacar cada ciudad y el coste en hombres de dicho ataque en función de las características del terreno y las defensas de la ciudad. Tus tropas sólo pueden atacar una ciudad desde otra ciudad ya conquistada con la exista una ruta.



a) ¿Qué tipo de algoritmo nos permitiría resolver este algoritmo? (1 pto)

b) ¿Cuál sería el coste mínimo en bajas en el ejemplo de la figura para conquistar el resto de ciudades? (2 ptos)

c) Escribe un algoritmo que resuelva este problema si la lista de rutas entre la ciudades viene dada por una matriz M de N x N elementos, donde el valor M[i,j] representa el coste en hombres de atacar a la ciudad j desde la ciudad i (y viceversa). Si el valor es M[i,j]=-1 significa que no existe ruta. Utiliza la diagonal principal (M[i,i]) para representar que la ciudad i ya está conquistada (0) o es enemiga (1).

El resultado indicará desde que ciudad se tiene que atacar a una ciudad dada y vendrá dado por un vector (R), donde el valor R[i]=j, indica que la ciudad i se atacará desde j. Las ciudades que inicialmente son aliadas tendrán un valor 0. Escríbelo en un folio a parte.

La cabecera del problema tiene que ser la siguiente (6 ptos)

```
Algoritmo CONQUISTA
DATOS: M: matriz[1..n][1..n] de R// Matriz con la información de rutas y bajas
AUX: cont: N                     // Contador de ciudades aliadas o conquistadas
 RES: R: vector [1..n] de 0..n   // Resultado indicando desde donde hay
                                 // que atacar cada ciudad
```

6) Dado el siguiente problema responde a las preguntas que se te plantean (10 ptos).

En un juego de rol, cada jugador tiene 8 atributos, 5 de lo cuales tiene que distribuir en fila sobre un tablero para comenzar a jugar. En función del atributo anterior, y posterior, que tenga un atributo, el valor propio del atributo se incrementará en una cantidad definida.

Diseña un algoritmo de BACKTRACKING que obtenga la mejor forma de distribuir los 8 atributos sobre el tablero. Los valores propios de los atributos están representados en un vector V[8], y el valor adicional a incrementar en función del atributo anterior está representado en una matriz ANT[8x8], donde el valor ANT[i,j] es la cantidad a incrementar al atributo i si el anterior es el j. Del mismo modo, la matriz POS[8x8] contiene los valores a incrementar en función del posterior.

a) ¿Qué tipo de algoritmo de backtracking se trata? (1 pto)

b) ¿De que orden será el coste temporal en el peor caso? (No es necesario calcular el coste exacto) (1 pto)

c) ¿Cuál es el coste espacial? (1 pto)

d) Si en vez de tener 8 cartas tenemos 10 cartas ¿Se modificaría alguno de los costes?. ¿Cuál o cuales?. Indica el nuevo valor del coste. (1 pto)

e) Implementa el algoritmo en un folio a parte, utilizando la siguiente cabecera: (6 pto)

```
Algoritmo ROL (i:N)
DATOS: ANT: matriz[1..8][1..8] de N  // Matriz con valores en función del anterior
       POS: matriz[1..8][1..8] de N  // Matriz con valores en función del posterior
       PES: vector[1..8] de N    // Valor propio de cada atributo
RES:   SOL: vector [1..5] de 0..8    // Vector indicando el orden en poner las
                                     // cartas de atributos
```