Parte 1

1	Indica el modelamiento, en donde lo esperado sería una matriz bidimensional de 9×9, donde las filas representan los 9 lugares (L1 a L9) y las columnas los 9 momentos posibles (combinación de 3 días y 3 franjas horarias por día). Cada celda de esta matriz almacenaría una actividad distinta, siguiendo las restricciones dadas. Esta representación es análoga a un Sudoku donde las filas corresponden a lugares, las columnas a momentos, y las "cajas" o regiones internas están definidas por las zonas (A, B, C) y los días, lo que impone restricciones adicionales. Alternativamente, el problema podría modelarse con una matriz tridimensional de $9 \times 3 \times 3$, donde la primera dimensión son los lugares, la segunda los días, y la tercera las franjas horarias. • 1 pto: Por cumplir con todo lo mencionado anteriormente (no es necesario mencionar que el problema es equivalente a un Sudoku). . • 0.5 pts: Indica su modelamiento, pero omite detalles de cómo se ajusta al problema (Por ejemplo, no explica qué representa una columna o una fila).
4	Explica brevemente como las variables, el dominio y las restricciones asociadas. Lo esperado es definir lo siguiente: • 0.5 pts: Las variables como cada asignación en la matriz. En el caso de haber modelado en 9x9, cada variable tendría la forma x_{ij} con $1 \leq i, j \leq 9$. Si se modeló 9x3x3 sería x_{ijk} donde $1 \leq i \leq 9$ y $1 \leq j, k \leq 3$. • 0.5 pts: El dominio para cada variable es un valor entre 1 y 9 representado una posible actividad. • 1 pto por restricción: Las restricciones dependen del modelamiento, pero lo esperado es que reflejen lo explicado en el enunciado utilizando las variables y dominio definidos. Por ejemplo, $\forall k(i \neq j \implies x_{ki} \neq x_{kj})$ significa que una misma actividad no puede repetirse en distintos lugares (en caso de que cada fila represente un lugar). Debiesen ser 3 restricciones y no necesariamente planteadas como fórmula. Se puede explicar con palabras, pero haciendo referencia al dominio y variables. Corresponde mitad de puntaje en caso de errores menores.
2	Lo esperado es explicar que el espacio de búsqueda se reduce considerablemente al existir restricciones, ya que se realiza una poda en el árbol de búsqueda al determinar que cierta rama no cumple con las restricciones impuestas. El ejemplo concreto basta con que sea una situación hipotética en que cierta decisión no se toma ya que no cumple con alguna restricción y, por lo tanto, toda la rama que derivaría de esa decisión es podada. 1 pto por explicar la reducción y 1 pto por el ejemplo. Corresponde puntaje parcial en caso de errores menores.

Parte 2

5	• 1 pto: Mencionar que se usa programación dinámica para resolver el problema de maximizar el peligro sin seleccionar elementos consecutivos. • 1 pto: Describir brevemente la definición de los estados: – dp[i]: Peligro máximo acumulable hasta el número i. – max-count[i] y min-count[i]: Número máximo/mínimo de elementos usados para alcanzar dp[i]. • 2 pts: Indicar la ecuación de recurrencia clave: dp[i] = max(dp[i-1], dp[i-2] + i * c) donde c = cantidad de inventos con peligrosidad i. (esto puede guardarse en un array auxiliar). • 1pto: Indicar la complejidad temporal y espacial. La complejidad temporal puede ser O(N + K) si se considera desde el comienzo del programa u O(K) si se considera sólo la ejecución del algoritmo DP. Por otro lado, la complejidad espacial puede ser O(N + K) si se toma en cuenta un array que guarda todos los inventos u O(K) en caso contrario.
2	Una forma de solucionarlo es en medio del DP, no sólo guardar el máximo peligro en cada iteración, sino también guardar en un array auxiliar si un nivel de peligro se seleccionó o no y luego con esa información recuperar los niveles de peligros utilizados. Luego es posible recuperar los inventos utilizando el arreglo original que se recibió como input. • 2 pts: Su método funciona y aprovecha el algoritmo DP para resolverlo. • 1 pto: Su método funciona, pero no aprovecha el algoritmo DP o la información obtenida durante éste para resolver el problema.

Parte 3

3	Explica cómo almacenó los datos, ya sea una matriz de adyacencia o una lista de adyacencia. En ambos casos debe explicar cómo se ve representado un nodo y sus respectivas aristas. • 3 pts: Menciona la estructura utilizada y explica cómo representa tanto nodos como aristas. • 1.5 ptos: Menciona la estructura, pero sólo explica nodos o aristas
3	Una forma de solucionarlo es utilizar DFS o BFS partiendo desde globito e ir sumando a un contador cada vez que se llega a un nodo no visitado. Este algoritmo tiene complejidad $O(V + E)$. Se aceptan formas alternativas y menos eficientes de resolver el problema, lo importante es que el algoritmo funcione y la complejidad indicada sea correcta. • 3 pts: El algoritmo y la complejidad indicada son correctos. • 1.5 pts: Errores menores en la complejidad y su justificación.