RAZONAMIENTO LOGICO:

EJERCICIO # 1: COMPLETAR LA SECUENCIA.

1. RQP, ONM, LKJ, ,FED

a. IHG

b. CAB

c. JKL

d. GHI

2. KBJ, LCK, MDL, NEM,

a. OEP

b. OFN

c. MEN

d. PFQ

3. 104, 109, 115, 122, 130,

a. 119

b. 125

c. 139

d. 134

4. 15, 31, 63, 127, 255,

a.110

b. 170

c. 511

d. 181

EJERCICIO 2: ¿DESEA BEBER ALGO?

Dado un grupo de 5 personas: A, B, C, D, E. Se tiene:

i) Una de ellas esa artista, otra es médico, otra es periodista, otra es deportista y otra es

juez.

ii) Tres de ellas A, C y el juez, prefieren beber té, B y el periodista prefieren beber café.

iii) El deportista, D y A, son amigos entre sí, pero dos de ellos prefieren el café.

iv) El artista es hermano de C.

Sabiendo esto responde:

1. ¿Quién es el Artista?

R/ A

2. ¿Quién es el Deportista?

R/ B

3. ¿Quién es el Medico?

R/D

4. ¿Cuál de los siguientes grupos incluye a una persona que prefiere el té pero que no es el

juez?

a. A-C-E

b. D-E

c. B-C-E

d. B-D

e. Ninguno de los anteriores

PRUEBA TECNICA DESARROLLADORES

Condiciones de la prueba

. Tiempo de implementación: 6 Horas.

· Se acepta presentación en pseudocódigo.

· Sin restricción de lenguaje.

. Solo se permite el uso de framework frontend (como bootstrap, handlebars, kendoUI

etc.).

Requerimiento

Una serie de excursionistas desea crear un software que permita determinar los elementos

que puedan usarse para escalar un risco basado en las propiedades calóricas y peso de cada

elemento, para escalar el risco se especificara la cantidad mínima de calorías además del peso

maximo que se puede llevar. El software debera de indicar al usuario el conjunto de

elementos óptimos basado en que cumplan el mínimo de calorías requeridas y el llevar el

menor peso posible.

Un ejemplo es:

Mínimo de calorías: 15

Peso máximo: 10

Elementos:

E1 Peso 5 Calorías: 3

E2 Peso 3 Calorías: 5

E3 Peso 5 Calorías: 2

E4 Peso 1 Calorías: 8

E5 Peso 2 Calorías: 3

En esto caso elementos viables son: E1 E2 E4, ya que su peso total seria 9 y brindan 16

calorías.

Requerimientos opcionales de solución:

1. La aplicación debería funcionar en al menos tres sistemas operativos OSX,

Linux, Windows (no necesariamente desplegable en todos).

2. La aplicación debe ser interoperable.

3. La aplicación debe ser de fácil mantenimiento.

4. El uso de control versión para la solución por ejemplo GitHub.

5. Persistencia de la información

6. Documente la escalabilidad de la solución

1. Leer mínimo\_calorias y peso\_maximo

2. Leer elementos (nombre, peso, calorias)

3. Para cada subconjunto posible de elementos:

- Calcular peso\_total y calorias\_total

- Si calorias\_total >= mínimo\_calorias y peso\_total <= peso\_maximo:

- Guardar subconjunto como candidato

4. Elegir el subconjunto candidato de menor peso\_total

5. Mostrar subconjunto óptimo