

Nombre

Pregunta 1 (3 puntos)

La cadena "**Interstellar Food**" está abriendo franquicias por toda la galaxia y los platos variarán de precio dependiendo de la escasez de los ingredientes en el planeta en el que esté el restaurante y de las especies que lo habiten. Para ello, necesita un programa que le permita calcular los precios de los platos de su menú dependiendo del planeta en la que estemos, la especie del comensal y obviamente el plato elegido.

El restaurante ofrece tres platos principales:

- **Nebulosa de Nutrientes:** Una nube de gas espacial rica en minerales.
Tiene el código 1 en el menú y cuesta 15 créditos intergalácticos en todos los planetas.
- **Estrella Enana a la Parrilla:** Ideal para paladares que disfrutan de sabores intensos.
Tiene el código 2 en el menú y cuesta 40 créditos intergalácticos en los planetas **Ganimedes** y **Raticulin** y 30 créditos intergalácticos en el resto.
- **Agujero Negro con Salsa de Singularidad:** ¡No apto para estómagos sensibles!
Tiene el código 3 en el menú y cuesta 45 créditos intergalácticos en el planeta **Pandora**, 50 en el planeta **Acheron** y 60 en el resto.

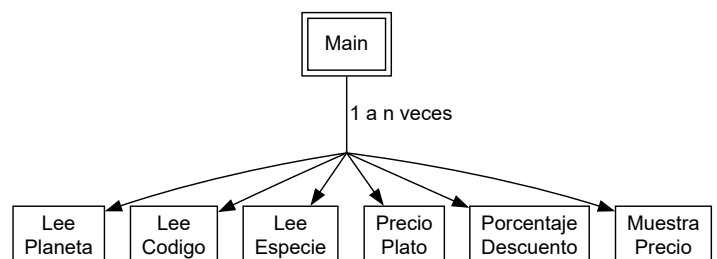
Descuentos a aplicar:

- **Gleepglorp:** Obtienen un 10% de descuento en cualquier plato.
- **Floofle:** Solo pagan la mitad si piden la Estrella Enana a la Parrilla.
- **Zorp:** Comen gratis, ¡son los críticos gastronómicos más temidos de la galaxia!
- Cualquier otra especie que no sea una de las anteriores no tendrá descuento.

El programa debe:

1. Seguir el diagrama de módulo de la imagen y usar switch de expresión para las estructuras condicionales.
2. Pedir al usuario que introduzca el código del plato (un número del 1 al 3) y el nombre de la especie.

✦ **Nota:** No hace falta que muestres un menú con los platos. Solo debe introducir un código y si el número es incorrecto deberá mostrar un aviso y pedirlo de nuevo.



3. Calcular el precio final teniendo en cuenta el plato y la especie.
4. Mostrar el precio final por consola.

5. Repetir los pasos 1-3 hasta que se introduzca la especie "Humano", momento en el cual se mostrará el mensaje *'¡Humanos no permitidos!, Avisando a la policía del sistema'* y finalizará el programa.

Ejemplo de ejecución:

```
Introduce el planeta: Acheron
Introduce el código del plato (1-3): 5
Código de plato no válido. Inténtalo de nuevo.
Introduce el código del plato (1-3): 3
Introduce la especie: Protoss
Precio base: 50
Total a pagar: 50 créditos intergalácticos.

Introduce el planeta: Kune
Introduce el código del plato (1-3): 2
Introduce la especie: Gleepglorp
Precio base: 30
Descuento: 10%
Total a pagar: 27 créditos intergalácticos.

Introduce el planeta: Raticulin
Introduce el código del plato (1-3): 3
Introduce la especie: Humano
¡Humanos no permitidos!, Avisando a la policía del sistema...
```

Criterios de calificación:

- (✅ **0,5 pts**) Define correctamente el módulo que lee código del plato.
- (✅ **1 pts**) Define correctamente el módulo que devuelve el precio usando switch de expresión.
- (✅ **0,5 pts**) Define correctamente el módulo que devuelve el porcentaje de descuento usando switch de expresión.
- (✅ **0,5 pts**) Define correctamente el módulo que muestra el precio final o el aviso de humanos no permitidos.
- (✅ **0,5 pts**) El programa compila y ejecuta correctamente de acuerdo a la especificación.

Pregunta 2 (3 puntos)

🚩 **Nota:** Para hacer este ejercicio se tiene que haber visto hasta **cadenas**.

Tenemos que desarrollar un programa que descifre un mensaje secreto codificado en una secuencia numérica.

Para ello, se te proporciona una **secuencia de números enteros**. Cada número de la secuencia representa una letra del alfabeto español (incluyendo la Ñ). Para descifrar el mensaje, debes calcular el resto de la división de cada número entre la longitud del alfabeto (**incluyendo el carácter espacio en blanco**). El resultado de esta operación te dará la posición de la letra correspondiente en el alfabeto.

El alfabeto a utilizar lo definiremos en la cadena: **" ABCDEFGHIJKLMNOPQRSTUVWXYZ"** (observa que incluye un espacio en la posición de índice 0).

Por ejemplo:

Si la longitud de la cadena del alfabeto 28 y un número en la secuencia es 31, el resto de la división (5 % 28) es 5. Esto significa que la letra correspondiente es la que está en el índice 5 de la cadena con el alfabeto, que en este caso es la letra 'E'.

Ejemplo de ejecución:

```
Secuencia recibida: 5 38 61 103 115 149 171 205 240 252 283 324 355 383 397 423 469 492
Mensaje Secreto: EJERCICIO CORRECTO
```

Debes implementar los siguientes métodos:

int[] SecuenciaNumerica(string entrada) : Esta función recibirá una cadena de texto con la secuencia numérica separada por espacios. Deberá convertir esta cadena en un array de enteros.

string Descifrar(int[] secuencia) : Esta función recibirá el array de enteros generado por la función SecuenciaNumerica. Deberá descifrar el mensaje utilizando el método del resto de la división explicado anteriormente. Para construir el mensaje descifrado, debes utilizar un **StringBuilder**.

Criterios de calificación:

- (✓ **1,25 pts**) Define correctamente el método **SecuenciaNumerica**.
- (✓ **1,25 pts**) Define correctamente el método **Descifrar** utilizando un **StringBuilder** para construir el mensaje, penalizará **0,5 pts** si no se usa **StringBuilder** para su realización.
- (✓ **0,5 pts**) El programa compila y ejecuta correctamente de acuerdo a la especificación.

Pregunta 3 (1 punto)

Escribe un programa en C# que permita al usuario calcular el **Valor Actual Neto (VAN)** de una inversión.

El VAN se calcula de la siguiente manera:

$$VAN = -A + \sum_{t=1}^n \frac{Q_t}{(1+K)^t}$$

O también:

$$VAN = -A + \frac{Q_1}{(1+K)^1} + \frac{Q_2}{(1+K)^2} + \frac{Q_3}{(1+K)^3} + \dots + \frac{Q_n}{(1+K)^n}$$

Donde:

- **n** es el **número de períodos** de la inversión en nuestro caso cada periodo será de un año.
- **A** es el **desembolso inicial** de la inversión.
- **qt** es el **flujo de efectivo** neto del período **t**. En otras palabras, es el dinero que se gana o se pierde en ese año.
- **K** es la **tasa de descuento**, esto es, el interés que se aplica a la inversión en cada período (cada año en nuestro caso) expresado en **tanto por uno**. Por ejemplo, si la tasa de descuento es del **10%**, **K** será **0.1**.

El usuario deberá introducir el **desembolso inicial**, el porcentaje de **interés anual** y el **flujo de efectivo neto** de cada año. El programa deberá mostrar el **VAN** de la inversión.

Debes usar un bucle `for` para realizar el cálculo.

Ejemplo de ejecución:

Cálculo del Valor Actual Neto (VAN)

Ingrese el desembolso inicial (A): 5000

Ingrese la tasa de descuento anual (K) como un decimal (por ejemplo, 0,1 para 10%): 0,1

Ingrese el número de períodos (n): 5

Ingrese el flujo de efectivo neto para el período 1: 1500

Ingrese el flujo de efectivo neto para el período 2: 2000

Ingrese el flujo de efectivo neto para el período 3: 2500

Ingrese el flujo de efectivo neto para el período 4: 3000

Ingrese el flujo de efectivo neto para el período 5: 3500

El Valor Actual Neto (VAN) de la inversión es: 4117.08

Criterios de calificación:

- (✓ 0.25 pts) El nombre de las variables es correcto y sigue la nomenclatura de C#.
- (✓ 0.25 pts) Utiliza un bucle for para realizar el cálculo.
- (✓ 0.25 pts) El cálculo del VAN es correcto.
- (✓ 0.25 pts) La entrada y la salida coincide con la especificación.

Pregunta 4 (4 puntos)

🚩 **Nota:** Para hacer este ejercicio se tiene que haber visto hasta **StringBuilder** pero se puede solucionar simplemente concatenando cadenas.

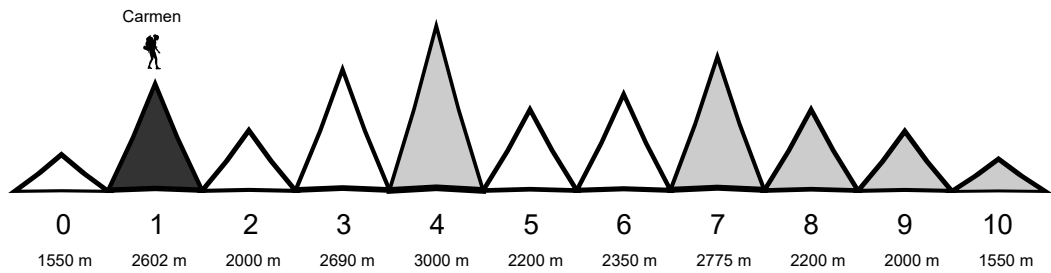
La pasión de Carmen son las montañas. Siempre que puede se pierde por los Pirineos. Cuando alcanza una cumbre le gusta mirar al horizonte y ver, de las cumbres visibles, cual es la más alta y cómo llegar a ella.

¿Podrías saber cuáles son las cumbres a las que Carmen querrá llegar o tendrá que hacer si empieza en una determinada cumbre?

Consideraciones:

1. Las montañas están numeradas y las tenemos **en un array de enteros** donde el contenido del índice `i` es la altura de la montaña `i`.
2. La primera montaña es la `0`. La montaña `i` tiene delante las montañas `i+1`, `i+2`, etc.
3. Obviamente, la última montaña no tiene ninguna delante. En este caso se considera que tiene una montaña de altura `0` delante y Carmen no tendrá cumbres a alcanzar.
4. Si hay 2 cumbres a alcanzar con la misma altura, Carmen irá a la más próxima.
5. **Entrada:** La montaña en la cual empieza Carmen (**índice en el array**).
6. **Salida:** La altura e índice de las siguientes cumbres a las que Carmen querrá llegar.

Por ejemplo, supongamos que se ha generado la siguiente cordillera y empezamos el camino en la cumbre **1** (**2602 m**).



En este caso la primera montaña visible más alta será la 4, después la 7 y consecutivamente las 8, 9 y 10.

Ejemplo de ejecución:

Cordillera generada

0	1	2	3	4	5	6	7	8	9	10
1550 m	2602 m	2000 m	2690 m	3000 m	2200 m	2350 m	2775 m	2200 m	2000 m	1550 m

Montaña de comienzo (0-10): 1

Cumbres a la que querrá llegar:

- 4 con 3000 m
- 7 con 2775 m
- 8 con 2200 m
- 9 con 2000 m
- 10 con 1550 m

Criterios de calificación:

- (☒ **0.5 pts**) Crea un programa principal que rellene un array de 11 montañas con alturas entre 1500 y 3000 metros.
- (☒ **2 pts**) Posteriormente, como en el ejemplo de ejecución me mostrará la cordillera con los índices alineados la izquierda 8 y en la siguiente fila las alturas (como en el ejemplo). Posteriormente, me pedirá la cumbre en la cual empieza y por último me mostrará las cumbres a las que Carmen querrá llegar.
- (☒ **0.5 pts**) El programa debe controlar la entrada correcta de datos y mostrarme un mensaje de error si no lo son.
- (☒ **1 pts**) Seguir **la siguiente modularización** con los interfaces definidos correctamente:

