Exámen técnico .NET developer

Nota : El tiempo de resolución de la prueba es de **2 días** como máximo después de su recepción. Una vez terminada la solución, subir el código fuente a un repositorio público de git (github por ejemplo) y brindar la URL como respuesta.

PARTE 01 : Resolver los siguientes casos usando íntegramente C#

Consideraciones preliminares:

Problema 01

Usando C#, crear una clase llamada **ChangeString** que tenga un método llamado **build** el cual tome un parámetro string que debe ser modificado por el siguiente algoritmo . Reemplazar cada letra de la cadena con la letra siguiente en el alfabeto. Por ejemplo reemplazar **a** por **b** ó **c** por **d**. Finalmente devolver la cadena.

Indicaciones

- Crear la solución en un solo archivo llamado **ChangeString.cs**
- El método build devuelve la salida del algoritmo
- Considerar el siguiente abecedario : a, b, c, d, e, f, g, h, i, j, k, l, m, n, ñ, o, p, q, r, s, t, u, v, w, x, y, z.

Ejemplos

- **entrada :** "123 abcd*3" **salida :** "123 **bcde***3"
- **entrada :** "**Casa 52" **salida :** "****Dbtb** 52"

Problema 02

Usando C#, crear una clase llamada **OrderRange** que tenga un método llamado **build** el cual tome un parámetro de colección de números enteros positivos (1,2,3, ...n) en cualquier orden.

El algoritmo debe retornar una colección de números pares y uno de números usando los números recibidos como parámetro. Ambas listas deben mostrarse en orden ascendente.

Indicaciones

- Crear la solución en un solo archivo llamado **OrderRange.cs**
- El método build devuelve la salida del

algoritmo Ejemplos

- **entrada :** [2, 1, 4, 5] **salida :** [1, 5] [2, 4]
- **entrada :** [4, 2, 9, 3, 6] **salida :** [2, 4, 6] [3, 9]
- **entrada :** [58, 60, 55, 48, 57, 73] **salida :** [48, 58, 60] [55, 57, 73]

Problema 03

Usando C#, crear una clase llamada **MoneyParts** que tenga un método llamado **build** que reciba como parámetro una cadena con un monto en soles y devuelva todas las combinaciones posibles en un arreglo.

Indicaciones

- Crear la solución en un solo archivo llamado **MoneyParts.cs**
- El método **build** devuelve la salida del algoritmo
- Considerar las siguientes denominaciones (0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100, 200)

Ejemplos

- **entrada :** "0.1" **salida :** [[0.05, 0.05], [0.1]]
- **entrada :** "0.5" **salida :** [[0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05, 0.05], ……. [0.1, 0.1, 0.1, 0.1, 0.1]]
- **entrada :** "10.50" **salida :** [[0.05, 0.05, …. 0.05], …… [10.0, 0.5]]

Los puntos suspensivos representan todas las combinaciones posibles para representar la cantidad monetaria ingresada.

PARTE 02 : Desarrollar la siguiente aplicación

Consideraciones preliminares:

- La aplicación debe ser realizada usando íntegramente C#
- Se debe usar el framework **ASP.Net Core** para el desarrollo de la aplicación web
- Se debe usar **WEB API**
- **No** se permite el **uso** de ORM,s como Entity Framework o Hibernate o Dapper, etc
- No realizar ninguna funcionalidad con JS en la vista, todo es por backend
- Se puede usar internet para el desarrollo de esta parte del examen
- Aplicar enfoque de diseño de software y patrones de diseño.

Problema práctico

Se desea realizar un pequeño prototipo de aplicación web que tenga como finalidad manejar bancos, sucursales y sus diversas órdenes de pagos. Para ello considerar los siguientes atributos de cada entidad :

Banco	Sucursales	Órdenes de pago
● nombre ● dirección ● fecha de registro	● nombre ● dirección ● fecha de registro	● monto ● moneda ● estado ● fecha de pago

La aplicación web considera las siguientes reglas de negocio. Un banco puede tener muchas sucursales y las órdenes de pago pueden ser pagadas en diferentes sucursales de cada banco en soles ó dólares. Asimismo, el estado de la orden de pago puede ser pagada, declinada, fallida y anulada.

En primer lugar, se desea crear una solución para el mantenimiento de todas las entidades en una aplicación web, con una pantalla de login considerando el siguiente cuadro de roles.
La sesión de la web debe expirar cada 5 minutos de inactividad.

Roles	Acceso a:
Operador1	Banco, Sucursales
Operador2	Órdenes de Pago
Administrador	Todos

En segundo lugar, se desea liberar un servicio web en formato JSON que pueda listar todas las órdenes de pago de una sucursal contemplando el filtrado por tipo de moneda.

Se debe liberar un servicio web en formato JSON que permita buscar todas las sucursales de acuerdo a un banco.