C#

La consola

<https://docs.microsoft.com/es-es/dotnet/api/system.console>

1. Presente el mensaje “Hola mundo” al usuario, espere la presión de una tecla y finalice.
2. Solicite el ingreso de un texto cualquiera y lo presente nuevamente al usuario.
3. Lea las teclas presionadas y finalice al presionar la tecla “X”.
4. Permita el ingreso de frases, finalizando cuando el usuario ingresa “fin”.
5. Lea las teclas presionadas y finalice al presionar Ctrl+F (Ctrl y F al mismo tiempo).
6. Lea las teclas presionadas y finalice al presionar Shift+Ctrl+F.

Cadenas

<https://docs.microsoft.com/es-es/dotnet/api/system.string>

1. Solicite al usuario el ingreso de su nombre y presente el mensaje “Hola [nombre]”.
2. Solicite al usuario el ingreso de una frase y presente un mensaje indicando si alguna contiene la palabra “fin”.
3. Solicite un ingreso al usuario y lo presente en mayúsculas.
4. Solicite un ingreso al usuario y lo presente en minúsculas.
5. Reemplace en una cadena ingresada por el usuario todas las vocales acentuadas por vocales sin acento y presente el resultado.
6. Solicite el ingreso de dos frases y determine si son iguales.
7. Solicite el ingreso de dos frases y determine si tienen la misma longitud.
8. Solicite el ingreso de dos frases y determine si son iguales, sin diferenciar mayúsculas de minúsculas.

Valores numéricos

<https://docs.microsoft.com/en-us/dotnet/api/system.int32>

<https://docs.microsoft.com/en-us/dotnet/api/system.decimal>

<https://docs.microsoft.com/en-us/dotnet/api/system.math>

1. Solicite un ingreso al usuario y determine si puede ser interpretado como un valor numérico.
2. Solicite un ingreso al usuario y determine si puede ser interpretado como un número entero.

*(De aquí en más todos los ingresos del usuario deben ser verificados, finalizando el programa con un mensaje de error en caso de ser inválidos)*

1. Solicite el ingreso de dos números al usuario y presente la suma, la resta, la multiplicación, la división, el resto, el primero elevado a la potencia del segundo.
2. Solicite el ingreso de un número al usuario y lo presente redondeado a dos decimales.
3. Solicite el ingreso de un número al usuario y presente la parte entera.
4. Solicite el ingreso de 2 números al usuario y determine el mayor.

Fechas

<https://docs.microsoft.com/en-us/dotnet/api/system.datetime>

1. Presente al usuario la fecha actual.
2. Presente al usuario la fecha actual con el siguiente formato: dd/MM/yyyy
3. Solicite el ingreso de una fecha al usuario y determine si es un ingreso válido.
4. Solicite el ingreso de una fecha al usuario utilizando, estrictamente, el formato dd/MM/yyyy
5. Le solicite al usuario el ingreso de tres números y determine si conforman una fecha válida interpretados como día / mes / año.
6. Solicite el ingreso de una fecha al usuario y le presente: el día, el año, el número y nombre del mes y el nombre y número del día de la semana.
7. Solicite el ingreso de una fecha al usuario y presente las fechas 30, 60, 90 y 180 días posteriores a la ingresada.
8. Solicite el ingreso de un número al usuario y determine si corresponde a un año bisiesto.
9. Solicite el ingreso de una fecha al usuario y presente la fecha correspondiente al primero de mes inmediato anterior.
10. Solicite el ingreso de una fecha al usuario y muestre sólo la hora.

Intervalos de tiempo

<https://docs.microsoft.com/en-us/dotnet/api/system.timespan>

1. Solicite un ingreso al usuario y determine si puede interpretarse como un intervalo de tiempo.
2. Solicite el ingreso de dos fechas al usuario y determine el intervalo en días, meses y años.
3. Solicite el ingreso de una fecha al usuario y determine cuántos días restan para el fin de año próximo más cercano.
4. Presente la fecha actual cada 10 segundos durante un minuto.

Validación del ingreso del usuario

1. Solicitar el ingreso de un número al usuario, repitiendo la operación (presentando un mensaje de error) hasta que el ingreso sea válido.
2. Solicitar el ingreso de una fecha al usuario, repitiendo la operación (presentando un mensaje de error) hasta que el ingreso sea válido.
3. Solicitar el ingreso de una cadena al usuario, repitiendo la operación (presentando un mensaje de error) hasta que el ingreso sea válido. Debe tener entre 10 y 20 caracteres.
4. Solicitar el ingreso de una serie de números al usuario, donde cada número de la serie debe ser mayor al anterior.

Arrays, listas y diccionarios

<https://docs.microsoft.com/en-us/dotnet/api/system.array>

<https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.list-1>

<https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.dictionary-2>

1. Solicite el ingreso de una serie de números al usuario y los presente en orden inverso.
2. Solicite el ingreso de una frase al usuario y presente cada palabra en una línea por separado.
3. Solicite el ingreso de una serie de fechas al usuario y las presente ordenadas en forma ascendente.
4. Solicite el ingreso de una palabra al usuario y la presente escrita al revés.
5. Solicite el ingreso de una serie de números hasta que el usuario ingrese -1. Luego presentar: el máximo, el mínimo, el promedio, la suma. Realizar todos los cálculos una vez finalizado el ingreso.
6. Realizar una aplicación que mantenga una lista de nombres. La aplicación debe interpretar los siguientes “comandos”, que el usuario ingresará hasta finalizar. Si el usuario ingresa…
   1. “alta [nombre]”, se insertará la cadena a la lista (ej.: “alta Pedro”).
   2. “baja [número entero]”, se eliminará la cadena en la posición indicada por el número ingresado. Debe presentar un mensaje de error si el número está fuera del rango de la lista.
   3. “baja [nombre]”, se eliminará la cadena de la lista. Debe presentar un mensaje de error si la cadena no existe.
   4. “mostrar”, mostrará las entradas de la lista, una por línea.
   5. “fin”, finalizará el programa.
   6. Cualquier otro ingreso indicará “comando inválido” y continuará.
7. Realizar una aplicación que solicite el ingreso de una serie de números de registro y nombres de alumno, hasta que el usuario ingrese el registro “0”. Luego, solicitará el ingreso de un número de registro y presentará (de existir) el nombre asociado, repitiendo hasta que el usuario ingrese “0” como número de registro.
8. Realizar una aplicación que:
   1. Permita ingresar y almacenar una lista de productos. La lista se compone de código de producto y precio.
   2. Permita ingresar y almacenar una lista de precios. La lista se compone de código de producto y precio. El ingreso finalizará cuando el usuario ingrese “0” como precio.
   3. Permita ingresar una serie de productos (hasta que el usuario ingrese “0”) y presente el monto total de acuerdo a los ingresos anteriores.

Colas (FIFO) y Pilas (LIFO)

<https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.queue-1>

<https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.stack-1>

1. Se le ha solicitado codificar un simulador de atención al cliente para una obra social que opera con 3 cajas. El sistema se ha diseñado como una aplicación de consola que recibe del usuario los siguientes comandos:
   1. “afiliado: [nombre]” – (Ej.: “afiliado:Perez”) le indica al sistema que el afiliado [nombre] se ha presentado en mesa de entradas (asuma que cada afiliado tiene un nombre único). Si hay una caja libre pasará directamente, si no quedará en espera.
   2. “caja: [n]” – (Ej.:”caja:1”) le indica al sistema que la caja n (1, 2 o 3) está libre. Debe asignársele un cliente según el orden de presentación en mesa de entrada.
   3. “fin” – Finaliza la ejecución.

Cualquier otro ingreso se considerará un error del operador, indicándose con un mensaje.

Luego de procesar cada comando el sistema mostrará:

1. La lista de afiliados en espera, según orden de atención.
2. El estado de cada caja (“libre” o el nombre del afiliado que está siendo atendido).
3. Se le ha solicitado codificar un sistema de seguimiento para el depósito de una empresa de logística. El sistema se ha diseñado como una aplicación de consola que recibe del usuario los siguientes comandos:
   1. “Lote [código]” – (Ej.: “Lote ABC334”) – Indica que producción ha ingresado al depósito el lote cuyo código se indica. Los códigos de lote constan de 3 letras seguidas por 3 números.
   2. “Camión [patente] Despacho [cantidad]” – (Ej.: “Camión ABCDEF Despacho 10”) Indica que ha arribado un camión identificado por [patente] (6 letras) que requiere ser cargado con [cantidad] de despachos.

Los camiones son atendidos por orden de llegada, y puestos en espera hasta que pueda completarse la carga. Los lotes son que llegan de producción son apilados, de manera tal que el último ingresado en producción será el primero en ser despachado.

Luego de procesar cada comando el sistema indicará:

1. Si se ha asignado un lote a un camión, la patente y el código de lote.
2. Si se ha completado un envío, la patente del camión despachado y la patente del próximo camión a despachar, de haber uno.
3. La cantidad de camiones en espera.
4. La cantidad de lotes en depósito.

Archivos

<https://docs.microsoft.com/en-us/dotnet/api/system.io.file>

<https://docs.microsoft.com/en-us/dotnet/api/system.runtime.serialization.formatters.binary.binaryformatter>

1. Cree un archivo de texto que contenga la cadena “hola mundo”.
2. Solicite el ingreso de la ruta de un archivo de texto y presente el contenido.
3. Solicite el ingreso de la ruta de un archivo de texto. Luego, permita el ingreso de una lista de cadenas (hasta que el usuario indique “fin”), grabándolas en el archivo a medida que el usuario las ingresa.
4. Implemente un programa con los siguientes comandos, utilizando un diccionario para mantener los datos en memoria:
   1. “Alta [legajo] [nombre]”: da de alta el legajo / nombre en el diccionario.
   2. “Baja [legajo]”: da de baja el legajo del diccionario.
   3. “Grabar [ruta de archivo]”: graba el diccionario en el archivo indicado.
   4. “Leer [ruta de archivo]”: Lee el diccionario a partir del archivo indicado.

Integración

1. Confeccionar una agenda a partir del siguiente menú principal:
   1. Ingresar datos (Apellido, Nombre, Teléfono)
   2. Buscar por apellido
   3. Buscar por nombre
   4. Grabar datos
   5. Leer datos
   6. Finalizar
2. Realizar un programa que permita el ingreso y ejecución de los siguientes comandos (diseñe la interacción con el usuario según su criterio. El sistema debe realizar las validaciones necesarias para que no se produzcan errores en tiempo de ejecución debido a un ingreso del usuario):
   1. Dar de alta un alumno (Número de registro y nombre completo)
   2. Dar de alta un curso (se ingresa el código del curso)
   3. Asignar un alumno a un curso.
   4. Desasignar un alumno de un curso.
   5. Ver la lista de alumnos asignados a un curso.
   6. Ver los cursos a los que está asignado un alumno.
   7. Ver la cantidad de alumnos en cada curso.
   8. Ver la cantidad de cursos de cada alumno.
3. Se le ha encomendado la programación de un sistema de Administración de Solicitudes de Mantenimiento. La empresa cliente cuenta con un plantel fijo de 30 técnicos, identificados por números de legajo consecutivos comenzando desde 1.

Al comienzo del día se reciben las Solicitudes de Mantenimiento. Cada una cuenta con los siguientes datos:

* 1. Número de Solicitud de Mantenimiento (identificador único)
  2. Razón Social o Nombre del Cliente
  3. Zona (Centro, Zona Norte, Zona Sur, Zona Oeste)

El sistema asignará automáticamente cada Solicitud a un técnico, cumpliendo con las siguientes condiciones:

1. Puede haber más de una Solicitud del mismo cliente. En este caso el sistema debe asignarlas al mismo técnico.
2. No pueden asignarse más de cuatro Solicitudes por día a cada técnico.
3. Todas las órdenes asignadas a un técnico deben ser dentro de una misma zona (varios técnicos pueden trabajar en la misma zona).

…y presentará una lista de técnicos, indicando en qué zona trabajarán, sus clientes asignados y la cantidad de solicitudes asignadas.

1. Se le ha encomendado un sistema de cálculo de calorías. Los nutricionistas lo utilizarán para validar que una dieta cumpla con los requisitos calóricos de un paciente, calculados de acuerdo a su peso. Se le requiere:
   1. La posibilidad de mantener una lista de platos junto con las calorías asociadas. Ejemplo: bife con puré - 1300 calorías, arroz con pollo – 1000 calorías, etc...
   2. La posibilidad de mantener una lista de dietas. Cada dieta tendrá un nombre (“Vegana”, “Hipercalórica”, “Carne blanca”), y se compone de una lista de platos.
   3. La posibilidad de ingresar el sexo y peso de un paciente y obtener una lista de las dietas que se ajustan a su contextura. La cantidad de calorías mínima por día para una persona surge de multiplicar su peso por 20 (si es mujer) o 22 (si es hombre). La cantidad máxima de calorías por día es un 50% por encima de la mínima.