Taller de Lenguajes I – 2020
Programador Universitario / Licenciatura en informática
Trabajo Práctico N° 3

OBJETIVOS

- Manejar arreglos dinámicos.
- Manejar Wiki con GitHub (investigación del alumno)
- Llevar a cabo la codificación de programas utilizando estructuras como un tipo de datos.

Ejercicios:

1) Si todavía no creó el repositorio, copie el siguiente enlace en su navegador: https://classroom.github.com/a/jKDGxp0W esto creará el repositorio para poder subir el **Trabajo Práctico Nro. 3**, realice los pasos ya aprendidos para *clonar* el repositorio en su máquina y poder comenzar a trabajar de forma *local*.

   Nota. No se olvide de incluir el archivo .gitignore en la raíz del repositorio para excluir los archivos .exe, .obj y .tds del mismo.

2) ¿Qué es una wiki?¿Para que suele usarse? Investigue, como usar la wiki del github. Cargue en la wiki de su repositorio toda las respuestas.

3) En la Wiki de su repositorio: explique con sus propias palabras cual es la diferencia entre memoria dinámica y estática. ¿Cuáles son las características de cada una? ¿En qué casos se debe usar cada una?

4) Considere la siguiente situación: En una distribuidora necesita implementar la carga de productos de sus preventistas, los cuales recolectan los productos que sus clientes van necesitando. El sistema que se utiliza en la empresa es desarrollado por equipos de programadores donde cada equipo se encarga de una tarea específica. Usted forma parte del equipo de programación que se encargará de hacer el módulo para los preventistas:

   Tareas

   Cada preventista puede visitar hasta 5 clientes, los cuales por cuestiones operativas solo puede pedir hasta 10 productos

   Las estructuras de datos necesarias son las siguientes:

```
Char *TiposProductos[]={“Galletas”,”Snack”,”Cigarrillos”,”Caramelos”,”Bebidas”};
struct Producto {
        int ProductoID;    //Numerado en ciclo iterativo
        int Cantidad;    // entre 1 y 10
        char *TipoProducto;  // Algún valor del arreglo TiposProductos
        float PrecioUnitario; // entre 10 - 100
};
struct Cliente {
        int ClienteID;              // Numerado en el ciclo iterativo
        char *NombreCliente;    // Ingresado por usuario
        int CantidadProductosAPedir; // (alteatorio entre 1 y 5)
        Pruducto *Productos //El tamaño de este arreglo depende de la variable
                            // “CantidadProductosAPedir”
};
```

1. Desarrollar una interfaz por consola donde se solicite al usuario la cantidad de clientes.
2. Solicitar al usuario la carga de los clientes creados dinámicamente en el paso anterior.
3. A medida que se dé de alta cada cliente, Generar aleatoriamente la cantidad de productos asociados al cliente y sus características. Ej: producto cargado nro. 2

   Producto
   {
   ProductoID=2
   Cantidad = 1;
   *****TipoProducto = “Snack”;
   PrecioUnitario = 100;
   }
4. Implemente una función que calcule el costo total de un producto. Esta función debe recibir como parámetro el producto y devolver el resultado de calcular la Cantidad por el PrecioUnitario.
5. Mostrar por pantalla todo lo cargado. Incluyendo un total a pagar por cliente (sumatoria del costo de todos los productos)