División de Ingeniería
Departamento de Tecnologías de Información y Electrónica
Programación Avanzada (TC2025)
Profesor: Dr. Vicente Cubells Nonell

Tecnológico de Monterrey

Tarea 1. Aritmética de apuntadores

Orientaciones sobre la entrega de la tarea

Realice los ejercicios que aparecen a continuación, cada uno como un proyecto diferente, y suba la respuesta a GitHub o Bitbucket. Comparta en Canvas la URL del repositorio donde aparecen las respuestas a los ejercicios.

Para evaluar la tarea, el profesor compilará y ejecutará el proyecto correspondiente a cada ejercicio de cada estudiante para comprobar que no contienen errores de compilación ni de ejecución, y además, se evaluará el uso de las técnicas correctas de programación vistas en las clases.

La tarea estará activa en BB hasta el 27 de agosto de 2015 a las 23:55 horas.

No se aceptan trabajos fuera de fecha ni por correo, en ambos casos la calificación de la tarea será 0 puntos.

Restricciones:

- Utilice aritmética de apuntadores
- No se pueden utilizar variables índices
- No se pueden utilizar arreglos estáticos
- Hay que utilizar estructuras

Ejercicio 1. (Puerto pesquero)

Un puerto pesquero desea llevar un registro de los barcos de pesca, sus propietarios y los tripulantes de cada embarcación. Para lo anterior, se desea almacenar de cada persona su nombre, apellidos, edad, y rol que desempeña dentro de la tripulación (arponero, cocinero, vigía, capitán, etc.). De las embarcaciones se requiere conocer el nombre, la eslora, la manga, el número máximo de tripulantes y su tripulación (información de los tripulantes).

Diseñe e implemente una aplicación en C utilizando las técnicas correctas de programación estudiadas en las clases, de tal manera que permita:

- Incorporar barcos al puerto (Tantos como se requieran).
- Incorporar tripulantes y asociarlos a los barcos existentes (siempre y cuando existan plazas disponibles). Cada tripulante puede serlo de un único barco.
- Conocer cuáles tripulantes se encuentran asignados a una embarcación determinada y el rol de cada uno de ellos. También se debe poder conocer al propietario de la embarcación (tenga en cuenta que este no es un tripulante).
- Mostrar el registro de todas las embarcaciones (con su tripulación y el número de plazas disponibles)

Ejercicio 2. (Encuesta)

Se desea programar una aplicación que permita aplicar, a una población de N personas, una encuesta que consiste de 10 preguntas, donde cada pregunta tiene 6 respuestas posibles (sólo se puede seleccionar una respuesta por pregunta). El programa debe generar aleatoriamente (ver funciones **srand** y **rand** de la biblioteca estándar) la respuesta de cada participante a cada pregunta.

El programa debe ser interactivo. Para lo anterior el programa debe:

- Solicitar los datos (nombre y edad, esta última debe ser mayor que 17 y menor que 120) de cada persona y almacenarlos en memoria.
- Solicitar el encabezamiento de cada pregunta (descripción de la pregunta) y las 6 opciones de respuesta posible para dicha pregunta y almacenarlo en una estructura de datos (arreglo, matriz, etc.) en memoria.
- Para cada participante inscrito, visualizar secuencialmente cada pregunta con sus posibles respuestas, solicitar la respuesta del usuario y almacenarla para un procesamiento posterior.
- Visualizar un histograma con los resultados de la encuesta. El histograma debe contener:
    - Cantidad de personas que respondieron cada pregunta de la encuesta (no necesariamente cada persona responde todas las preguntas)

- Para cada pregunta, generar un gráfico de la frecuencia con que se seleccionó cada respuesta. Por ejemplo, para la pregunta 1 debe aparecer algo como lo siguiente:

Pregunta 1

No. Respuesta	Frecuencia	
1	20	**********
2	6	******
3	14	**********
4	35	**********
5	15	**********
6	10	*********

- Generar un histograma como el anterior pero teniendo en cuenta los rangos de edades (18-25, 26-35, 36-45, 46-65, 66-120) en lugar de la frecuencia.

Ejercicio 3. (Hospital)

Se quiere desarrollar una aplicación para llevar el control de pacientes ingresados en un hospital. De cada paciente interesa conocer su nombre, apellidos, edad, teléfono de contacto y la cama asignada. Inicialmente, el hospital cuenta con un número de camas fijas pero se sabe que si se requiere, es posible incorporar más camas (se identifican con un número consecutivo) en cualquier momento para poder atender más pacientes. Por cuestiones de logística, siempre que se requiera incrementar el número de camas, se incrementan de 5 en 5.

Diseñe e implemente dicha aplicación en C utilizando las técnicas correctas de programación estudiadas en las clases, de tal manera que se permita:

- Incorporar pacientes al hospital (Tantos como se requiera. Si se llenan las camas disponibles y llega un nuevo paciente hay que permitir su ingreso)
- Conocer qué paciente se encuentra asignado a una cama dada o si esta se encuentra disponible
- Dar de alta a un paciente por el número de cama (se retira del hospital y la cama queda disponible)

- Mostrar el listado de todos los pacientes que se encuentran en el hospital, incluyendo el número de la cama asignada
- Conocer el número de camas disponibles y el número de camas ocupadas.