Trabajo Práctico Nivel 3

Modularidad: conceptos; Módulo concepto; Módulo: clasificación; Ámbito; Parámetros; Tipo de dato Puntero. Procedimientos definidos por el Programador.

Fecha de Inicio 27/09 *Fecha de Fin 01/10*

Ejercicio 1: Modificar el ejercicio 2 del TP de nivel 1 para que la transformación de segundos a Hora, Minutos y segundos sea un módulo.

Ejercicio 2: Crear un módulo que manipule dos números enteros quitando la última cifra del primero y añadiéndola al final del segundo. Realizar un programa que utilice dicho módulo para invertir un número.

Ejercicio 3: Implementar un módulo que reciba dos variables enteras e intercambie sus valores. Luego, utilizando el módulo realizado, implementar un programa que reciba tres variables de tipo entero y los reasigne de forma tal que a < b < c

Ejercicio 4: Construir una calculadora que trabaje solo con fracciones y que permita realizar las siguientes operaciones:
a) Operaciones de lectura y escritura de fracciones en la entrada/salida estándar.
b) Operaciones aritméticas entre fracciones: suma, incremento (de una fracción con otra), resta, decremento (de una fracción con otra), producto, cociente y simplificación de una fracción a irreducible.
c) Operaciones de comparación de fracciones, tales como: igualdad, mayor, menor, mayor o igual y menor o igual.

Crear un programa que mediante un menú de opciones permita trabajar con diferentes operaciones.

Ejercicio 5: Mejoramos el ejercicio 6 del TP de nivel 2
Modificar el programa del estacionamiento de acuerdo a los siguientes requerimientos:

a) Se debe poder ingresar la fecha del día y el saldo con el que se abre la caja. Esta tarea debe ser un módulo llamado abre_caja.
b) Se debe poder ingresar una cantidad NO conocida de vehículos y para cada vehículo mostrar cuanto debe abonar por su estadía.
c) Al finalizar el día debe mostrar la fecha, cantidad de vehículos que ingresaron y al lado el importe recaudado, también mostrar el saldo total de dinero en la caja. Esta tarea debe ser un módulo llamado cierre_dia.

Ejercicio 6: Dada una lista de N fechas ingresadas por el usuario como números enteros con el formato (DDMMAAAA), se pide validar cada una de esas fechas e indicar cuantas fechas son válidas y cuantas no lo fueron. Se debe tener en cuenta los años bisiestos.
Nota: un año será bisiesto si es divisible entre 4**.** Sin embargo**,** no puede ser divisible entre 100, a menos que también lo sea por 400. Se debe crear un módulo que separe la fecha en día, mes y año.

Ejercicio 7: Una agencia espacial transmite y recibe mensajes con sus satélites mediante una serie de bits (0 ó 1), Como estos mensajes son números grandes, requieren ser codificados para que viajen por el espacio. La codificación consiste en reemplazar grupos de bits de un mismo valor por la cantidad de bits del grupo, seguida del valor del bit, Por ejemplo, si el número es = (11000011111) su codificación resulta: (214051). Cuando la agencia recibe un mensaje codificado hace el proceso inverso para descubrir el mensaje original.
Se pide crear un programa para que, mediante un menú de opciones, la agencia pueda enviar o recibir mensajes. Cuando el mensaje se envía, el usuario ingresa dicho mensaje y se tiene que modificar por la codificación correspondiente. Y cuando el mensaje se recibe el usuario ingresa el mensaje recibido y éste se modifica por el mensaje original.
Nota: Asumir que los números binarios a codificar siempre empiezan con al menos un bit 1. Utilizar como tipo de dato long int para el número binario y su codificación. Hacer pruebas con números no muy grandes pues de otra manera la memoria se desbordará. Más adelante se podrán utilizar otros tipos de datos que permitan una mejor solución al problema. También tener en cuenta que se debe validar que el mensaje binario a enviar sea efectivamente un número con solo unos y ceros.