

Programación I – Lenguaje C – Ejercitación (Rev. 19 - 22/03/2011)

Conceptos preliminares:

Busque las definiciones en libros, diccionarios, Internet, etc. de los siguientes términos:

- 0.1) Lenguaje natural, lenguaje informático y lenguaje de programación.
- 0.2) Paradigmas de programación: imperativo, funcional, lógico, objetos.
- 0.3) Generaciones de lenguajes de programación: Primera, segunda, tercera, cuarta y quinta.
- 0.4) Lenguajes de Programación de alto, medio y bajo nivel.
- 0.5) Sintaxis, léxico, gramática y semántica.
- 0.6) Programa, Algoritmo, Estructura de datos y Token.
- 0.7) Instrucción, comando, sentencia y parámetro.
- 0.8) Variables y constantes.
- 0.9) Acumuladores y Contadores.
- 0.10) Qué son las palabras reservadas, las del usuario y qué beneficios aporta la Notación Húngara?

Estructura General del Lenguaje:

1.1) ¿De las siguientes palabras creadas por el usuario, cuales son válidas para el lenguaje C?

A01 65	Dato	
Saludo-1	lj4	
PoRcEnTaJe	int	
Float	hora hoy	
keyboard	printf	

1.2) Se va a realizar un programa en C que utilizará los siguientes datos sobre **personajes históricos**: edad, altura en metros, peso en gramos, día, mes y año de nacimiento (el año puede ser AC ó DC). Realizar la declaración de la estructura de datos con los tipos de datos primitivos más ajustados posibles. Tener en cuenta que los datos son referidos a una persona.

1.3) Observe el siguiente bloque de código:

```
{
    int a=3, b=5, c=0;
    c= ( ++a == --b ) ? 1 : 0;
    printf("a=%i , b=%i, c=%i", a, b, c);
}
```

¿Qué valores serán mostrados en pantalla? justifique. Que pasa si se reemplaza `++a` por `(a++)`?

1.4) ¿Cual de los dos bloques se ejecutará en la sentencia if (verdadero o falso)? Explique porque pasa eso

```
char a=4, b=2;
if ( b++ < a-- && b++ == a-- && b++ > a-- ) {
    printf("verdadero\n");
} else {
    printf("falso\n")
}
printf("a=%i, b=%i\n", a, b);
```

1.5) Realizar un programa que solicite un valor numérico de tipo entero decimal por pantalla y muestre su representación binaria.

Manejo de estructuras lógicas

2.1) Realizar un programa que solicite un valor por pantalla y muestre si este número es par o impar.

2.2) Ingresar dos números enteros, decir si el primero es múltiplo o divisor del segundo (o ambos). Considerar la eventualidad de ingresar números negativos y cero.

2.3) Realizar un programa que calcule el promedio de altura de los alumnos de un curso de N alumnos.

2.4) Realizar un programa que vaya leyendo números hasta que el usuario introduzca un número negativo. En ese momento, el programa deberá mostrar por pantalla el mayor, el menor y el promedio de todos los números positivos que se han ingresado.

2.5) Realizar un programa que encuentre las raíces del polinomio de segundo grado $ax^2 + bx + c=0$, utilizando la fórmula resolvente. Se requiere que si la raíz es única, se indique y se muestre un solo valor. En el caso que las raíces sean imaginarias la fórmula para obtener ambos valores complejos. Tu programa además deberá contemplar cualquier situación excepcional: divisiones por 0, raíces de números negativos, etc.

resolvente : $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$ En el caso de raíces complejas: $\frac{-b}{2a} \pm \frac{\sqrt{-(b^2 - 4ac)}}{2a}i$

2.6) Realizar una función que verifique si una fecha del tipo DD/MM/AA ingresada contiene una fecha válida. Para determinar si un año determinado es bisiesto se debe considerar: si el año es divisible por 4, que no sea divisible por 100 o que el año sea divisible por 400. Para mayor referencia http://es.wikipedia.org/wiki/A%C3%B1o_bisiesto.

2.7) Realizar un programa que muestre por pantalla todos los N° primos existentes entre dos ingresados.

- Contemplar que ambos números sean enteros positivos mayores que la unidad.
- Calcular además el costo del proceso (cantidad de divisiones) hasta verificar cada valor.
- Considerar que un número primo es aquel que posee solamente dos divisores enteros positivos, 1 y el mismo número.
- Para reducir el costo del algoritmo considerar que si el número no es divisible por dos no será divisible por ningún número par, además si no se encuentra ningún divisor entero antes de la raíz cuadrada del número se puede descartar que no se hallará ningún divisor.

2.8) realizar dos funciones para transformar un carácter ASCII de minúsculas a mayúsculas y viceversa. Verificar que el carácter ASCII sea realmente una letra. Realizar el programa de prueba.

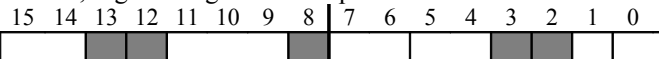
2.9) Realizar un programa que muestre por pantalla la lista de códigos ASCII y ASCII extendido en decimal, hexadecimal y carácter. Considerar que algunos caracteres no tienen representación gráfica como el BEEP o el TAB. En ese caso mostrar solamente sus valores numéricos.

Tratamiento de arrays y strings

- 3.1) Solicitar un nombre por teclado, mostrar el mismo nombre escrito en mayúsculas y luego en minúsculas. Utilizar las funciones creadas en el ejercicio 2.8.
- 3.2) Hacer una función que reciba un string, lo invierta y lo muestre por pantalla.
- 3.3) Hacer una función que reciba un string, ordene los caracteres del mismo, y muestre por pantalla el string ordenado.
- 3.4) Hacer una función que reciba un string, cuente las letras, consonantes y vocales.
- 3.5) Hacer una función que reciba un string, y muestre por pantalla la palabra más larga.
- 3.6) Hacer una función que reciba un string por parámetro, hacer con cada letra el siguiente procesamiento: si la letra es impar reemplazar por la letra siguiente, si la letra es par reemplazar con la letra anterior, devolver el nuevo string.
- 3.7) Hacer una función que haga el proceso inverso al punto 3.6 para obtener el string original.
- 3.8) Hacer un programa que solicite un string de tamaño máximo 80 caracteres, luego realizar una operación XOR con cada letra de un string que tengamos predefinido como clave (si la clave tiene menor cantidad de letras que el string ingresado, hacer el XOR repitiendo la clave desde el principio). mostrar la secuencia generada (Observar que el nuevo string seguramente no estará formado por valores ASCII de letras, por lo tanto mostrarlo como un **arreglo de chars**, si su valor ASCII es mayor que 32 mostrar cada **char** como caracter, sino mostrar el caracter como un valor numerico). hacer el proceso inverso para obtener la cadena original.

Punteros

- 4.1) Realizar una función que intercambie un valor entre dos variables enteras cualquiera. Realizar un programa de prueba donde se intercambien entre sí las variables A, B y B, C.
- 4.2) Hacer un programa que transforme los caracteres en minúsculas de la pantalla a mayúsculas.
- 4.3) Realizar un programa que intercambie los caracteres de la pantalla de derecha a izquierda y de arriba hacia abajo.
- 4.4) Realizar un programa que tome el vector de 2000 valores de 16 bits ubicado a partir de la dirección de memoria b800:0000 y los ordene mediante cada uno de los métodos de ordenación conocidos (burbujeo, intercambios, comparaciones sucesivas, quicksort, etc..).
- 4.5) hacer un programa que lea dos Bytes de configuración de la BIOS en la dirección 0x0040:0010 y determine la configuración del PC, según la siguiente interpretación:



Bits	Significado
0	1= si al menos 1 unidad de diskettes conectada
1	1= coprocesador conectado
5-4	Modo de video en BOOT 00 = EGA/VGA 01 = color 40 * 25 10 = color 80 * 25 11 = monocromo 80 * 25
7-6	Cantidad de unidades de diskettes (si bit 0=ON) 00 = 1 unidad 01 = 2 unidades 10 = 3 unidades 11 = 4 unidades
11 - 9	Cantidad de puertos serie (RS 232)
15-14	Cantidad de puertos de impresora

Ver *low memory map* de *Ralph Brown* para más información y más configuraciones.

Matrices y Archivos.

5.1) Torneo de Fútbol:

Se posee en disco el archivo **"Part96.dat"** que contiene 15 registros con la siguiente estructura:
 20 chars con el nombre del equipo.
 14 unsigned short int con los goles a favor de cada uno de los 14 partidos
 14 unsigned short int con los goles en contra de cada uno de los 14 partidos

Se debe hacer la lista del torneo, mostrando por pantalla en forma de tabla:

- Nombre del equipo
- Puntos (cada partido ganado son 3 puntos, cada empatado 1 punto)
- Total de Goles a favor y en contra
- Cantidad de partidos ganados, empatados y perdidos.

Al final indicar quién fue el ganador del torneo, considerando que a igualdad de puntos, el ganador se obtiene por diferencia de goles a favor.

5.2) **Billetes de \$2:**

Se requiere administrar un archivo que contenga una lista de billetes de \$2 para jugar en el programa televisivo "Sorpresa y 1/2". El archivo debe tener la siguiente estructura de registro:

- char Serie
- long int Numero
- char Dueño (letra inicial)

5.2.a) Se necesita un proceso para agregar nuevos billetes al archivo. Grabar el registro convirtiendo automáticamente Serie y la Inicial del Dueño a mayúscula.

5.2.b) Se necesita un proceso para verificar en la lista si se encuentra el billete que salió en el programa. Se debe ingresar serie y número del billete, procesar todo el archivo, verificando billetes de la misma serie. si el billete existe: mostrar un mensaje tipo "Billete encontrado, el dueño es X felicidades". Si no se encontró, al final del programa mostrar "El más cercano es: S99999999 pertenece a X".

5.2.c) Se necesita un proceso para mostrar el importe total acumulado y la cantidad de billetes e importe que tiene cada uno de los integrantes.

5.3) **VIAJES.DAT**

Realizar un programa para la terminal de micros de Monte Chingolo que acceda a los datos del archivo "VIAJES.DAT" cuya estructura de registro es la siguiente:

Tipo	Dato almacenado
char[25]	Lugar de Origen o Destino
char	Hora de arribo ó de salida
char	Minuto de arribo ó de salida
char	Estado
char[20]	Empresa
unsigned int	Número de Anden (hay 25 andenes por piso)

Se necesita un programa que liste todos los viajes **para una hora determinada** de la siguiente forma:

Si es arribo **D** si es salida **A**

El **nombre del lugar** de origen o destino.

El **andén (PB** para los andenes de Planta Baja y **1P** para 1º Piso)

La **hora** de partida

el **Tipo** de Servicio

El nombre de la **empresa**.

Las **comodidades** del micro en forma abreviada.

Tener en consideración que el Bit más significativo del Número de anden, indica por verdadero si el colectivo estaciona en Planta Baja o en primer piso.

El Byte de estado se interpreta de la siguiente forma:

Bit	Mostrar	Significado
0	Video	Video
1	W/C	Baño
2	Azaf	Azafata
3,4	Comun Semic Cama Ejec.	00- Servicio Común 01- Semicama 10- Cama 11- Ejecutivo
5		No se utiliza
6	Bar	Bar a bordo
7		1-Si es arribo, 0- si es partida

Ejemplo de salida en pantalla:

```

Ingrese la Hora: 16_
Lugar                Andén  Hora Servicio  Empresa
-----
De BRAGADO           PB  5  16:00  Comun  Belluci Hnos.
  *W/C
A  Bahia Blanca      PB  5  16:10  Ejec.  Belluci Hnos.
  *VIDEO *W/C *AZAF
De Buenos Aires     PB 10  16:22  Semic  Chevallier
  *VIDEO *W/C *AZAF
De Miramar           1P  4  16:25  Ejec.  El Rapido
  *W/C *BAR
A  Bariloche         PB 10  16:40  Cama   Chevallier
  *VIDEO *W/C *BAR
    
```

