MANUAL DE PRACTICAS

DE LABORATORIO:

"Algoritmos Computacionales"

Elaborado por:
Dr. en C. y E. José Luis Sánchez Ramírez
& Dra. en C. y E. Cristina Juárez Landín
Colaboración:
Mtro. en S. C. Marco Alberto Mendoza Pérez
Fecha: Septiembre de 2017

Validado por el H. Consejo Académico,
En su Sesión Ordinaria el 28 de Septiembre de 2017

Revisado por:
Dr. Samuel Olmos Peña
Coordinador de Licenciatura en Informática Administrativa
Fecha: 9 de octubre de 2017

Aprobado por el H. Consejo de Gobierno,
En su Sesión Extra Ordinaria el 16 de octubre de 2017

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

INDICE

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRESENTACIÓN

Las presentes prácticas de laboratorio fueron desarrolladas en estricto apego al programa de la Unidad de Aprendizaje de Algoritmos Computacionales perteneciente al mapa curricular de la Licenciatura en Informática Administrativa, con la finalidad de que los alumnos puedan poner en práctica los conocimientos teóricos que se van adquiriendo a lo largo del curso.

Este manual consta de 9 prácticas, cuya complejidad es acorde al tema que abarca. En este sentido la primera práctica es la más sencilla, pero a su vez es fundamento de las prácticas subsecuentes.

Cada práctica consta de un Objetivo, Marco teórico, Listado de equipo y materiales a utilizar, Desarrollo y al final el alumno debe realizar las Conclusiones de cada práctica, así mismo al finalizar cada práctica se ofrecen Referencias bibliográficas que servirán de apoyo al estudiante para realizar todas las actividades que indican la prácticas sin dificultades.

Las primeras 6 prácticas están planteadas para trabajar en el Software libre FreeDFD para generación e implementación de Diagramas de Flujo, y las 3 últimas están planificadas para que se realicen en un Lenguaje de Programación como Lenguaje C o C++, con la finalidad de implementar de manera práctica los algoritmos.

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRACTICA 1

INTRODUCCIÓN AL SOFTWARE FreeDFD

OBJETIVO:

Que el alumno se introduzca en el entorno de la interfaz de FreeDFD, para que se familiarice con cada uno de los elementos que componen el menú de la interfaz.

INTRODUCCIÓN:

Para conocer el entorno de trabajo sobre el que se realizarán diagramas de flujo es necesario tener conocimientos previos del manejo de una PC, relacionarse con la interfaz de FreeDFD. FreeDFD es un programa de libre disposición para ayuda al diseño e implementación de algoritmos expresados en diagramas de flujo (DF). Además incorpora opciones para el depurado de los algoritmos, lo que facilita enormemente la localización de los errores de ejecución y lógicos más habituales.
Su utilización es muy sencilla, al tratarse de una herramienta gráfica, en esta práctica nos vamos a centrar en el uso básico de las herramientas de diseño y depuración.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- El Software FreeDFD

Material:

- Lápiz o Bolígrafo (al menos dos colores diferentes)
- Regla
- Goma

DESARROLLO:

ACTIVIDAD 1:
Iniciar la aplicación FreeDFD, observe todas las opciones de menú que tiene.
En la siguiente página dibuje el menú, con cada una de las opciones que presenta, tanto en el menú como en los botones.
NOTA: Si es necesario puede utilizar el reverso de estas páginas para realizar sus anotaciones

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

ACTIVIDAD 2:

2.1. INICIO DE DFD

La ejecución de DFD presenta una pantalla de inicio, donde tiene una barra de menús desplegables y otra barra de herramientas por botones, como se muestra.

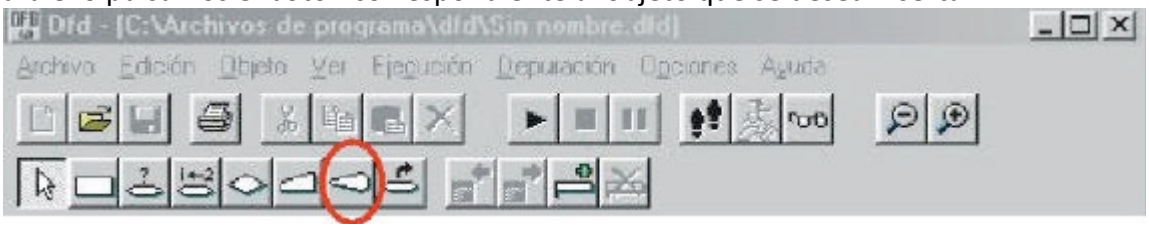
Aunque puede accederse a todas las opciones que comentaremos a continuación a través del menú, y con atajos de teclado, en estas notas las describiremos a través de los botones correspondientes.

- El bloque de botones de objetos nos permite seleccionar los distintos elementos (objetos) que vamos a introducir en el DF: sentencias de asignación, selección, iteración, operación, resultado, etc...
- El bloque de ejecución permite poner en funcionamiento el algoritmo
- El bloque de depuración se utiliza, en caso de funcionamiento incorrecto, para detectar errores en la construcción del algoritmo y corregirlos.
- Los botones de subprogramas permiten introducir funciones definidas por el programador
- Los restantes botones tienen una funcionalidad similar a la de las restantes aplicaciones Windows: abrir fichero, guardar fichero, cortar, pegar, ... Puede verse su tarea asociada acercando el cursor del ratón (sin hacer clic) al botón correspondiente.

2.2 Primer ejemplo de diseño con FreeDFD

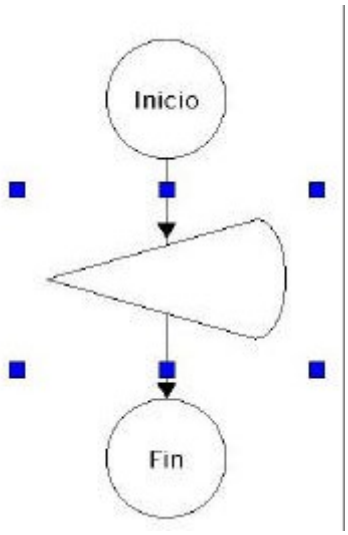
Construiremos un primer ejemplo sencillo de algoritmo para ilustrar las capacidades más básicas de DFD. Dicho algoritmo consistirá en pedir un número al usuario y presentarlo por pantalla.

La operación básica será la de inserción de objetos. En primer lugar, insertaremos una sentencia de salida que le pida al usuario el número que posteriormente se va a imprimir. Para ello pulsamos el botón correspondiente al objeto que se desea insertar

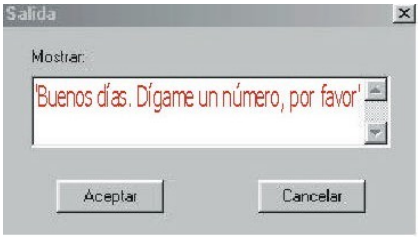


y llevamos el ratón al punto donde vamos a insertarlo. La inserción se realiza pulsando el botón izquierdo, con lo que tendremos una situación como la siguiente:

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx



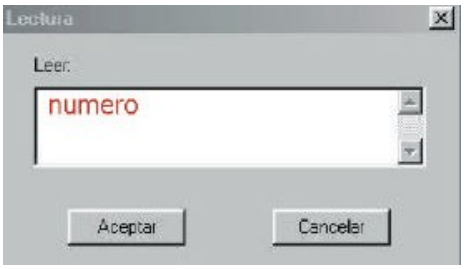
Los puntos azules indican qué objeto se acaba de insertar. Para introducir en la sentencia de salida el mensaje que queremos imprimir será necesario EDITAR dicho objeto, haciendo doble clic sobre el mismo. De este modo se abre una ventana donde podemos dicho mensaje (por ejemplo 'Buenos días. Dígame un número, por favor').



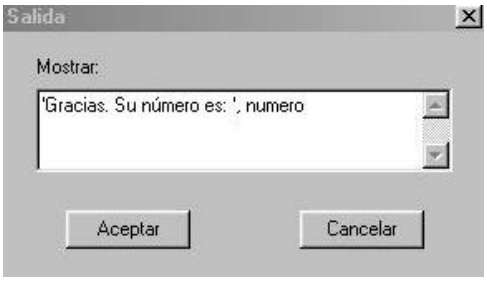
Como el mensaje es una cadena de caracteres, no debemos olvidarnos de las comillas simples al inicio y final de la misma.
Seguidamente vamos a insertar una sentencia de ENTRADA, para almacenar en una variable el valor del número que nos proporcione el usuario. Para ello pulsaremos el botón correspondiente.

y lo insertaremos a continuación de la sentencia de salida anterior. Si editamos el objeto, haciendo doble clic sobre el mismo, aparecerá una pantalla cuyo cuadro de texto nos permitirá darle nombre a la variable donde vamos a guardar el valor (en este ejemplo la variable se va a llamar numero:

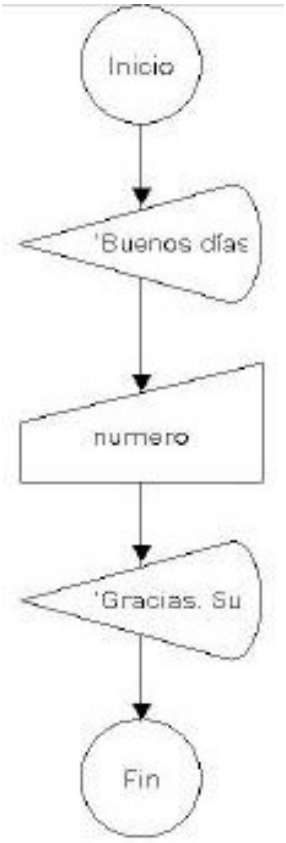
Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx



Para finalizar, mostraremos al usuario el número que ha introducido, para lo cual insertaremos una nueva sentencia de SALIDA, que editaremos para que muestre el siguiente mensaje:



Con lo que el algoritmo tendrá el siguiente aspecto en pantalla:



Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

2.3 Un primer ejemplo de ejecución con FreeDFD

Tras haber diseñado el algoritmo podemos probar a ejecutarlo, al objeto de detectar posibles errores en él. Para ello utilizaremos los botones de ejecución, y en particular el botón EJECUTAR

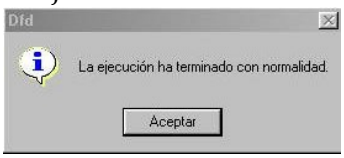
Que pondrá en marca el algoritmo.

La primera sentencia en ejecutarse será la de SALIDA, que mostrará en pantalla el mensaje correspondiente:

Seguidamente la de ENTRADA, que nos muestra un cuadro de texto donde introduciremos el valor que queramos darle a la variable (por ejemplo, 123.45):

y, finalmente, la última sentencia de SALIDA:

Cuando el algoritmo finaliza su ejecución sin error se muestra el siguiente mensaje:



Dado que el algoritmo es correcto, procederemos a guardarlo (por ejemplo, con el nombre entradasalida). La opción de guardar es similar a la de cualquier aplicación Windows, por lo que no merece mayor comentario. Únicamente recordar que en general, durante el proceso de elaboración de un algoritmo (que puede ser largo) debemos guardar frecuentemente en disco el diseño, al objeto de prevenir posibles fallos o errores que dejen inutilizado el ordenador y provoquen la pérdida del trabajo realizado.

2.4 Ejercicio 1: Con el objeto de ver ejemplos de errores, modificar el algoritmo anterior en el siguiente sentido:

1. errores de sintaxis: Eliminar una de las comillas en alguna de las sentencias de salida y ejecutar el algoritmo.
2. errores de ejecución: Eliminar la sentencia de entrada (para ello seleccionar dicha sentencia haciendo clic sobre el objeto y pulsar el botón ELIMINAR o la tecla SUPRIMIR). Ejecutar el algoritmo.

Ejercicio 2: diseñar un nuevo algoritmo que pida al usuario dos números a y b y le diga cuál es su suma. Guardar su ejercicio.

CONCLUSIONES:

El alumno debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.

REFERENCIAS

Sugerida

- Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo, Ed. Servicio de Publicaciones de la Universidad de Málaga, Mayo 2000

PRACTICA 2

IMPLEMENTACIÓN DE ALGORITMOS Y DIAGRAMAS DE FLUJO CON FreeDFD

OBJETIVO:

Que el alumno realice la implementación de algoritmos sencillos, de modo que pueda poner en práctica lo visto en clase y al mismo tiempo utilizar los diagramas de flujo como una herramienta de representación gráfica de algoritmos.

INTRODUCCIÓN:

Para poner en práctica los temas vistos en clase el alumno debe elaborar diferentes algoritmos, tanto propuestos en clase como propuestos por el mismo. Se debe de poner mucha atención en que los algoritmos que realice cumplan con las características de un algoritmo (Preciso, Definido y Finito). En todos los casos debe de realizar primero la propuesta de algoritmo y diagrama de flujo a mano y una vez que haya concluido debe de hacer su reporte en limpio, realizando el algoritmo y diagrama de flujo en computadora.
Nota: Para la elaboración de los diagramas de flujo, utilizar la herramienta FreeDFD.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- Editor de Diagramas de Flujo FreeDFD
- Procesador de Texto o Presentaciones (Word o PowerPoint)

Material:

- Esta práctica impresa
- Lápiz o Bolígrafo (al menos dos colores diferentes)
- Regla
- Goma

DESARROLLO:

I. Elaborar los siguientes algoritmos:
   1. Calcular el valor de la suma 1+2+3+4+ …+100.
   2. Realizar la suma de todos los números pares entre 2 y 1000.

II. Implemente un algoritmo en FreeDFD que resuelva una ecvuación de segundo grado de la forma:

$$ax^2 + bx + c = 0$$

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

III. Finalmente elabore los diagramas de flujo en FreeDFD de cada uno de los problemas anteriores

NOTA: Si es necesario puede utilizar el reverso de estas páginas para realizar sus anotaciones

CONCLUSIONES:

El alumno debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.

REFERENCIAS

Sugerida

- Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo, Ed. Servicio de Publicaciones de la Universidad de Málaga, Mayo 2000

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRACTICA 3

UTILIZACIÓN DE ASIGNACIÓN, DECISIÓN Y CICLOS EN FreeDFD

OBJETIVO:

Que el alumno conozca y utilice los distintos tipos de objetos que maneja FreeDFD.

INTRODUCCIÓN:

DFD permite incluir los objetos básicos de programación estructurada: asignación, selección, lazos y subprogramas. Cualquier objeto que se inserte en el algoritmo puede ser editado haciendo doble clic, lo que permite definir los elementos que lo componen. Esto quiere decir que la EDICIÓN permitirá, por ejemplo, en el caso de:

- Sentencias de salida: indicar la expresión que se va a presentar en pantalla.
- Sentencias de entrada: indicar los nombres de las variables donde se guardará la información.
- Sentencias de asignación: indicar las expresiones y los nombres de las variables donde se guardará el resultado.
- Estructuras de selección: indicar la condición.

Otra acción interesante sobre los objetos es la SELECCIÓN de los mismos (clic sobre el objeto), que permite realizar acciones como eliminarlos y cortarlos o copiarlos para posteriormente pegarlos en otro punto del algoritmo.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- El Software FreeDFD

Material:

- Lápiz o Bolígrafo

DESARROLLO:

Utilizando las herramientas de FreeDFD de “asignación”, “decisión”, “ciclo para” y “ciclo mientras” realizar las siguientes actividades.

ACTIVIDAD 1: (Utilizando la herramienta “*asignación*”)

- Diseñar y ejecutar un algoritmo que pida dos números “*a*” y “*b*” al usuario y calcule su suma, resta y producto, y muestre los resultados al usuario.

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

ACTIVIDAD 2: (Utilizando la herramienta "*decisión*")

- Diseñar y ejecutar un algoritmo que indique si un número "*a*" pedido por teclado es positivo o negativo. Guardarlo con el nombre seleccion1.
- Modificar el algoritmo de la actividad 1 para que incluya la división, y que no produzca error de ejecución cuando "*b*" sea igual a cero.

ACTIVIDAD 3: (Utilizando la herramienta "*ciclo para*")

- Diseñar y ejecutar un algoritmo que calcule el factorial de un número "*n*" pedido al usuario por teclado. Probarlo con valores n=-1, 0, 1, 2 y 100.

ACTIVIDAD 4: (Utilizando la herramienta "*ciclo mientras*")

- Implementar el problema anterior, pero utilizando el "ciclo mientras".

Para cada una de las actividades realizar su implementación en el software FreeDFD, verificar el funcionamiento y entregar su reporte con sus diagramas, explicaciones y conclusiones.

CONCLUSIONES: **(El alumno debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.)**

REFERENCIAS

Sugerida

- Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo, Ed. Servicio de Publicaciones de la Universidad de Málaga, Mayo 2000
- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRACTICA 4

MANEJO DE VECTORES Y SUS PRINCIPALES OPERACIONES EN FreeDFD

OBJETIVO:

Que el alumno conozca y utilice la forma en que se manejan los vectores en FreeDFD, así como los algoritmos de búsqueda de valores con diferentes grados de dificultad que se pueden implementar, así como la forma en que se puede convertir un vector en una matriz y viceversa.

INTRODUCCIÓN:

En la práctica anterior se aprendió a definir vectores en DFD, así como también se implementó un algoritmo de ordenamiento que permitió ordenar los datos del vector de manera descendente y ascendente. En esta práctica se realizarán tres importantes operaciones que se pueden realizar con un vector.

- Buscar un valor determinado.
- Convertir un vector en Matriz.
- Convertir una matriz en Vector.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- El Software FreeDFD

Material:

- Libreta y Bolígrafo (para realizar anotaciones)

DESARROLLO:

Utilizando las herramientas de FreeDFD realizar las siguientes actividades.

ACTIVIDAD 1:

- Diseñar y ejecutar un algoritmo que permita introducir un vector de “n” elementos, posteriormente debe de preguntar un valor a buscar y debe indicar si el valor dado está o no en el vector.

ACTIVIDAD 2:

- Modificar el algoritmo anterior para decir además si el valor dado se encuentra en el vector, cuantas veces aparece.

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

ACTIVIDAD 3:

- Modificar el algoritmo anterior para que nos indique, aparte de cuantas veces aparece, en que posiciones se encuentra dentro del vector.

ACTIVIDAD 4:

- Diseñar y ejecutar un algoritmo que permita introducir un vector de "n" elementos, posteriormente debe de preguntar un valor "r" (múltiplo de "n"), y debe de convertir al vector de 1*n elementos en una matriz de r*(n/r) elementos.

ACTIVIDAD 5:

- Diseñar y ejecutar un algoritmo que permita introducir una matriz de "n*m" elementos, posteriormente debe convertir la matriz en un vector de 1*(n*m) elementos.

Para cada una de las actividades realizar su implementación en el software FreeDFD, verificar el funcionamiento y entregar su reporte con sus diagramas, explicaciones y conclusiones.

CONCLUSIONES: **(El alumno debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.)**

REFERENCIAS

Sugerida

- Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo, Ed. Servicio de Publicaciones de la Universidad de Málaga, Mayo 2000
- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRACTICA 5

MANEJO Y REALIZACIÓN DE OPERACIONES CON MATRICES EN FreeDFD

OBJETIVO:

Que el estudiante aprenda la definición, manejo y utilización de variables como matrices, para su definición, manejo e implementación en FreeDFD. Las matrices son variables que están compuestas de un número de "nxm" variables del mismo tipo acomodados en un arreglo de "n" filas y "m" columnas.

INTRODUCCIÓN:

Para definirlas y utilizarlas en FreeDFD es necesario hacer uso de ciclos de repetición, ya sea Ciclo "Para" o "While". Generalmente utilizamos el tipo "Para", pero también es posible utilizar el ciclo "While".
Como sabemos los objetos que se inserten en el algoritmo pueden ser editados haciendo doble "clic", lo que permite definir los elementos que los componen. Esto quiere decir que la EDICIÓN permitirá, por ejemplo, en el caso de:

- **Sentencias de salida:** indicar la expresión que se va a presentar en pantalla.
- **Sentencias de entrada:** indicar los nombres de las variables donde se guardará la información.
- **Sentencias de asignación:** indicar las expresiones y los nombres de las variables donde se guardará el resultado.
- **Estructuras de selección:** indicar la condición.

Otra acción interesante sobre los objetos es la SELECCIÓN de los mismos (clic sobre el objeto), que permite realizar acciones como eliminarlos y cortarlos o copiarlos para posteriormente pegarlos en otro punto del algoritmo.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- El Software FreeDFD

Material:

- Lápiz o Bolígrafo

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

DESARROLLO:

Utilizando las herramientas de FreeDFD de “salida”, “entrada”, “asignación”, “decisión” y “ciclo para” realizar las siguientes actividades.

ACTIVIDAD 1:

- Diseñar y ejecutar un algoritmo que pida dos números “*n*” y “*m*” al usuario y posteriormente almacene los datos de una matriz.

ACTIVIDAD 2:

- Modificar el algoritmo para que muestre los valores de la matriz introducidos.

ACTIVIDAD 3:

- Modificar nuevamente el algoritmo para que pueda almacenar los valores de dos matrices de las mismas dimensiones “nxm”, una vez almacenados debe mostrar los valores almacenados.

ACTIVIDAD 4:

- Hacer la modificación al algoritmo para que una vez almacenadas las dos matrices el programa realice la suma de estas y muestre el resultado.

ACTIVIDAD 5:

- Finalmente introducir una condición al programa de modo que pregunte al usuario si desea sumar o restar las matrices y una vez dada la opción el algoritmo realice la opción seleccionada y muestre el resultado obtenido.

Para cada una de las actividades realizar su implementación en el software FreeDFD, verificar el funcionamiento y entregar su reporte con sus diagramas, explicaciones y conclusiones.

CONCLUSIONES: **(El alumno debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.)**

REFERENCIAS

Sugerida

- Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo, Ed. Servicio de Publicaciones de la Universidad de Málaga, Mayo 2000
- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRACTICA 6

CREACION Y UTILIZACIÓN DE SUBPROGRAMAS EN FreeDFD

OBJETIVO:

Que el alumno conozca y pueda generar subprogramas en FreeDFD de forma eficiente la forma en que se crean y utilizan programas en FreeDFD, realizará implementaciones sencillas con esta herramienta y posteriormente los aplicará en sus prácticas ya hechas.

INTRODUCCIÓN:

La utilización de subprogramas o subrutinas es de mucha ayuda e importancia cuando estamos realizando un algoritmo o programando. Por que permite eficiente el proceso simplificando la complejidad de un programa que de forma general se dividiría en varios subprogramas.

Los botones que están relacionados con el uso de subprogramas son los siguientes:

 Se utiliza para crear un subprograma.

 Se utiliza para eliminar un subprograma.

 Se utiliza para pasarnos a un subprograma hacia la derecha.

 Se utiliza para pasarnos a un subprograma hacia la izquierda.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:

Equipo:

- PC

Software:

- El Software FreeDFD

Material:

- Libreta y Bolígrafo (para realizar anotaciones)

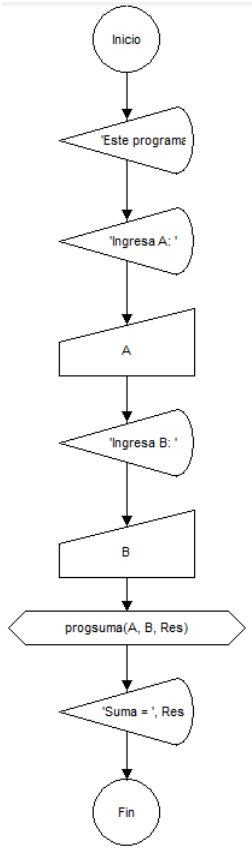
DESARROLLO:

Utilizando las herramientas de FreeDFD realizar las siguientes actividades.

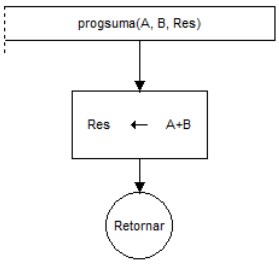
ACTIVIDAD 1:

- Realizar un programa que nos permita leer dos valores “A” y “B”, después realizar un subprograma que realice la suma, los parámetros de este subprograma son los valores a sumar “A”, “B” y el resultado “Res”. (como se muestra a continuación)

Programa Principal



Subprograma para hacer la suma.



ACTIVIDAD 2:

- Realizar un programa que nos permita introducir un valor “L” y después nos permita introducir los “L” valores de un vector pero utilizando un subprograma, cuyos parámetros son el tamaño “L” del vector y el vector “V” donde guardaremos.

ACTIVIDAD 3:

- En el mismo programa generar un subprograma que nos permita mostrar los valores introducidos, en este caso al igual que en el caso anterior los parámetros que le debemos dar es el tamaño del vector “L” y el mismo vector a desplegar “V”.

ACTIVIDAD 4:

- Realizar un subprograma que permita ordenar los elementos de un vector, en este caso los parámetros del subprograma son el tamaño del vector “L” y el mismo vector a ordenar “V”.

ACTIVIDAD 5:

- Realizar un subprograma que permita multiplicar una matriz de n*m, por una matriz de m*k, los parámetros de entrada son las matrices 1 y 2 y la salida es la matriz 3, que tiene el resultado.

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

Para cada una de las actividades realizar su implementación en el software FreeDFD, verificar el funcionamiento y entregar su reporte adjuntado evidencia de los diagramas, explicaciones y conclusiones.

CONCLUSIONES: **(El alumno debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.)**

REFERENCIAS

Sugerida

- Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo, Ed. Servicio de Publicaciones de la Universidad de Málaga, Mayo 2000
- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

PRACTICA 7

ALGORITMOS PARA ENTRADA Y SALIDA DE DATOS

OBJETIVO:

El alumno realizará programas en lenguaje C utilizando las instrucciones de entrada y salida de datos de manera correcta, así como en el desarrollo de sus respectivos algoritmos y diagramas de flujo.

INTRODUCCIÓN:

Los programas interactúan con el exterior, a través de datos de entrada o datos de salida. El lenguaje C proporciona facilidades para entrada y salida, para lo que todo programa deberá tener el archivo de cabecera stdio.h. En C la entrada y salida se lee y escribe de los dispositivos estándar de entrada y salida, se denominan stdin y stdout respectivamente. La salida, normalmente, es a pantalla del computador, mientras que la entrada se capta del teclado.

En el archivo stdio.h están definidas macros, constantes, variables y funciones que permiten intercambiar datos con el exterior.
Nota: El compilador de C que se utilizará es Turbo C++, aunque la práctica se puede desarrollar sin problema en algún otro compilador, solo tomando en cuenta las pequeñas variaciones que existen.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:

Equipo:

- PC

Software:

- Compilador de lenguaje C (Turbo C++)

Material:

- Diccionario Inglés-Español
- Lápiz o Bolígrafo (al menos dos colores diferentes)
- Regla o escuadra
- Esta práctica impresa

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

DESARROLLO:

FUNDAMENTO:

Salida

La salida de datos de un programa se puede dirigir a diversos dispositivos, por ejemplo la pantalla, una impresora o archivos. La función printf() visualiza en la pantalla datos del programa, transforma los datos que están en representación binaria, a ASCII según los códigos de formato predefinidos.

La forma general que tiene la función printf() es:

printf ("cadena_de_control", dato_1, dato_2, ..., dato_n);

Donde:

cadena_de_control: Contiene los códigos del formato de los datos que se desean mostrar en pantalla.

dato_1,dato_2,...,dato_n: Son las variables y/o constantes que se desean mostrar

A continuación se muestran los códigos de formato más utilizados y su significado.

Códigos de formato	*Significado*
%d	El dato se convierte a entero decimal
%o	El dato entero se convierte a octal
%x	El dato entero se convierte a hexadecimal
%u	El dato entero se convierte a entero sin signo
%c	El dato se considera de tipo caracter
%f	El dato se considera de tipo flotante en notación decimal con parte entero y con n dígitos de precisión
%s	El dato es una cadena de caracteres
%lf	El dato es de tipo doble.

Así por ejemplo si:

```
suma=0;
suma=suma+10;
printf("%s %d","Suma = ", suma);
```

Se visualizará en pantalla como,

Suma = 10

El número de argumentos de printf( ) es indefinido, por lo que se pueden mostrar cuantos datos sean necesarios.

Así, suponiendo que se tienen las siguientes asignaciones:

i = 5; j = 12; c = 'A'; n = 40.791512;

La instrucción:

printf("%d %d %c %f", i, j, c, n);

Visualizará en la pantalla

5 12 A 40.79152

La función printf( ) convierte, da formato de salida a los datos y los escribe en pantalla. La cadena de control contiene códigos de formato que se asocian uno a uno con los datos.

Cada código comienza con el carácter de %.A continuación puede especificarse el ancho mínimo del dato y termina con el carácter de conversión. Así, suponiendo que se tiene las siguientes asignaciones:

i = 11; j = 12; c = 'A'; n = 40.791512;

La instrucción:

printf("%x %3d %c %.3f" , i, ,j, c, n);

Visualizará en pantalla

B 12 A 40.792

El primer dato es 11 en hexadecimal (%x), el segundo es el número entero 12 en un ancho de 3, le sigue el carácter A y, por último el número real n redondeado a 3 cifras decimales (%.3f). Un signo menos (-) a continuación de % indica que el dato se ajuste a la izquierda en vez del ajuste a la derecha por default, por ejemplo:

Sentencia	***Salida en pantalla***
`printf("%15s","Hola Pablo");`	Hola Pablo
`printf("%-15s","Hola Pablo");`	Hola Pablo

Para el ejemplo anterior, la salida en pantalla tiene 15 espacios, y como se puede ver, la línea que tiene un menos (-) está justificada a la izquierda.

El lenguaje C utiliza secuencias de escape para visualizar caracteres que no están representados por símbolos tradicionales. Las secuencias de escape proporcionan flexibilidad en las aplicaciones mediante efectos especiales. A continuación se muestra una tabla conteniendo las secuencias de escape más comunes:

Secuencia de Escape	***Significado***
\a	Alarma
\b	Retroceso de espacio
\f	Avance de Página
\n	Retorno de carro y avance de línea
\r	Retorno de carro
\t	Tabulación
\v	Tabulación Vertical
\\	Barra Inclinada
\?	Signo de Interrogación
\"	Dobles Comillas

Entrada

La entrada de datos a un programa puede tener diversas fuentes como son el teclado, un archivo en disco. La entrada que consideramos ahora es a través del teclado, asociado al archivo estándar de entrada stdin. La función más utilizada, por su versatilidad, para entrada formateada de datos es scanf( ). El archivo de cabecera stdio.h del lenguaje C

proporciona la definición (el prototipo) de scanf( ), así como de las otras funciones de entrada o de salida. La forma general que tiene la función scanf( ) es:

scanf("cadena de control", var1, var2, ...);

Donde:

cadena de control contiene los tipos de datos y, si se desea, su formato
var1, var2, ... son variables que se van a capturar

Los códigos de control de formato más comunes son los mismos utilizados para la salida de datos printf( ).

Por ejemplo:

int n; double x;
scanf("%d %lf", &n, &x);

La entrada de los datos tiene que ser de la forma:

134 -1.4E-4

En este caso la función scanf( ) devuelve n=134, x=-1.4E-4 (en doble precisión). Los argumentos var1, var2, . . . de la función scanf( ) se pasan por dirección o referencia pues van a ser modificados por la función para devolver los datos. Por ello necesitan el operador de dirección, el prefijo &. Un error frecuente se produce al escribir:

scanf("%lf", x);
en vez de: *scanf("%lf", &x);*

A continuación se presentan algunos ejemplo típicos del uso de la función scanf( ):

```
printf("Introduzca v1 y v2: ");
scanf( "%d %f", &v1, &v2 );
printf("Precio de venta al público ");
scanf( "%f", &Precio_venta );
printf("Base y altura: ");
scanf( "%f %f", &b, &h );
```

La función *scanf( )* termina cuando ha captado tantos datos como códigos de control se han especificado, o cuando un dato no coincide con el código de control especificado.

A continuación se presenta un ejemplo de un pequeño programa que imprime en la pantalla “Bienvenido a la Programación en C”, así mismo también como compilarlo y ejecutarlo. Para que puedas observar como funciona, realiza cada uno de los pasos que a continuación se describen:

1.- Iniciar Turbo C++, e introduce el siguiente programa en C en el editor de programas.

```
#include <stdio.h>
int main( )
    {
     printf("Bienvenido a la Programación en C\n");
     return 0;
    }
```


Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

2.- Guarda tu programa con el nombre PRACTI02.CPP

3.- Compila tu programa utilizando el menú compilar, esta instrucción compilará el programa y si no se detecta ningún error se generará un archivo objeto llamado PRACTI02.OBJ y un archivo ejecutable PRACTI02.EXE.
4.- Si el programa tiene algún error hay que corregirlo y volver a compilar, si esta libre de errores entonces ya puedes ejecutar el programa en la opción "RUN" del menú "COMPILE" o alternativamente, tecleando la tecla de función "F5".

ACTIVIDADES:

i) Hacer un programa en C que imprima en la pantalla las iniciales de su nombre, por ejemplo José Luis Sánchez Ramírez, el programa deberá imprimir JLSR utilizando asteriscos (*).
ii) Hacer un programa en C que lea una cantidad cualesquiera de pesos, y que calcule e imprima su equivalente en dólares, libras, euros y francos suizos utilizando formato para 2 decimales, justificación a la derecha de las cantidades y mostrando la información en diferentes renglones, si se sabe que los tipos de cambio son los siguientes: 11.10 pesos por dolar, 16.80 pesos por libra, 15.45 pesos por euro y 21.07 pesos por franco suizo.
iii) Hacer un programa en C para convertir una medida dada en pies a su equivalente en a) yardas; b) pulgadas; c) metros; d) centímetros y e) milímetros, sabiendo que 1 pie = 12 pulgadas, 1 yarda = 3 pies, 1 pulgada = 2.54 cm. Utilizar formato para los valores decimales.
iv) Hacer un programa que calcule el sueldo neto semanal a percibir por un empleado, los datos a capturar son sueldo diario, días trabajados, porcentaje de ispt e imss. El sueldo neto semanal será calculado de la siguiente manera : El sueldo semanal será el sueldo diario por los días trabajados mas el séptimo día proporcional (El cual será calculado en base a la cantidad de días trabajados, esto quiere decir que si el trabajador asistió los 6 días trabajados entonces tiene derecho a un séptimo día de sueldo equivalente a un sueldo diario, de no contar con los 6 días trabajados solo tendrá derecho a la parte proporcional según los días que trabajo), una vez calculado el sueldo semanal entonces se calcularan el ispt y el impuesto del imss, los cuales se descontaran del sueldo semanal para obtener finalmente el sueldo neto semanal del empleado. Utilizar formato de 2 decimales.
v) Hacer un programa en C para obtener la hipotenusa de un triángulo rectángulo, teniendo como datos de entrada los valores de los catetos.
vi) Hacer un programa en C que desglose una cantidad N de minutos, en su equivalente período de tiempo dado en meses, semanas, días, horas, y segundos.
vii) Hacer un programa en C que lea el radio de un círculo y a continuación visualice el área del círculo, el diámetro del círculo y la longitud de la circunferencia del círculo.

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

viii) Hacer un programa en C que lea la masa de 2 cuerpos y la distancia entre ellos y que calcule e imprima la fuerza gravitacional entre ellos, sabiendo que la fuerza de atracción entre dos cuerpos *m1* y *m2*, separadas por una distancia *d* esta dada por la formula $F=(G*m1*m2)/d^2$ donde *G* es la constante de gravitación universal igual a 6.673E-8, el resultado será en dinas.

ix) Hacer un programa en C para calcular e imprimir la distancia que recorrerá un haz de luz en 1 hora, si este viaja a una velocidad de 300,000 km/seg.

x) Hacer un programa en C que lea una temperatura en grados Celsius (Centígrados) y que obtenga su equivalente en grados Fahrenheit, Kelvin y Ranking, éstos se calculan de la siguiente manera, para convertir de Celsius a Fahrenheit se multiplica la temperatura en Celsius por 32 y se multiplica por 9/5, para convertir a Kelvin se le suman 273 grados a la temperatura en Celsius y para convertir a Rankine se le suman 460 a la temperatura en Fahrenheit.

xi) Hacer un programa en C para calcular el área de un triángulo mediante la fórmula: $area = \sqrt{p(p-a)(p-b)(p-c)}$, donde p es el semiperímetro, dado por: $p = (a+b+c)/2$, siendo *a, b* y *c* los tres lados del triangulo.

Todos los alumnos deberán hacer los incisos i y xi, de los incisos ii al x, el profesor seleccionará cual o cuales deberán realizar cada alumno.

CONCLUSIONES:

Los resultados serán solo a nivel visualización, no serán impresos, éstos podrán variar dependiendo de los ejercicios que el alumno realice. Pero debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.

REFERENCIAS

Sugerida

- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009
- Programación en C. Gottfried, Byron. Mc Graw Hill. México 2007, 5ª Edición

PRACTICA 8

ALGORITMOS CON EL BUCLE O CICLO DE REPETICION *"FOR"*

OBJETIVO:

El alumno utilizará el bucle o ciclo de control repetitivo en sus programas de manera adecuada

INTRODUCCIÓN:

Los ciclos sirven para repetir la ejecución de una sentencia o bloque de sentencias, éstas sentencias se ejecutan un número determinado de veces de acuerdo a un valor predefinido o el cumplimiento de una determinada acción o condición. Los ciclos también conocidos como estructuras de control repetitivas, corresponden a unas de las estructuras de control más poderosas en lo que respecta al área de programación, en lenguaje de programación C se cuenta con 3 tipos de ciclos: el ciclo *while*, el ciclo *do_while* y el ciclo *for*, en esta práctica nos centraremos en el primero de estos.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- Compilador de lenguaje C (Turbo C++)

Material:

- Diccionario Inglés-Español
- Lápiz o Bolígrafo (al menos dos colores diferentes)
- Regla o escuadra
- Esta práctica impresa

DESARROLLO:

FUNDAMENTO:
El ciclo for tiene la siguiente sintaxis

```
for (Valor Inicial ; Valor Final ; Incremento o decremento de la Variable)
        {
        (Bloque de Instrucciones);
        }
```

El ciclo for se ejecutará un cierto número de veces especificando desde un valor inicial hasta el valor final. Si el valor final es mayor que el valor inicial significa que el ciclo va en incremento así que llegaremos al valor final sumándole uno o más al valor de la variable

(incremento). Si el valor inicial es mayor al valor final significa que es un ciclo en decremento así que llegaremos al valor final restándole uno o más a la variable (decremento). Si en lugar de un bloque de instrucciones se tiene una sola instrucción podrán omitirse los corchetes { }. Se recomienda su uso cuando se necesita ejecutar un número predeterminado de veces un bloque de instrucciones.

Actividad 1:
En esta actividad se van a escribir distintos bucles *for* para sumar los enteros del 1 al 5. En la forma 1 se dejan en blanco las partes de inicialización y actualización del bucle *for*; el resultado es correcto porque dichas tareas se han incluido de otra forma. La forma 2 es completamente estándar. En la forma 3 todo se ha introducido en el paréntesis del bucle; en la inicialización se da valor inicial tanto a *i* como a *suma*; en la actualización se modifica también tanto *i* como *suma*. El resultado es que el bucle sólo necesita una sentencia vacía, representada por el punto y coma (;). La forma 4 es una variante de la forma 3.

Se debe crear un programa cuyo nombre será PRACT03A.cpp. A continuación se debe copiar el siguiente programa en el editor de código, compilarlo y realizar su ejecución.

```
#include <stdio.h>  // Para función printf()

void main(void)
        {
        int i=1, suma=0;
        for( ; i<=5; )                          // forma 1
                {
                 suma += i;
                 ++i;
                }
        printf("Suma 1 = %d", suma);
        suma=0;
        for(i=1; i<=5; ++i)                     // forma 2
                suma += i;
        printf("Suma 2 = %d", suma);

        for(i=1, suma=0; i<=5; ++i, suma+=i)    // forma 3
                ;
        printf("Suma 3 = %d", suma);
        for(i=1, suma=0; i<=5; suma+=i, ++i)    // forma 4
                ;
        printf("Suma 4 = %d", suma);
        }
```


Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

Después de crear, compilar y de ejecutar este programa, observará que la forma 3 da un resultado diferente de las demás. ¿Sabrías explicar por qué? ¿Trata de corregirla?

Actividad 2:

En esta actividad se va a realizar un menú, desde el cual se pueden escoger distintas opciones. Para ello se va a usar un bucle *for* sin ningún parámetro que actúa como un bucle infinito, esto quiere decir que nunca terminará, por eso se termina el programa por medio de la función *exit(0)* aunque también se puede probar a utilizar la sentencia *break*; para salir del bucle *for*.

Se debe crear un programa cuyo nombre será PRACT03B.cpp. A continuación se debe escribir el siguiente programa en el editor de código, compilarlo y realizar su ejecución.

```
#include <stdio.h>    //para printf(); y scanf();
#include <conio.h>  //para clrscr() y getch();
void main(void)
        {
        int opcion;
        clrscr(); //borra la pantalla al inicio
        for(;;)
                {
                printf("\n\tMi Menu");
                printf("\n\t-------");
                printf("\n\t 1. Opción 1");
                printf("\n\t 2. Opción 2");
                printf("\n\t 3. Opción 3");
                printf("\n\t 4. Opción 4");
                printf("\n\t 5. Salir");
                printf("\n\tSelecciona una opción: ");  //pedimos al usuario que escoja.
                scanf("%d", &opcion);
                //evaluamos la opcion escogida
                if (opcion==1)
                        {
                        printf("\n\tHas elegido la opción 1");
                        }
                else if (opcion==2)
                        {
                        printf("\n\tHas elegido la opción 2");
                        }
                else if (opcion==3)
                        {
                        printf("\n\tHas elegido la opción 3");
                        }
```


Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

```
        else if (opcion==4)
                {
                printf("\n\tHas elegido la opción 4");
                }
        else if (opcion==5)
                {
                printf("\n\tHasta pronto! Oprime cualquier tecla para salir");
                getch(); break; //Salir del ciclo
                }
        else
                {
                printf("\n\tOpción no valida, selecciona otra opción…");
                }
        }
}
```

CONCLUSIONES:

Los resultados serán solo a nivel visualización, no serán impresos, éstos podrán variar dependiendo de los ejercicios que el alumno realice. Pero debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.

REFERENCIAS

Sugerida

- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009
- Programación en C. Gottfried, Byron. Mc Graw Hill. México 2007, 2ª Edición

PRACTICA 9

ALGORITMOS CON ESTRUCTURAS DE DECISIÓN MÚLTIPLE *"SWITCH & CASE"*

OBJETIVO:

El alumno utilizará las estructuras de decisión múltiple en sus programas de manera adecuada

INTRODUCCIÓN:

El lenguaje C tiene incorporada una sentencia de bifurcación múltiple llamada switch, la computadora comprueba una variable sucesivamente frente a una lista de constantes enteras o caracteres. Después de encontrar una coincidencia, la computadora ejecuta la sentencia o bloque de sentencias que se asocian con la constante.

MATERIAL Y EQUIPO A UTILIZAR:

Para la realización de esta práctica son necesarios los siguientes componentes:
Equipo:

- PC

Software:

- Compilador de lenguaje C (Turbo C++)

Material:

- Diccionario Inglés-Español
- Lápiz o Bolígrafo (al menos dos colores diferentes)
- Regla o escuadra
- Esta práctica impresa

DESARROLLO:

FUNDAMENTO:
La forma general de la sentencia switch es:

```
switch(variable)
        {
         case constante1:
         secuencia de sentencias
         break;
         case constante2:
         secuencia de sentencias
         break;
         .
         .
```

```
        .
        default:
        secuencia de sentencias
    }
```

Donde la computadora ejecuta la sentencia default si el valor de la variable no coincide con ninguna constante de la lista. El default es opcional. Cuando el valor de la variable coincide con alguna constante de la lista, la computadora ejecuta las sentencias asociadas con el case hasta encontrar un break.

Otro elemento importante es la instrucción break al final de las instrucciones de cada caso. Es imprescindible, pues de otra manera se seguirían ejecutando el resto de instrucciones que van después hasta encontrar un break, lo cual provocaría una ejecución incorrecta.

También es posible poner el mismo código para diversos casos:

```
switch(expresion)  {
                    case v1 :
                    case v2 :
                    case v3 : instr1_1;
                              ...
                              instr1_N1 ;
                              break;
                  }
```

Ejemplo de sentencia switch:

```
int i;
...
switch(i){
        case 1:     printf("i es uno \n");
                    break;
                    /* si no ponemos el break aquí, continuara */
                    /* ejecutando el código del siguiente case */
        case 2:     printf("i es dos \n");
                    break;
        default:    printf("i no es ni uno ni dos \n");
                    break;  /* para default, el break es opcional */
        }
```

ACTIVIDAD:

Realiza nuevamente el programa de Menú utilizando la sentencia switch(); (el que se hizo con ciclo *for* infinito)

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

II Estructuras de repetición

El lenguaje C también aporta estructuras para la repetición de una o varias instrucciones (bucles). El bucle fundamental es el while, con la siguiente sintaxis:

while (expresion) instruccion;

O para múltiples instrucciones:

```
while (expresion) {
                instruccion1;
                instruccion2;
                ...
                instruccionN;
                }
```

Con esta estructura se ejecutará la instrucción o el bloque mientras la expresión se evalúe a cierto. Existe una estructura similar, el do while, que hace la comprobación de continuidad al final en vez de al principio, y que mantiene la siguiente sintaxis:

```
do instruccion;
while (expresion);
```

O para múltiples instrucciones:

```
do          {
   instruccion1;
   instruccion2;
   ...
   instruccionN;
   } while (expresion);
```

La diferencia es que con esta última estructura siempre se ejecuta al menos una vez la instrucción o bloque asociados. De nuevo, la ejecución se repetirá mientras la expresión se evalúe a verdadero.

Ejemplos de utilización:

```
while (i<30)      { /* test al principio del bucle */
                printf();
                ....
                }
```


Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

```
....
      do                {
                printf();
                ....
                } while (i<30); /* test al final del bucle */
....
```

ACTIVIDAD: Realiza nuevamente el programa de suma de números (1-100), pero utilizando la sentencia do while();

CONCLUSIONES:

Los resultados serán solo a nivel visualización, no serán impresos, éstos podrán variar dependiendo de los ejercicios que el alumno realice. Pero debe expresar sus conclusiones personales al finalizar la práctica. Para lo cual puede utilizar el siguiente espacio o el reverso de las hojas.

REFERENCIAS

Sugerida

- Lenguaje C. Ceballos, Francisco Javier. RA-MA Addison. EUA 2009
- Programación en C. Gottfried, Byron. Mc Graw Hill. México 2007, 5ª Edición

Centro Universitario UAEM Valle de Chalco
Av. Hermenegildo Galeana No 3, Col. Ma. Isabel, Valle de Chalco, C.P. 56615,
Edo. De México, Tel: (55) 59714940, 59787577 y 30921763
Página: http://cux.uaemex.mx e-mail: rgcruzf@uaemex.mx

REFERENCIAS GENERALES

- Estévez, N. E. H., Enseñar a Aprender, Ed. Maestros y Enseñaza Paidós, 2002.
- Rugarcía, T. A., et. al., El futuro de la educación en ingeniería, Universidad Iberoamericana, 2001.
- Cooper, J. M., Estrategias de enseñanza, Ed. LIMUSA, 2003.
- Albarrán Trujillo S.E, Salgado Gallegos M, Programación Básica, UAEM, 2000
- Gottfried, Byron, Programación en C, 5a. Edición, Mc Graw Hill, México, 2007.
- Joyanes, Luis, Fundamentos de programación. Algoritmos y estructura de datos, McGraw-Hill, México, 2002.
- Balcázar, José Luis, Programación metódica, McGraw-Hill, España, 2003.
- Ceballos, Francisco Javier, Lenguaje C, RA-MA Addison, E.U.A, 2009.
- Heileman, Gregory, Algorithmics, the Spirit of Computing, Addison Wesley, Massachusetts, 2005.
- Joyanes, Aguilar Luis y Zahonero, Martínez I., Programación en C, Metodología, Estructura de Datos y Objetos, McGraw Hill, México, 2001.