Clase 1

Lenjuaje C--- compilacion y enlasado. JS es interpretado  
Es un ejecutable a traves del sistema operativo  
  
  
Editor  
Compilador  
Linkeador  
  
El compilador nos informa los errores del código, traduce de codigo fuente a binario  
linkeador: agarra el código binario y el de todas las bibliotecas para enlazarlas. Da como salido un ejecutable.  
  
  
funcion mostrar= int main (void)  
  
operadores igual q JS:  
Aritméticos: \* / + - %  
Relacionales: == != < >  
Lógicos: ! && ||  
  
TERMINA CON ;  
  
Hay q poner q tipo de dato y declararlo si o si:  
int: Entero  
char: Caracter   
float: Flotante (numeros reales, 16 decimales en gral)  
double: flotante doble (doble de decimales, mas preciso, 32 decimales en gral.)  
short int: soporta desde -32 mil y pico hasta 32 mil y pico (reduce el tamaño pero soporta menos numeros)  
long int:   
Estas son palabras reservadas  
  
  
VARIABLES SE DECLARAN AL PRINCIPIO SI O SI!!!

Clase 2

Do While:  
Siempre entra una vez.  
  
int i=0;  
do{  
-  
-  
i++;  
}while(i<10); //1ro hacer, luego evaluar el while.  
  
Cuando se la cant de repeticiones conviene FOR, si no la conosco conviene WHILE o DO WHILE.  
  
EJEMPLO:  
int edad;  
printf("ingrese edad: ")  
scanf("%d", &edad);  
while(edad<0){  
----  
}  
  
lo mismo con DO WHILE:  
do{  
printf("ingrese edad: ")  
scanf("%d", &edad);  
}while(edad<0);

Clase3

FUNCIONES (bajar git bash)  
  
preguntas basicas para desarrollar una funcion:  
Que hace??  
Que necesita/recibe??  
Que devuelve??  
  
Main: es el inicio de ejecución de las funciones. Recibe lo q esta entre parentesis, por ahora esta vacio, no recibe nada. Devuelve un entero (int) (return 0).  
  
Los parámetros q recibe una función son delimitados por comas ","  
  
Pasaje por valor: la función recibe una copia.  
  
Pasaje por referencia: la funcion recibe la direccion de memoria no recibe un valor, porner el "&" antes de la variable.

Clase 4

Las bibliotecas tienen archivos .h    .c     y el main.c   donde codificamos

.h   ------> head (cabeza)

ahora el prototipo de función lo vamos a poner en el .h y el cuerpo de la función en el .c

estos dos archivos DEBEN llamarse iguales!!!!

// #include "./calculo.h" ----> ruta relativa (es lo mismo que arriba, va a buscarlo dentro del proyecto.

// #include "../calculo.h" ----> en este caso con el "../" vamos una carpeta hacia atras, para ir 2 carpetas atras: "../../calculo.h"

// ""D:/GOMEZ NICOLAS RODRIGO/clase 4/calculo.h" ------> ruta absoluta

Recursividad:

//RECURSIVIDAD: APLICADA CON FACTOREO

/\*

#include <stdio.h>

int factorial(int );

void main(void)

{

int valor,result;

printf("\nIngrese numero:");

scanf("%d",&valor);

result=factorial(valor);

printf("\nEl factorial de %d es %d",valor,result);

}

int factorial(int n)

{

int resp;

if(n==1)

{

return 1;

}

resp=n\* factorial(n-1);

return (resp);

}

\*/

getch() ---> obtiene una letra por teclado (similar a scan).

getche()---> lo mismo pero muestra la letra tecleada.

Clase 5

Arrays (se conocen como sinonimos de Vetor o Arreglado)

int vec [5] ----> se guardan 5 numeros de caracter entero en distintos directorios (int) de 0 a k-1, siendo "k" en este caso "5"

vec ----> es solo en nombre, se puede poner cualquier nombre en esta variable.

con estas pocas lineas se pueden cargar una cant ilimitada de datos:

jnt i;

int edad [40]

for(i=0; i<40; i++)

{

printf("edad: ");

scanf("%d", &edad[i];

}

------------------------------------

#define CANT 40 ---->declaro constante (conviene ponerlo el nombre en mayuscula por costumbre profesional.

for(i=0; i<cant; i++)

{

printf("%d", edad[i];

}

Clase 6

Cadenas (String)

char nombre[31];

0 1 2 3 4 ... 30

'j' 'u' 'a' 'n' '\0''.....' el ultimo es un solo caracter para darle fin y q luego no imprima nada q haya estado almacenado

el '\0' se escribe automaticamente, no se debe escribir.

printf("%s", nombre);

(

#define ESC 27 ----->constante

char tecla;

do{

----

----

----

tecla=getch(); ----> espera que aprietes una tecla

}while(tecla!=ESC) ---> o tecla!=27----> en caso de no definir la constante

)

char nombre[31];

printf("Ingrese nombre: ");

fflush(stdin); --->limpia

scanf("%s", nombre); ----> no lleva '&' ya q el nombre representa la direccion del 1er elemento del vector es igual a "&nombre[0]"

LOS OPERADORES RELACIONALES NO SIRVEN PARA CADENAS ( == != < > >= <= )

SE UTILIZA LA SIGUIENTE FUNCION:

strcmp (cadena1, cadena dos) -----> devuelve un INT

si son iguales devuelve = 0

si la 1er cadena es mayor devuelve >0

si la 2da cadena es mayor devuelve <0

vemos reflejado en funcionamiento como son comparados los vectores

compara el 1er elemento si son distintos deja de coparar.

strupc(); ---> como toupper(); pasa a mayuscula... el 1ro para vectores, el 2do para elementos.

strlwr(); ---> como tolower(); pasa a miniscula... el 1ro para vectores, el 2do para elementos.

ARRAYS DE CADENAS... PARA CARGAR VARIOS NOMBRES

char nombre [CANT][31] ------> en este caso en CANT especificamos la cantidad de nombres y en el 31 la cantidad de elementos de cada nombre.

Clase 7

strcmp(cadena1, cadena2) compara cadenas

strcmp(cadena1, cadena2) compara cadenas sin tener en cuenta minisculas y mayusculas

strupr() pone todo en mayuscula

strwr() poneen minusculas

para estas funciones: es necesaria la biblioteca---> string.h

strcat(destino, origen); copia la cadena "origen" a la cadena destino, si en destino hay un hola y en origen hay un chau.. la cadena destino queda: holachau.

esta funcion concatena lo del origen en el destino.

HECHO POR PROFESOR, ORDENAMIENTO CON DOS CRITERIOS:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX 5

int main(void){

int nota[MAX], notaAux, i, j;

char nombre[MAX][31], nombreAux[31];

for(i=0; i<MAX; i++){

system("cls");

printf("Ingrese el nota: ");

scanf("%d", &nota[i]);

printf("Ingrese el nombre: ");

fflush(stdin);

gets(nombre[i]);

}

for(i=0; i<MAX-1; i++){

for(j=i+1; j<MAX; j++){

if(nota[i] < nota[j]){

strcpy(nombreAux, nombre[i]);

strcpy(nombre[i], nombre[j]);

strcpy(nombre[j], nombreAux);

notaAux = nota[i];

nota[i] = nota[j];

nota[j] = notaAux;

}

if(nota[i] == nota[j]){

if(strcmp(nombre[i],nombre[j]) > 0){

strcpy(nombreAux, nombre[i]);

strcpy(nombre[i], nombre[j]);

strcpy(nombre[j], nombreAux);

//notaAux = nota[i];

//nota[i] = nota[j];

//nota[j] = notaAux;

}

}

}

}

system("cls");

printf("\nnota\tNombre");

for(i=0; i<MAX; i++){

printf("\n%d\t%s", nota[i], nombre[i]);

}

return 0;

}

\*/

Clase 8

ESTRUCTURAS

Crear tipos de datos porpios en funcion de los tipos de datos primitivos que ya conocemos (int, float, char, double, void)

struct alumno alu; ----> "struc alumno es el tipo de dato creado... "alu" es el nombre de la variable con ese tipo

definir ese tipo de dato: ------> esto se declara en las bibliotecas .h, junto con los prototipos de las funciones

struct alumno{

int legajo;

char nombre [31];

etc, etc;

}

struct alumno alu;

printf("legajo: ");

scanf("%d", &alu.legajo); ---> el %d va xq legajo es de ese tipo

printf("Nombre: ");

fflush(stdin);

scanf("%[^\n]", alu.nombre);

\*/

/\*

VALIDACIONES!!!!!!!!!!!!!!

isalpha ----> funcion q devuelve 0 si no es una letra o 1 si es una letra !!

isdigit ----> funcion q devuleve 0 si no es un numero o 1 si es un numero !!

usar getch ---> para no mostrar el caracter q se escribe

Clase 10

MATRICES

el nombre de la matris es igual a la posicion 0 de la matris

int mat[2][3];

mat + z ----> donde z es igual al tipo de dato, al ser un entero son 4 bytes

sizeof() ---> poner una variaoble o un tipo de dato

float matriz[2][3];

int i,j;

for(i=0;i<2;i++)

{

for(j=0;j<3;j++)

{

printf("M[%d][%d]: ",i,j);

scanf("%f",&matriz[i][j]);

}

}