ESTRUCTURAS

43) REALIZAR UNA ESTRUCTURA Q' PIDA LOS DATOS DE UN ALUMNO.

```
STRUCT ALUMNO {
  INT NOTA;
  INT EDAD;
  CHAR NOMBRE[15];
};// SIEMPRE PONGA LA LLAVE DE CIERRE;

INT MAIN(){
  STRUCT ALUMNO F1; // ALUMNO PASA A LLAMARSE F1
  F1.EDAD = 21; // LLEGO A LA ESTRUCTURA INTERNA Y DE DOY UN VALOR
  F1.NOTA = 7;

  CHAR NOMBRE_AUX[] = "LOLANDO MEDRANO";
  STRCPY(F1.NOMBRE, NOMBRE_AUX);
  PRINTF("EL NUMBRE : %S, EDAD: %D TIENE UNA NOTA DE: %D", F1.NOMBRE, F1.EDAD, F1.NOTA);
  RETURN 0;
}
```

44) RALIZAR UNA ESTRUCTURA Q' ME PIDA UN CONTACTO CON ALGUNOS DATOS

```
STRUCT CONTACTO {
  CHAR NOMBRE[20];
  CHAR DIRECCION[60];
  INT EDAD;
  INT TELEFONO;
};
TYPEDEF STRUCT CONTACTO CONTACTO_AUX; // CAMBIO LA VARIABLE A OTRO NOMBRE PARA SIEMPRE
INT MAIN(){
    CONTACTO_AUX F1;
    PRINTF("INGRESE NOMBRE");
    GETS(F1.NOMBRE); // OBTENGO DIRECTAMENTE LO Q' ESTA EN LA ESTRUCTURA
    PRINTF("INGRESE EDAD");
    SCANF("%D", &F1.EDAD);
    PRINTF("NOMBRE : %S, EDAD: %D", F1.NOMBRE, F1.EDAD);
RETURN 0;
}
```

45) REALIZE UNA ESTRUCTURA QUE PERMITA ALMACENAR VIDEOS CON SUS RESPECTIVAS VISITAS Y TIEMPO.

```
TYPEDEF STRUCT VIDEO {
    CHAR TITULO[40];
    INT VISITAS;
    FLOAT TIEMPO;
} VIDEO_AUX; // LA VARIABLE (NUEVA) DE LA ESTRUCTURA

VIDEO_AUX CREAR_VIDEO() {
    VIDEO_AUX NUEVO_VIDEO; // NUEVO_VIDEO VARIABLE DE LA FUNCIÓN (LOCAL)
    PRINTF("INGRESE TITULO: ");
    GETS(NUEVO_VIDEO.TITULO);
    PRINTF("INGRESE EL Nº DE VISITAS: ");
    SCANF("%d", &NUEVO_VIDEO.VISITAS);
    PRINTF("INGRESE EL TIEMPO DEL VIDEO: ");
    SCANF("%F", &NUEVO_VIDEO.TIEMPO);
    RETURN (NUEVO_VIDEO);
}
INT MAIN() {
    VIDEO_AUX V1 = CREAR_VIDEO(); // ESTO SE REALIZA EN LA MAIN LA FUNCIÓN
    IMPRIMIR_VIDEO(V1);
RETURN 0;
VOID IMPRIMIR_VIDEO (VIDEO_AUX VIDEO) {
    PRINTF("TITULO: %S, TIEMPO: %.2, VISITAS: %.0", VIDEO.TITULO, VIDEO.TIEMPO, VIDEO.VISITAS);
}
```

46) REALICE UNA ESTRUCTURA AVANZADA GUARDE LOS DATOS DE UNA PERSONA Y LA FECHA ESTE EN UNA ESTRUCTURA APARTE.

```
TYPEDEF STRUCT FECHA {
    INT DIA;
    INT MES;
    INT AÑO;
} FECHA_X;

TYPEDEF STRUCT CONTACTO {
    CHAR NOMBRE[40];
    LONG TELEFONO;
    FECHA_X FECHA_NACIMIENTO; // UNA NUEVA ESTRUCTURA (AVANZADA)
} CONTACTO_T;

FECHA_X CREAR_FECHA() {
    FECHA_X NUEVA_FECHA;
    PRINTF("INGRESE EL DIA: \n");
    SCANF("%d", &NUEVA_FECHA.DIA);
    FFLUSH(STDIN);
    PRINTF("INGRESE EL MES: \n");
    SCANF("%d", &NUEVA_FECHA.MES);
    FFLUSH(STDIN);
```

⊛

```
    PRINTF("INGRESE EL AÑO: \n");
    SCANF("%d", &NUEVA_FECHA.AÑO);
    FFLUSH(STDIN);
    RETURN (NUEVA_FECHA);
}
```

```
CONTACTO_T CREAR_CONTACTO(){
  CONTACTO_T NUEVO_CONTACTO;
  PRINTF("INGRESAR EL NOMBRE: ");
  GETS(NUEVO_CONTACTO.NOMBRE);
  PRINTF("\n");
  PRINTF("INGRESAR EL TELEFONO: ");
  SCANF("%L", &NUEVO_CONTACTO.TELEFONO);
  NUEVO_CONTACTO.FECHA_NACIMIENTO = CREAR_FECHA(); //ASIGNO
  RETURN(NUEVO_CONTACTO);
}
```

```
INT MAIN(){
  CONTACTO_T C1;
  C1 = CREAR_CONTACTO();
  RETURN 0;
}
```

ORDENAMIENTO.

47) REALIZAR UN ORDENAMIENTO BURBUJA DONDE TENGA UN VECTOR DONDE SE ORDENE DE MENOR A MAYOR Y MAYOR A MENOR MOSTRANDO AMBAS POR PANTALLA

```
INT ARRAY[5] = {3,5,2,1,4}; //5 NUMEROS ENTEROS
INT I,J,AUX; // 3 VARIABLES
```

// SIGUIENTE (CAMBIO) BURBUJA

```
FOR(I=0; I<5; I++){
  FOR(J=0; J<5; J++){
    IF(ARRAY[J] > ARRAY[J+1]){ //SI EL Nº ES MAYOR AL SIGUIENTE
      AUX = ARRAY[J]; //CONFIRMA
      ARRAY[J] = ARRAY[J+1]; //SE PARA EL CAMBIO
      ARRAY[J+1] = AUX; //SE TERMINA (CAMBIO) AUX
    }
  }
}
FOR(I=0; I<5; I++){ //F. ASCENDENTE
  PRINTF("%I", ARRAY[I]);
}
FOR(I=4; I>=0; I--){ //F. DESCENTE
  PRINTF("%I", ARRAY[I]);
}
RETURN 0;
```

49) REALIZAR UN ORDENAMIENTO SELECCIÓN DONDE TENGA UN VECTOR Q' ME ORDENE DE FORMA ASCENDENTE Y DESCENDENTE CON 5 NÚMEROS.

```
INT A[5] = {2,4,1,5,3}; //SELECCIONA Nº POR Nº Y LO COLOCA
INT I,J,AUX,MIN;
```

ORDENADAMENTE (BUSCA EL 1ro DE SELECCIÓN LA LISTA Y LO CAMBIA POR EL SIGUIENTE

```
FOR(I=1; I<5; I++){
  MIN = I; // PRIMER POSICIÓN DE LA LISTA (3)
  FOR(J=I+1; J<=5; J++){ // PARA Q' RECORRA LOS Nº RESTANTES
    IF(A[I] < A[MIN]){ // COMPRUEBA SI EL MIN (3) ES EL MAS CHICO DE LA LISA
      MIN = J; //EL RESTO DE LAS POSICIONES LO IGUALA AL MÁS CHICO
    }
  }
  AUX = A[I];
  A[I] = A[MIN];
  A[MIN] = AUX;
}
PRINTF("ASCENDENTE");
FOR(I=1; I<=5; I++){
  PRINTF("%I", A[I]);
}
PRINTF("DESCENDENTE");
FOR(I=4; I>=0; I--){
  PRINTF("%I", A[I]);
}
RETURN 0;
```

5) HACER 2 ESTRUCTURAS, UNA LLAMADA PROMEDIO Q' TENDRÁ LOS SIGUIENTES MIEMBROS:
NOTA 1, NOTA 2, NOTA 3; Y OTRA LLAMADA ALUMNO Q' TENDRÁ LOS SIGUIENTES MIEMBROS:
NOMBRE, SEXO, EDAD, HACER Q' LA ESTRUCTURA PROMEDIO ESTÉ ANIDADA EN LA ESTRUCTURA ALUMNO,
LUEGO PEDIR TODOS LOS DATOS PARA UN ALUMNO, LUEGO CALCULAR SU PROMEDIO, Y POR ÚLTIMO
IMPRIMIR TODOS SUS DATOS INCLUIDO EL PROMEDIO.

```
STRUCT PROMEDIO {
    FLOAT NOTA1;
    FLOAT NOTA2;
    FLOAT NOTA3;
};
STRUCT ALUMNO {
    CHAR NOMBRE[20];
    CHAR SEXO[20];
    INT EDAD;
    STRUCT PROMEDIO PROM;
} ALUMNO1;
INT MAIN() { FLOAT PROMEDIO=0;
PRINTF("INGRESE SU NOMBRE: ");
GETS(ALUMNO1.NOMBRE);
PRINTF("INGRESE SU SEXO: ");
GETS(ALUMNO1.SEXO);
PRINTF("INGRESE SU EDAD: ");
SCANF("%I", &ALUMNO1.EDAD);
PRINTF("INGRESE LAS NOTAS: ");
SCANF("%F %F %F", &ALUMNO1.PROM.NOTA1, &ALUMNO1.PROM.NOTA2, &ALUMNO1.PROM.NOTA3);

PROMEDIO=(ALUMNO1.PROM.NOTA1 + ALUMNO1.PROM.NOTA2 + ALUMNO1.PROM.NOTA3)/3;
PRINTF("DATOS DEL ALUMNO");
PRINTF("NOMBRE: %S", ALUMNO1.NOMBRE);
PRINTF("SEXO: %S", ALUMNO1.SEXO);
PRINTF("EDAD: %I", ALUMNO1.EDAD);
PRINTF("\n");
PRINTF("PROMEDIO: %.2F", PROMEDIO);

RETURN 0;
}
```

53) Realizar un ordenamiento por inserción un vector de 5 posiciones de forma ascendente y descendente.

```
CHAR a[5] = {'e','i','a','o','u'};   → válido con carácter
INT a[5] = {4, 2, 5, 1, 3};   // si nº sig > nº actual (cambio)
                               // si una parte está ordenada y la otra desordenada, ordena desordenados para el principio.
INT I, POS, AUX; // 3 variables
FOR (I = 0; I < 5; I++){
    POS = I; // número q' guardamos.
    AUX = a[I]; // guarda el valor dentro del vector
    WHILE ((POS>0) && (AUX < a[POS-1])) { // mientras que la posición sea de 1 para adelante y que el número q' está a la izquierda sea mayor al aux.
        a[POS] = a[POS-1]; // intercambiamos el nº dentro en el pos.
        POS--;
    }
    a[POS] = AUX;
}
PRINTF ("ASCENDENTE");
FOR (I=0; I<5; I++){
    PRINTF ("%I", a[I]);
}
PRINTF ("DESCENDENTE");
FOR (I=4; I>=0; I--){
    PRINTF ("%I", a[I]);
}
RETURN 0;
```

INSERCIÓN

54) Cargar vector con 50 números no ordenados, pedir un número, buscar en forma binaria y dar la posición donde está o un mensaje no encontrado.

```
INT C[5] = {1, 5, 8, -5, -10, 2, 6, ..., 50};
INT I, POS, AUX, DATO, INF, SUP, MITAD;
CHAR BAND = 'F';
FOR (I=0, I<5; I++){
    POS = I;
    AUX = C[I];
    WHILE ((POS>0) && (AUX < C[POS-1])) {
        C[POS] = C[POS-1];
        POS--;
    }
    C[POS] = AUX;
}
FOR (I=0; I<50; I++){
    PRINTF ("%I", C[I]);
}
PRINTF ("DIGITE UN NÚMERO");
SCANF ("%I", &DATO);

INF = 0;
SUP = 50;
(*)

(*) WHILE (INF <= SUP){
        MITAD = (INF+SUP)/2;
    IF (C[MITAD] == DATO){
        BAND = 'V';
        BREAK;
    }
    IF (C[MITAD] > DATO){
        SUP = MITAD;
        MITAD = (INF+SUP)/2;
    }
    IF (C[MITAD] < DATO){
        INF = MITAD;
        MITAD = (INF+SUP)/2;
    }
}
IF (BAND == 'F'){
    PRINTF ("NO EXISTE EL Nº")
}
ELSE {
    IF (BAND == 'V'){
    PRINTF ("EL Nº EXISTE, POSICIÓN: %I", MITAD);
    }
}
RETURN 0;
```

Notas:

55) HACER UN ARRAY DE 10 NÚMEROS DESORDENADOS, LUEGO ORDENARLOS CON EL MÉTODO BURBUJA, POSTERIORMENTE PEDIR UN DATO A BUSCAR Y UTILIZAR LA BÚSQUEDA BINARIA PARA SABER SI EXISTE O NO.

```
INT a[10] = {3, 6, 1, 2, 7, 9, 4, 8, 10, 5};
INT I, J, AUX, DATO, INF, SUP, MITAD;    CHAR BAND = 'F';
FOR(I=0; I<10; I++){
  FOR(J=0; J<10; J++){
    IF(a[J] > a[J+1]){
      AUX = a[J];
      a[J] = a[J+1];
      a[J+1] = AUX;
    }
  }
}
FOR(I=0; I<10; I++){
  PRINTF("%I", a[I]); //DE 0 A 10 ME ORDENA TODOS LOS NÚMEROS MOSTRÁNDOLOS EN PANTALLA.
}
PRINTF("DIGITE UN NÚMERO");
SCANF("%I", &DATO); // VOY A GUARDAR UN NÚMERO EN PARTICULAR
//BUSQUEDA BINARIA
INF = 0;
SUP = 10;
WHILE (INF <= SUP) {
  MITAD = (INF + SUP)/2;
IF(a[MITAD] == DATO){
  BAND = 'V';
  BREAK;
}
*
```

M. BURBUJA

```
* IF(a[MITAD] > DATO){
    SUP = MITAD;
    MITAD = (INF + SUP)/2;
  }
  IF(a[MITAD] < DATO){
    INF = MITAD;
    MITAD = (INF + SUP)/2;
  }
}
IF(BAND == 'F'){
  PRINTF("NO EXISTE EL Nº")
}
ELSE IF BAND == 'V'){
PRINTF("EL Nº EXISTE, EN LA POS...
}
RETURN 0;
```

56) CARGAR VECTOR CON 100 VALORES ORDENADOS, PEDIR UN NÚMERO, BUSCARLOS EN FORMA BINARIA Y DAR LA POSICIÓN DONDE ESTÁ O UN MENSAJE NO ENCONTRADO.

```
INT B[100] = {1, 5, 8, 11, 13, 10 .. 100};
INT I, J, AUX, DATO, INF, SUP, MITAD;   CHAR BAND = 'F';
FOR(I=0; I<=100; I++){
  FOR(J=0; J<=100; J++){
  IF(B[J] > B[J+1]){
      AUX = B[J];
      B[J] = B[J+1];
      B[J+1] = AUX;
  }
 }
}
FOR(I=0, I<100; I++){
  PRINTF("%I", a[I]);
}
PRINTF("DIGITE UN NÚMERO");
SCANF("%I", &DATO);
```

Lógica Multiplicativa

```
INF = 0;
SUP = 4;
WHILE (INF <= SUP){
    MITAD = (INF + SUP)/2;
    IF (B[MITAD] == DATO){
        BAND = 'V';
        BREAK;
    }
    IF (B[MITAD] > DATO){
        SUP = MITAD;
        MITAD = (INF + SUP)/2;
    }
    IF (B[MITAD] < DATO){
        INF = MITAD;
        MITAD = (INF + SUP)/2;
    }
}
IF (BAND == 'F'){
    PRINTF("NO EXISTE EL Nº");
}
ELSE IF (BAND == 'V'){
    PRINTF("EL Nº EXISTE, EN LA POS: %I", MITAD);
}
RETURN 0;
```

57) REALIZAR UNA BÚSQUEDA BINARIA CON UN VECTOR DE 5 POSICIONES LA LISTA DEBE ESTAR SI O SI ORDENADA. Y VER SI ENCUENTRO UN Nº PARTICULAR Y QUE ME INDIQUE SI LA BANDERA CAMBIA O NO.

```
INT V[5] = {1,2,3,4,5};
INT INF, SUP, MITAD, DATO;
CHAR BAND = 'F'; // UNA BANDERA PARA SABER SI ENCUENTRO O NO EL DATO

DATO = 3; // VERÉ SI EL Nº = PERTENECE O NO A MI LISTA DE NÚMEROS
// BUSQUEDA BINARIA
INF = 0;
SUP = 5;
WHILE (INF <= SUP){ // (3,5)
    MITAD = (INF + SUP)/2;
    IF (V[MITAD] == DATO){ // SI ENCUENTRA EL DATO SOLICITADO (3).
        BAND = 'V'; // LA BANDERA CAMBIARÁ DE VALOR.
        BREAK; // PORQUE YA ENCONTRÓ EL NÚMERO SOLICITADO DEL BUCLE WHILE.
    } // EN CASO DE Q' NO ENCONTRARA EL DATO.
    IF (V[MITAD] > DATO){ // SI ES SUPERIOR AL DATO (>3)
        SUP = MITAD; // DEBO USAR EL SUPERIOR.
        MITAD = (INF + SUP)/2; // CAMBIA NUEVAMENTE SU VALOR. (1,25)
    }
    IF (V[MITAD] < DATO){ // CASO CONTRARIO (MENOR A 3)
        INF = MITAD; // USO EL INFERIOR.
        MITAD = (INF + SUP)/2; // CAMBIA OTRA VEZ EL VALOR (3,75)
    }
}
IF (BAND == 'F'){ // SI NO CAMBIO DE VALOR LA BANDERA
```

```
PRINTF ("VALOR NO ENCONTRADO");
}ELSE {
  IF ( BAND == 'V') {// SI LA BANDERA CAMBIO DE VALOR
  PRINTF("EL NÚMERO EXISTE, EN LA POS: %d", MITAD);
}
  RETURN 0;
```

5.) DAR EL SUBTOTAL DE LAS PROVINCIAS Y EL TOTAL SIN QUE NOS MUESTRE LOS DETALLES (USAR ESTRUCTURAS)

PROVINCIA	IMPORTE	
A	200	
A	100	
A	50	← 350
B	87	
B	300	← 387
C	500	
C	200	← 700
	TOTAL 1550	

PROVINCIA 1
TOTAL XX.
PROVINCIA 2
TOTAL XX.
TOTAL GENERAL XX.

5.) REALIZE UNA RECURSIVIDAD CON CASO BASE Y RECURSIVO CON UN NÚMERO INTRODUCIDO POR PANTALLA

```
LONG FACTORIAL (INT N); //FUNCIÓN RECURSIVA
INT MAIN() {
  INT NÚMERO;
  PRINTF ("DIGITE UN NÚMERO: ");
  SCANF ("%I", &NÚMERO);
  PRINTF ("\n EL FACTORIAL DEL NÚMERO ES: %Li", FACTORIAL (NUMERO)); // LLEGA PRIMERO EL DATO Q' INTRODUCIMOS Y LLEGA HASTA EL DATO BASE (1!)
  RETURN 0;
}
LONG FACTORIAL (INT N){ //FACTORIAL 4! = 4x3! -> 4.3.2.1
  IF (NC = 1){//1! SERÁ NUESTRA BASE
    RETURN 1;//RETORNO EL VALOR 1 -> 1! = 1
  }
  ELSE{
    RETURN (N* FACTORIAL (N-1)); // LLAMAMOS AL CASO CONTRARIO SI TODO! ES MAYOR A 1!
  }
}
```

RECURSIVIDAD.
ES LA FUNCIÓN Q' SE LLAMA A SÍ MISMA.

6.) REALIZE UNA RECURSIVIDAD CON UN NÚMERO EN PARTICULAR EN UN MENÚ SIMPLE Y LLAMADO AL MENÚ UNA VEZ Q' SE INGRESE EL NÚMERO DE TABLA

```
INT MAIN() {
  INT OPCIÓN = 0;
  WHILE (OPCIÓN != 0) {
    SWITCH (OPCIÓN) {
      CASE 1: ...
      CASE 2: ...
      CASE 3: ...
      CASE 0: "REPETIR" -> ESTO ES UN PRINTF
        MAIN (); -> RECURSIVIDAD
```