Universidad Autónoma de Baja California
Facultad de Ingeniería Arquitectura y Diseño

Ingeniero en Software y Tecnologías Emergentes.

Programación estructurada

ARCHIVOS BINARIOS
(Archivos indexados)

Actividad 14

Danna Guadalupe Sandez Islas

373080

1.- Agregar :

El programa deberá ser capaz de agregar un registro al arreglo de índices y al final del archivo Binario. (agregar forma automatica no repetido el campo llave)

```
case 1: // agregar
    if ((i + 1) <= N)
    {
        temp = gen_per();
        while (busq_opt(indice, i, temp.enrollment, ord) != -1)
        {
            temp.enrollment = numAleatorio(300000, 399999);
        }
        registros[i] = temp; // agregado a el vector de registro
        indice[i].noempleado = registros[i].enrollment;
        indice[i].indice = i;
        agregar(registros, i); // agregar al binario
        i++;
    }
    printf("Persona agregada exitosamente!\n");
    ord = 0;
```

```
void agregar(Tdatos registro[], int i)
{
    FILE *fa;
    fa = fopen("datos.dat", "ab");
    if (fa)
    {
        fwrite(&registro[i], sizeof(Tdatos), 1, fa);
        fclose(fa);
    }
    else
    {
        printf("Error al agregar datos al archivo\n");
    }
}
```

2.- Eliminar :

El programa deberá buscar una **noempleado** en el vector de índices por medio del método de búsqueda más óptimo.

La función deberá *retornar*, el índice donde se encuentra la matrícula en el archivo Binario, utilizar banderas para escoger el método más adecuado.

Una vez obtenido el índice moverse dentro del archivo binario (usar fseek) usando el índice del vector de índices.

Leer el registro en la posición correcta, preguntar si se quiere eliminar registro.

Cambiar el status del registro si la respuesta es afirmativa, volver a posición anterior y sobreescribir el registro.

```c
void eliminar(Tdatos registros[], Tindice index[], int i, int ord)
{
    int mat, enc, op = 0;
    FILE *fa;
    mat = validar("Matricula a buscar para eliminar: ", 300000, 399999);
    enc = busq_opt(index, i, mat, ord);
    if (enc != -1)
    {
        printf("Nombre: %s\nApellidos: %s %s\nEdad: %d\nSexo: %s\nEstado: %s\nPuesto: %s\nMatricula: %d\nCelular: %d\n\n", registros[enc].name, registros[enc].Last
        op = validar("Desea eliminar el registro?\n1-Si\n2-No\nIngrese una opcion: ", 1, 2);
        system("CLS");
        if (op == 1)
        {
            fa = fopen("datos.dat", "r+b");
            if (fa)
            {
                rewind(fa);
                fseek(fa, sizeof(Tdatos) * index[enc].indice, SEEK_SET);
                fread(&registros[index[enc].indice], sizeof(Tdatos), 1, fa);
                registros[enc].status = 0;
                fseek(fa, sizeof(Tdatos) * index[enc].indice, SEEK_SET);
                fwrite(&registros, sizeof(Tdatos), 1, fa);
                printf("Registro eliminado\n");
                fclose(fa);
            }
            else
            {
                printf("Problemas para abrir el archivo\n");
            }
        }
        else
        {
            printf("El registro no se elimino\n");
        }
    }
    else
    {
```

```
Matricula a buscar para eliminar: 300275
Nombre: LAURA
Apellidos: RUIZ RUIZ
Edad: 32
Sexo: MUJER
Estado: AG
Puesto: ING. DE SISTEMAS
Matricula: 300275
Celular: 1009587

Desea eliminar el registro?
1-Si
2-No
Ingrese una opcion:
```

3.- Buscar :

El programa deberá buscar un **noempleado** en el vector de índices por medio del método de búsqueda más óptimo.

La función deberá *retornar*, el índice donde se encuentra la matrícula en el archivo Binario, utilizar banderas para escoger el método más adecuado.

Una vez obtenido el índice moverse dentro del archivo binario (usar fseek) usando el índice del vector de índices.

Leer el registro en la posición correcta, y desplegar el registro.

```
void buscar(Tindice index[], Tdatos registros[], int i, int ord)
{
    int mat, enc;
    FILE *fa;
    mat = validar("Matricula a buscar para mostrar: ", 300000, 399999);
    enc = busq_opt(index, i, mat, ord);
    if (enc != -1)
    {
        fa = fopen("datos.dat", "rb");
        fseek(fa, sizeof(Tdatos) * index[enc].indice, SEEK_SET);
        fread(&registros, sizeof(Tdatos), 1, fa);
        if (registros[enc].status == 1)
        {
            printf("Nombre: %s\n Apellidos: %s %s\nEdad: %d\nSexo: %s\nEstado: %s\nPuesto: %s\nMatricula: %d\nCelular: %d\n", registros[index[enc].indice].name,
        }
        fclose(fa);
    }
    else
    {
        printf("Empleado no encontrado\n");
    }
}
```

```
Matricula a buscar para mostrar: 300275
Nombre: LAURA
 Apellidos: RUIZ RUIZ
Edad: 32
Sexo: MUJER
Estado: AG
Puesto: ING. DE SISTEMAS
Matricula: 300275
Celular: 1009587
Presione una tecla para continuar . . .
```

4.- Ordenar :

El programa deberá ordenar el vector de índices por medio del método de ordenación más óptimo. Utilizar banderas para escoger el método más adecuado por el que se ordenará por el campo llave (**noempleado**) o no ordenarse si ya está ordenado.

```
int ordenar(Tindice index[], int i, int ord)
{

    if (ord == 1) // ya esta ordenado
    {
        ord= mergeSort(index, 0, i - 1);
    }
    else
    {
        ord = burbuja(index, i);
    }
    return ord;
}
```

2344	322715	GLORIA	SERRANO	ZAMORA	22	MUJER	AsServ	HG	1005633
2345	322721	CARMEN	ROJAS	SUAREZ	25	MUJER	DisUX	YN	1001015
2346	322727	LAURA	MENDEZ	MERCADO	46	MUJER	RepVInt	JC	1003955
2347	322734	ALBERTO	SUAREZ	ROSALES	30	HOMBRE	ChefEje	TS	1027274
2348	322738	JAVIER	ROLDAN	SOLANO	24	HOMBRE	Contado	DF	1008744
2349	322754	CLARA	VASQUEZ	FLORES	24	MUJER	EspERen	YN	1023949
2350	322772	AURORA	DIAZ	ROJAS	27	MUJER	GteProj	TL	1028926
2351	322777	GABRIELA	RIVERA	MEDINA	25	MUJER	IngCivil	CH	1005407
2352	322788	GLORIA	HIDALGO	ARIAS	50	MUJER	IngElec	SP	1004046
2353	322790	ELENA	VALENCIA	CRUZ	31	MUJER	EspSeg	GT	1002928
2354	322802	CLARA	DURAN	CASILLAS	22	MUJER	ChefEje	QT	1016929
2355	322805	MERCEDES	VARGAS	FLORES	21	MUJER	Enferm	CL	1027310
2356	322814	MERCEDES	ESPINOZA	TORRES	18	MUJER	EdCont	HG	1030967
2357	322822	CARMEN	MENDOZA	CRUZ	50	MUJER	Traduct	QT	1003715
2358	322840	CLARA	CASILLAS	SOLANO	43	MUJER	EspRRHH	DG	1027866
2359	322844	ALEJANDRA	SOTO	TORRES	50	MUJER	Enferm	GT	1025537
2360	322846	JOSE	CABRERA	OSORIO	32	HOMBRE	AnlVent	SL	1016785
2361	322854	MIGUEL	ESPINOSA	FERNANDEZ	28	HOMBRE	DisGraf	BC	1001279

5 Y 6.- Mostrar Todo:

El programa deberá mostrar todos los registros del Archivo Binario, preguntar: **ordenado o normal**. Usar el vector de índices para imprimirlo ordenado, y directamente desde el archivo si es normal.

```c
void imprimir(Tdatos registros[], Tindice index[], int i)
{
    int op;
    op = validar("1-Ordenado\n2-Normal\nSelecciona una opcion: ", 1, 2);
    system("CLS");
    if (op == 1)
    {
        print_ord(index, registros, i);
    }
    else
    {
        print_norm(registros, i);
    }
}
```

```c
void print_ord(Tindice index[], Tdatos reg[], int i)
{
    FILE *fa;
    printf("------------------------------------------------------------------------------------------------------------------------
    printf("  No.  | MATRICULA |     NOMBRE     |  APELLIDO P.  |     APELLIDO MAT.     | EDAD |  SEXO  |      PUESTO      |      ESTADO
    printf("------------------------------------------------------------------------------------------------------------------------

    fa = fopen("datos.dat", "r+b");
    if (fa)
    {
        for (int j = 0; j < i; j++)
        {
            if (reg[j].status == 1)
            {
                rewind(fa);
                fseek(fa, index[j].indice * sizeof(Tdatos), SEEK_SET);
                fread(&reg[index[j].indice], sizeof(Tdatos), 1, fa);
                printf("%-9d  %-11d  %-15s  %-20s  %-17s  %-7d  %-13s  %-18s  %-13s  %d\n", j + 1, reg[j].enrollment, reg[j].name, reg[j].LastName
            }
        }
        fclose(fa);
    }
    else
    {
        printf("Error al abrir archivo\n");
    }
}
```

7.- GENERAR ARCHIVO TEXTO:

El programa deberá generar un archivo de texto, el usuario debe proporcionar el nombre del archivo.

El programa deberá mostrar todos los registros del Archivo Binario, preguntar: **ordenado o normal**. Usar el vector de índices para imprimirlo ordenado, y directamente desde el archivo si es normal.

el programa podrá generar múltiples archivos para comprobar las salidas.

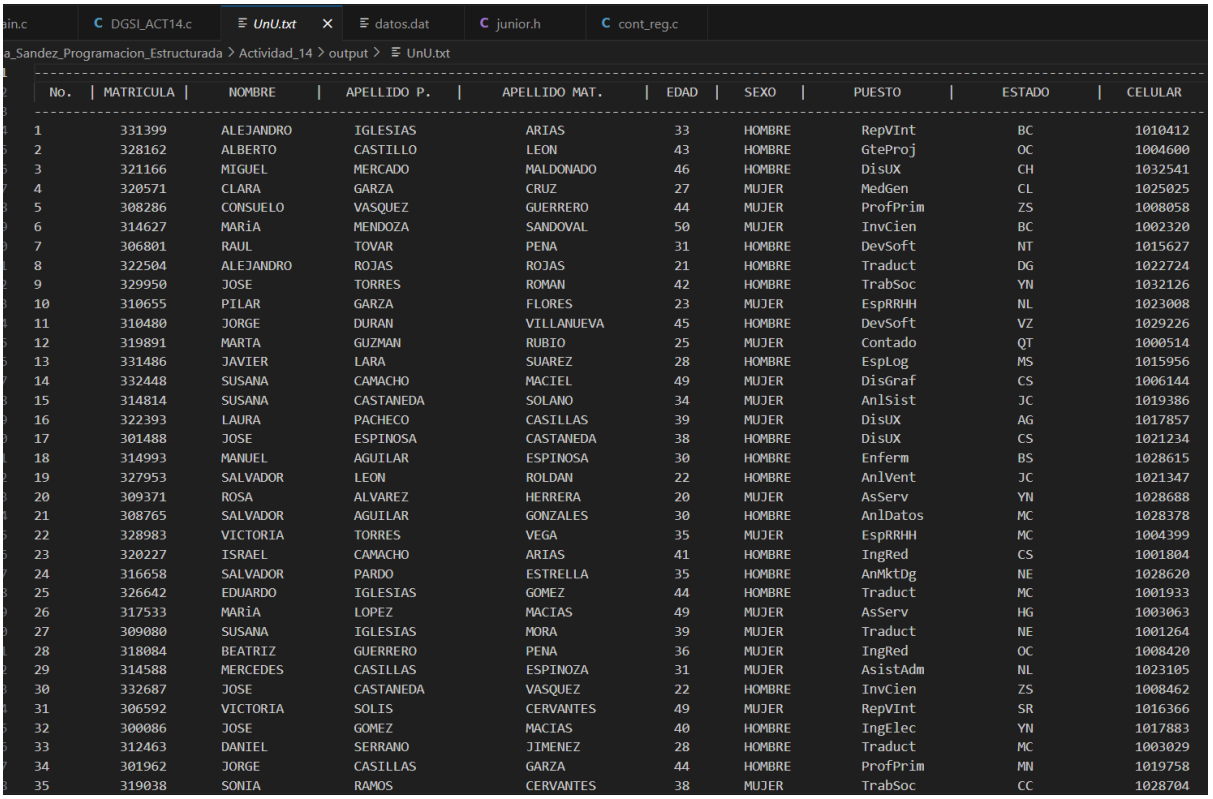
```c
void archivo_txt(Tindice index[], Tdatos reg[], int i)
{
    char nom[20];
    gen_nom_arch(nom);
    strcat(nom, ".txt");
    int op;
    op = validar("1-Ordenado\n2-Normal\nSelecciona una opcion: ", 1, 2);
    system("CLS");
    if (op == 1)
    {
        strcat(nom, "_ord");
        gen_arch_ord(nom, index, reg, i);
    }
    else
    {
        gen_arch_norm(nom, reg, i);
    }
}
```

```c
void gen_arch_ord(char nom[], Tindice index[], Tdatos reg[], int i)
{
    FILE *fa;
    fa = fopen(nom, "w");
    if (fa)
    {
        fprintf(fa, "--------------------------------------------------------------------------------------------------------
        fprintf(fa, "  No.  | MATRICULA |      NOMBRE      |  APELLIDO P.   |     APELLIDO MAT.     | EDAD |  SEXO  |      PUESTO      |      ESTADO
        fprintf(fa, "--------------------------------------------------------------------------------------------------------
        for (int j = 0; j < i; j++)
        {
            if (reg[j].status == 1)
            {
                fprintf(fa, "%-9d  %-11d  %-15s  %-20s  %-17s  %-7d  %-13s  %-18s  %-13s  %d\n", j + 1, reg[index[j].indice].enrollment, reg[index[j].indice
            }
        }
        printf("Archivo generado con exito\n");
        fclose(fa);
    }
    else
    {
        printf("Error al abrir archivo\n");
    }
}
```

```c
void gen_arch_norm(char nom[], Tdatos reg[], int i)
{
    FILE *fa;

    fa = fopen(nom, "w");

    if (fa)
    {
        fprintf(fa, "--------------------------------------------------------------------------------------------------------
        fprintf(fa, "  No.  | MATRICULA |      NOMBRE      |  APELLIDO P.   |     APELLIDO MAT.     | EDAD |  SEXO  |      PUESTO      |      ESTADO      | CELULA
        fprintf(fa, "--------------------------------------------------------------------------------------------------------
        for (int j = 0; j < i; j++)
        {
            if (reg[j].status == 1)
            {
                fprintf(fa, "%-9d  %-11d  %-15s  %-20s  %-17s  %-7d  %-13s  %-18s  %-13s  %d\n", j + 1, reg[j].enrollment, reg[j].name, reg[j].LastName1, reg[j].LastNa
            }
        }
        printf("Archivo generado con exito\n");
        fclose(fa);
    }
    else
    {
        printf("Error al abrir archivo\n");
    }
}
```



ain.c | DGSI_ACT14.c | UnU.txt | datos.dat | junior.h | cont_reg.c

a_Sandez_Programacion_Estructurada > Actividad_14 > output > UnU.txt

No.	MATRICULA	NOMBRE	APELLIDO P.	APELLIDO MAT.	EDAD	SEXO	PUESTO	ESTADO	CELULAR
1	331399	ALEJANDRO	IGLESIAS	ARIAS	33	HOMBRE	RepVInt	BC	1010412
2	328162	ALBERTO	CASTILLO	LEON	43	HOMBRE	GteProj	OC	1004600
3	321166	MIGUEL	MERCADO	MALDONADO	46	HOMBRE	DisUX	CH	1032541
4	320571	CLARA	GARZA	CRUZ	27	MUJER	MedGen	CL	1025025
5	308286	CONSUELO	VASQUEZ	GUERRERO	44	MUJER	ProfPrim	ZS	1008058
6	314627	MARiA	MENDOZA	SANDOVAL	50	MUJER	InvCien	BC	1002320
7	306801	RAUL	TOVAR	PENA	31	HOMBRE	DevSoft	NT	1015627
8	322504	ALEJANDRO	ROJAS	ROJAS	21	HOMBRE	Traduct	DG	1022724
9	329950	JOSE	TORRES	ROMAN	42	HOMBRE	TrabSoc	YN	1032126
10	310655	PILAR	GARZA	FLORES	23	MUJER	EspRRHH	NL	1023008
11	310480	JORGE	DURAN	VILLANUEVA	45	HOMBRE	DevSoft	VZ	1029226
12	319891	MARTA	GUZMAN	RUBIO	25	MUJER	Contado	QT	1000514
13	331486	JAVIER	LARA	SUAREZ	28	HOMBRE	EspLog	MS	1015956
14	332448	SUSANA	CAMACHO	MACIEL	49	MUJER	DisGraf	CS	1006144
15	314814	SUSANA	CASTANEDA	SOLANO	34	MUJER	AnlSist	JC	1019386
16	322393	LAURA	PACHECO	CASILLAS	39	MUJER	DisUX	AG	1017857
17	301488	JOSE	ESPINOSA	CASTANEDA	38	HOMBRE	DisUX	CS	1021234
18	314993	MANUEL	AGUILAR	ESPINOSA	30	HOMBRE	Enferm	BS	1028615
19	327953	SALVADOR	LEON	ROLDAN	22	HOMBRE	AnlVent	JC	1021347
20	309371	ROSA	ALVAREZ	HERRERA	20	MUJER	AsServ	YN	1028688
21	308765	SALVADOR	AGUILAR	GONZALES	30	HOMBRE	AnlDatos	MC	1028378
22	328983	VICTORIA	TORRES	VEGA	35	MUJER	EspRRHH	MC	1004399
23	320227	ISRAEL	CAMACHO	ARIAS	41	HOMBRE	IngRed	CS	1001804
24	316658	SALVADOR	PARDO	ESTRELLA	35	HOMBRE	AnMktDg	NE	1028620
25	326642	EDUARDO	IGLESIAS	GOMEZ	44	HOMBRE	Traduct	MC	1001933
26	317533	MARiA	LOPEZ	MACIAS	49	MUJER	AsServ	HG	1003063
27	309080	SUSANA	IGLESIAS	MORA	39	MUJER	Traduct	NE	1001264
28	318084	BEATRIZ	GUERRERO	PENA	36	MUJER	IngRed	OC	1008420
29	314588	MERCEDES	CASILLAS	ESPINOZA	31	MUJER	AsistAdm	NL	1023105
30	332687	JOSE	CASTANEDA	VASQUEZ	22	HOMBRE	InvCien	ZS	1008462
31	306592	VICTORIA	SOLIS	CERVANTES	49	MUJER	RepVInt	SR	1016366
32	300086	JOSE	GOMEZ	MACIAS	40	HOMBRE	IngElec	YN	1017883
33	312463	DANIEL	SERRANO	JIMENEZ	28	HOMBRE	Traduct	MC	1003029
34	301962	JORGE	CASILLAS	GARZA	44	HOMBRE	ProfPrim	MN	1019758
35	319038	SONIA	RAMOS	CERVANTES	38	MUJER	TrabSoc	CC	1028704