

**Ingeniero en computación**

**Ingeniero en Software y tecnologías emergentes**

**Materia:** Programación Estructurada / Clave 36276

**Alumno:** Eliel Alfonso Ontiveros Ojeda

**Matrícula:** 368746

**Maestro:** Pedro Núñez Yépiz

**Actividad No. :** 14 (ANEXO)

**Tema - Unidad :** ARCHIVOS DE TEXTO Y BINARIOS

**Ensenada Baja California a 28 de noviembre del 2023**

```

void menu(void)
{
    Tkey *indiceVecto = NULL;
    Twkr *registros = NULL;
    int numRegistro = 0;
    FILE *archivoBinari;

    cargarArchiv (&indiceVecto , &registros, &numRegistro );
    char currentPath[FILENAME_MAX];
    if (!getcwd(currentPath, sizeof(currentPath)))
    {
        perror("Error getting current director ");
    }
    else
    {
        printf("Current working directory: %s\n", currentPath);
    }

    int opcion;
    archivoBinari = fopen("datos.dat", "rb+");
    if (!archivoBinari)
    {
        perror("Error al abrir el archivo binari ");
        exit(EXIT_FAILURE);
    }

    do
    {
        system("CLS");
        ImprimirMen ();
        printf("\nSeleccione una opcion: ");
        opcion = val_num(0, 8);

        switch (opcion)
        {
            case 1:
                agregarRegistr (archivoBinari , &indiceVecto , &numRegistro );
                break;

            case 2:
                eliminarRegistr (archivoBinari , indiceVecto , numRegistro );
                break;

            case 3:
            {
                Tkey matriculaBusca;
                printf("Ingrese el numero de empleado a buscar: ");
                matriculaBusca = val_num(300000, 399999);

                int indiceEncontrad = buscarMatricul (indiceVecto , numRegistro , matriculaBusca );

                if (indiceEncontrad != -1)
                {
                    fseek(archivoBinari , indiceEncontrad * sizeof(Twkr), SEEK_SET);
                    Twkr registroEncontrad;
                    fread(&registroEncontrad , sizeof(Twkr), 1, archivoBinari );
                    printf("Registro encontrado \n");
                    imprimirRegistr (&registroEncontrad );
                }
                else
                {
                    printf("Numero de empleado no encontrado \n");
                }
                system("PAUSE");
                break;
            }

            case 4:
                ordenarIndice (indiceVecto , numRegistro );
                break;

            case 5:
                mostrarTodo(archivoBinari , indiceVecto , numRegistro );
                break;

            case 6:
                generarArchivoText (archivoBinari , indiceVecto , numRegistro );
                break;

            case 7:
                empaquetarArchiv (archivoBinari );
                break;
        }

    } while (opcion != 0);

    fclose(archivoBinari );
    printf("Saliendo del Program \n");
    free(indiceVecto );
    free(registros);
}

```

```

void ImprimirMen (void)
{
    system("cls");
    printf("----- \n");
    printf("-          Men \n");
    printf("u----- \n");
    printf("|| 1 - Agregar          | \n");
    printf("|| 2 - Eliminar         | \n");
    printf("|| 3 - Buscar           | \n");
    printf("|| 4 - Ordenar          | \n");
    printf("|| 5 - Imprimir Archivo Original | \n");
    printf("|| 6 - Generar Archivo Texto | \n");
    printf("|| 7 - Empaquetar       | \n");
    printf("|| 0 - Salir            | \n");
    printf("t----- \n");
    printf("----- ");
}

```

```

Tkey generarMatriculaUnic (const Tkey *indiceVecto , int numRegistro )
{
    Tkey nuevaMatricul;
    bool matriculaExistent;

    do
    {
        nuevaMatricul = numero_aleatori (300000, 399999);
        matriculaExistent = false;
        for (int i = 0; i < numRegistro ; i++)
        {
            if (indiceVecto [i] == nuevaMatricul )
            {
                matriculaExistent = true;
                break;
            }
        }

    } while (matriculaExistent );

    return nuevaMatricul;
}

```

```

void agregarRegistr (FILE *archivoBinari , Tkey **indiceVecto , int *numRegistro )
{
    Twrkr nuevoUsuari ;
    indi indice[MAX_REGISTRO ];
    char temp[30];
    int c, i;

    c = rand() % (2 - 1 + 1) + 1;

    nuevoUsuari .status = ;
    o 1
    genNombre(temp, c);
    strcpy(nuevoUsuari .name, temp);
    o
    genApellido(temp);
    strcpy(nuevoUsuari .LastName1, temp);
    o
    genApellido(temp);
    strcpy(nuevoUsuari .LastName2, temp);
    o
    genSexo(temp);
    strcpy(nuevoUsuari .sex, temp);
    o
    genPuesto(temp);
    strcpy(nuevoUsuari .JobPstion, temp);
    o
    genEstado(temp);
    strcpy(nuevoUsuari .state, temp);
    o
    nuevoUsuari .age = numero_aleatori (18, 60);
    nuevoUsuari .cellPhone = numero_aleatori (1000000, 1999999);
    o
    o
    nuevoUsuari .enrollment = generarMatriculaUnic (*indiceVecto , *numRegistro );
    o
    a
    r
    s
    fseek(archivoBinari , 0, SEEK_END);
    o
    fwrite(&nuevoUsuari , sizeof(Twrkr), 1, archivoBinari );
    (*numRegistro )++;
    o
    s
    *indiceVecto = (Tkey *)realloc(*indiceVecto , (*numRegistro + *numRegistro / 4)
    * sizeof(Tkey));
    r
    s
    s

    if (*indiceVecto == NULL)
    {
        r
        perror("Error reallocating memory for indiceVecto ");
        exit(EXIT_FAILURE );
    }
    E

    (*indiceVecto )[*numRegistro - 1] = nuevoUsuari .enrollmen ;
    r
    s
    o
    t
    indi *indiceArray = (indi *)malloc(*numRegistro * sizeof(indi));
    s

    if (indiceArray == NULL)
    {
        perror("Error allocating memory for indiceArra ");
        exit(EXIT_FAILURE );
    }
    E

    for (int i = 0; i < *numRegistro ; i++)
    {
        s
        indiceArray[i].indice = i;
        indiceArray[i].llave = (*indiceVecto )[i];
    }
    r

    crear_index(*numRegistro , indiceArray);
    s
    free(indiceArray);

    printf("Registro Creado \n");
    system("BAUSE");
}

```

## Menu

- |   |   |                           |
|---|---|---------------------------|
| 1 | - | Agregar                   |
| 2 | - | Eliminar                  |
| 3 | - | Buscar                    |
| 4 | - | Ordenar                   |
| 5 | - | Imprimir Archivo Original |
| 6 | - | Generar Archivo Texto     |
| 7 | - | Empaquetar                |
| 0 | - | Salir                     |

Seleccione una opcion: 1

Registro Creado

Presione una tecla para continuar . . .

```

void eliminarRegistr (FILE *archivoBinari , Tkey *indiceVecto , int numRegistro )
{
    o
    Tkey matricula;
    int indiceEncontrad ;
    o
    printf("Ingrese el numero de empleado a eliminar: ");
    matricula = val_num(300000, 399999);

    indiceEncontrad = buscarMatricul (indiceVecto , numRegistro , matricula);
    o
    a
    r
    s
    if (indiceEncontrad != -1)
    {
        o
        fseek(archivoBinari , indiceEncontrad * sizeof(TWrkr), SEEK_SET);
        o
        o
        TWrkr registroEncontrad ;
        fread(&registroEncontrad , sizeof(TWrkr), 1, archivoBinari );
        o
        o
        printf("Registro encontrad \n");
        imprimirRegistr (&registroEncontrad );
        o
        o
        char respuesta;
        printf("Desea eliminar el registro? (1 = Si, 2 = No): ");
        respuesta = val_num(0, 1);

        if (respuesta == 1)
        {
            registroEncontrad .status = 0;
            o
            fseek(archivoBinari , -sizeof(TWrkr), SEEK_CUR);
            fwrite(&registroEncontrad , sizeof(TWrkr), 1, archivoBinari );
            o
            o
            printf("Registro eliminado exitosament \n");
        }
        e.
        else
        {
            printf("Eliminacion cancelad \n");
        }
        a.
    }
    o.
    else
    {
        printf("Numero de empleado no encontrad \n");
    }
    o.

    system("PAUSE");
}

```

```

Seleccione una opcion: 2
Ingrese el numero de empleado a eliminar: 300019
Registro encontrado:
Status: 1
Noempleado: 300019
Nombre: MIGUEL
Apellido Paterno: HIDALGO
Apellido Materno: FLORES
Sexo: HOMBRE
Posicion: InvCien
Estado: DF
Edad: 21
Num. Telefono: 1016520
-----
Desea eliminar el registro? (1 = Si, 2 = No):

```

```

int buscarMatricul (const Tkey *indiceVecto , int numRegistro , Tkey matricula)
{
    for (int i = 0; i < numRegistro ; ++i)
    {
        if (indiceVecto [i] == matricula)
        {
            return i;
        }
    }
    return -1;
}

```

```

void buscarRegistr (FILE *archivoBinari , const Tkey *indiceVecto , int numRegistro ,
Tkey matricula)
{
    int indiceEncontrad = buscarMatricul (indiceVecto , numRegistro , matricula);
    if (indiceEncontrad != -1)
    {
        fseek(archivoBinari , indiceEncontrad * sizeof(TWrkr), SEEK_SET);
        TWrkr registroEncontrad ;
        fread(&registroEncontrad , sizeof(TWrkr), 1, archivoBinari );
        printf("Registro encontrad \n");
        printf("Status: % \n", registroEncontrad .status);
        printf("Noempleado: % \n", registroEncontrad .enrollmen );
        printf("Nombre: % \n", registroEncontrad .name);
        printf("Apellido Paterno: % \n", registroEncontrad .LastName1);
        printf("Apellido Materno: % \n", registroEncontrad .LastName2);
        printf("Sexo: %s\n", registroEncontrad .sex);
        printf("Posicion: % \n", registroEncontrad .JobPstion);
        printf("Estado: % \n", registroEncontrad .state);
        printf("Edad: %d\n", registroEncontrad .age);
        printf("Num. Telefono % \n", registroEncontrad .cellPhone);
        printf("-----\n");
    }
    else
    {
        printf("Matrícula no encontrad \n");
    }

    system("PAUSE");
}

```

```

Seleccione una opcion: 3
Ingrese el numero de empleado a buscar: 300019
Registro encontrado:
Status: 0
Noempleado: 300019
Nombre: MIGUEL
Apellido Paterno: HIDALGO
Apellido Materno: FLORES
Sexo: HOMBRE
Posicion: InvCien
Estado: DF
Edad: 21
Num. Telefono: 1016520
-----
Presione una tecla para continuar . . .

```

```

void imprimirRegistr (const TWkr *registro)
{
    o
    printf("Status: % \n", registro->status);
    printf("Noempleado: % \n", registro->enrollmen );
    printf("Nombre: % \n", registro->name);
    printf("Apellido Paterno: % \n", registro->LastName1);
    printf("Apellido Materno: % \n", registro->LastName2);
    printf("Sexo: %s\n", registro->sex);
    printf("Posicion: % \n", registro->JobPstion);
    printf("Estado: % \n", registro->state);
    printf("Edad: %d\n", registro->age);
    printf("Num. Telefono: % \n", registro->cellPhone);
    printf("d----- \n");
}

-

void mostrarTodo(FILE *archivoBinari , const Tkey *indiceVecto , int numRegistro )
{
    o
    r
    s

    int opcionOrden;

    printf("Seleccione la opción \n");
    printf("1: Imprimir ordenado por matrícula \n");
    printf("2. Imprimir norma \n");
    opcionOrden = val_num(1, 2);

    if (opcionOrden == 1)
    {
        TWkr *registros = (TWkr *)malloc(numRegistro * sizeof(TWkr));
        fseek(archivoBinari , 0, SEEK_SET);s
        fread(&registros, sizeof(TWkr), numRegistro , archivoBinari );
        s
        o
        qsort(registros, numRegistro , sizeof(TWkr), compararRegistrosPorMatricul );
        s
        a
        for (int i = 0; i < numRegistro ; ++i)
        {
            s
            imprimirRegistr (&registros[i]);
        }
        o

        free(registros);
    }
    else if (opcionOrden == 2)
    {
        fseek(archivoBinari , 0, SEEK_SET);
        o
        TWkr registro;
        while (fread(&registro, sizeof(TWkr), 1, archivoBinari ) == 1)
        {
            o
            imprimirRegistr (&registro);
        }
        o
    }
    else
    {
        printf("Opción no válida \n");
        a.
    }

    system("PAUSE");
}

```

-----  
Status: 1  
Noempleado: 332757  
Nombre: PEDRO  
Apellido Paterno: QUINTERO  
Apellido Materno: VALDES  
Sexo: HOMBRE  
Posicion: EspLog  
Estado: JC  
Edad: 44  
Num. Telefono: 1023786  
-----

Status: 1  
Noempleado: 332761  
Nombre: MERCEDES  
Apellido Paterno: TORRES  
Apellido Materno: FLORES  
Sexo: MUJER  
Posicion: EspSeg  
Estado: SP  
Edad: 49  
Num. Telefono: 1004974  
-----

Presione una tecla para continuar . . . █

```

void generarArchivoText (FILE *archivoBinari , const Tkey *indiceVecto , int
numRegistro )
{
    char nombreArchivoText [100];
    printf("Ingrese el nombre del archivo de texto: ");
    validar_strin (nombreArchivoText , 40);
    FILE *archivoText = fopen(nombreArchivoText , "w");
    if (!archivoText )
    {
        perror("Error al abrir el archivo de text ");
        exit(EXIT_FAILURE );
    }

    int opcionOrden;

    printf("Seleccione la opción \n");
    printf("1: Imprimir ordenad \n");
    printf("2. Imprimir norma \n");
    opcionOrden = val_num(1, 2);

    if (opcionOrden == 1)
    {
        Tkey *copiaIndiceVecto = (Tkey *)malloc(numRegistro * sizeof(Tkey));
        if (copiaIndiceVecto == NULL)
        {
            perror("Error allocating memory for copiaIndiceVecto ");
            exit(EXIT_FAILURE );
        }

        memcpy(copiaIndiceVecto , indiceVecto , numRegistro * sizeof(Tkey));
        ordenarIndice (copiaIndiceVecto , numRegistro );
        for (int i = 0; i < numRegistro ; ++i)
        {
            int matriculaOrdenad = copiaIndiceVecto [i];
            int indiceOrdenad = buscarMatricul (indiceVecto , numRegistro ,
matriculaOrdenado);
            fseek(archivoBinari , indiceOrdenad * sizeof(TWrkr), SEEK_SET);
            TWrkr registro;
            fread(&registro, sizeof(TWrkr), 1, archivoBinari );
            imprimirRegistroEnText (archivoText , &registro);
        }

        free(copiaIndiceVecto );
    }

    else if (opcionOrden == 2)
    {
        fseek(archivoBinari , 0, SEEK_SET);
        TWrkr registro;
        while (fread(&registro, sizeof(TWrkr), 1, archivoBinari ) == 1)
        {
            imprimirRegistroEnText (archivoText , &registro);
        }
    }
    else
    {
        printf("Opción no válid \n");
    }

    fclose(archivoText );
    printf("Archivo de texto generado exitosamente: % \n", nombreArchivoText );
}

```



```

void imprimirRegistroEnText (FILE *archivoText , const TWkr *registro)
{
    o
    fprintf(archivoText , "Status: % \n", registro->status);
    fprintf(archivoText , "Noempleado: % \n", registro->enrollmen );
    fprintf(archivoText , "Nombre: % \n", registro->name);
    fprintf(archivoText , "Apellido Paterno: % \n", registro->LastName1);
    fprintf(archivoText , "Apellido Materno: % \n", registro->LastName2);
    fprintf(archivoText , "Sexo: %s\n", registro->sex);
    fprintf(archivoText , "Posicion: % \n", registro->JobPstion);
    fprintf(archivoText , "Estado: % \n", registro->state);
    fprintf(archivoText , "Edad: %d\n", registro->age);
    fprintf(archivoText , "Num. Telefono: % \n", registro->cellPhone);
    fprintf(archivoText , "d----- \n");
}
    o
    -

```

```

void empaquetarArchiv (FILE *archivoBinari )
{
    o
    FILE *doc = fopen("data.dat", "rb");
    ;
    FILE *respaldo = fopen("data.bak", "wb");

    TWkr registros;

    if (doc)
    {
        while (fread(&registros, sizeof(TWkr), 1, doc))
        {
            if (registros.status == 1)
            {
                fwrite(&registros, sizeof(TWkr), 1, respaldo);
            }
        }
    }

    fclose(doc);
    fclose(respaldo);

    printf("Archivo Empaquetad \n");
    system("BAUSE");
}

```

ABV






LOL

datos

Archivo    Editar    Ver

Status: 0  
Noempleado: 331399  
Nombre: ALEJANDRO  
Apellido Paterno: IGLESIAS  
Apellido Materno: ARIAS  
Sexo: HOMBRE  
Posicion: RepVInt  
Estado: BC  
Edad: 33  
Num. Telefono: 1010412  
-----  
Status: 1  
Noempleado: 328162  
Nombre: ALBERTO  
Apellido Paterno: CASTILLO  
Apellido Materno: LEON  
Sexo: HOMBRE  
Posicion: GteProj  
Estado: OC  
Edad: 43  
Num. Telefono: 1004600  
-----  
Status: 1  
Noempleado: 321166  
Nombre: MIGUEL  
Apellido Paterno: MERCADO  
Apellido Materno: MALDONADO  
Sexo: HOMBRE  
Posicion: DisUX  
Estado: CH  
Edad: 46  
Num. Telefono: 1032541  
-----  
Status: 1  
Noempleado: 320571  
Nombre: CLARA  
Apellido Paterno: GARZA  
Apellido Materno: CRUZ  
Sexo: MUJER  
Posicion: MedGen

Ln 10, Col 23

 datos	28/11/2023 09:41 a. m.	Archivo BAK	736 KB
 datos	28/11/2023 09:37 a. m.	Archivo DAT	737 KB
 EAOO_PE_ACT14	28/11/2023 09:39 a. m.	Aplicación	495 KB
 HOLABUENASTARDES	28/11/2023 09:41 a. m.	Archivo	672 KB
 index	28/11/2023 09:35 a. m.	Archivo DAT	198 KB