

PROYECTO SOFTWARE

PRESENTA:

- OSCAR AYALA ELIZALDE
- JUAN CARLOS FLORES MORA
- MIGUEL ALEJANDRO FLORES SOTELO
- SERGIO DE JESÚS CASTILLO MOLANO

ÍNDICE

```
4 # Prevent database truncation if the transaction fails
5 abort("The Rails environment is running in production mode!
6 require 'spec_helper'
7 require 'rspec/rails'
8
9 require 'capybara/rspec'
10 require 'capybara/rails'
11
12 Capybara.javascript_driver = :webkit
13 Category.delete_all; Category.create!(name: "Electronics")
14 Shoulda::Matchers.configure do |config|
15   config.integrate do |with|
16     with.test_framework :rspec
17     with.library :rails
18   end
19 end
20
21 # Add additional requires below this line to access routes in subresources:
22 # For example, given the routes
23 #   resources :articles
24 #   resources :comments, :comment do
25 #     get 'subComment/:id', on: :comment
26 #   end
27 #
28 # In your feature spec, you can now do
29 #   visit article_path(@article)
30 #   click_on 'Comments'
31 #   click_on 'SubComment', id: @comment.id
32
33 # Requires supporting ruby files with custom matchers and helpers
34 # in spec/support/ and its subdirectories. This directory also
35 # contains supporting files for this generator.
36 # run as spec files by default. You can change this by
37 # specifying the --type option for generator or by overwriting
38 # spec_type in spec/spec.rb. For example:
39 #   # in _spec.rb
40 #   spec_type :unit
41
42 # in _spec.rb will both be required by default. If you need to
43 # load more specific files from a folder, you can add more paths to
44 # the require_paths array:
45 #   # in _spec.rb
46 #   require_paths = [
47 #     "spec/support/rails"
48 #   ]
49
50 # run twice. It is recommended that you do not name
51 # your test file as _spec.rb. You can configure this
52 # behavior with the --name option for generator or by
53 # overwriting the spec_file_name variable in spec/spec.rb
54 #   # in _spec.rb
55 #   spec_file_name = "spec"
56
57 # Note that there is a missing trailing slash at the end of
58 # the mongoid path in the original template. This is fixed
59 # here to prevent errors during the generation of the
60 # Mongoid configuration file.
```

| | |
|--------------------------------|----|
| 1. PLANTEAMIENTO DEL PROBLEMA | 4 |
| 2. DIAGRAMA DE FLUJO | 5 |
| 3. CÓDIGO FUENTE | 6 |
| 4. EJEMPLOS DEL FUNCIONAMIENTO | 27 |
| 5. CONCLUSIÓN | 30 |

INTRODUCCIÓN

En la actualidad optimizar las tareas cotidianas, nos ayuda a ahorrar tiempo ya que es uno de los recursos más importantes en el campo empresarial.

Es por ello que softwork se creo con el propósito de facilitar la gestión de empleados para un manejo óptimo y eficiente, permitiendo acciones como:

- Carga de empleados.
- Alta de nuevos empleados.
- Baja de empleados existentes.
- Modificar características del empleado.
- Mostrar la lista de empleados.
- Consultar los datos de un empleado en específico.

Así como un apartado únicamente para empleados donde pueden consultar sus datos por medio de su ID.





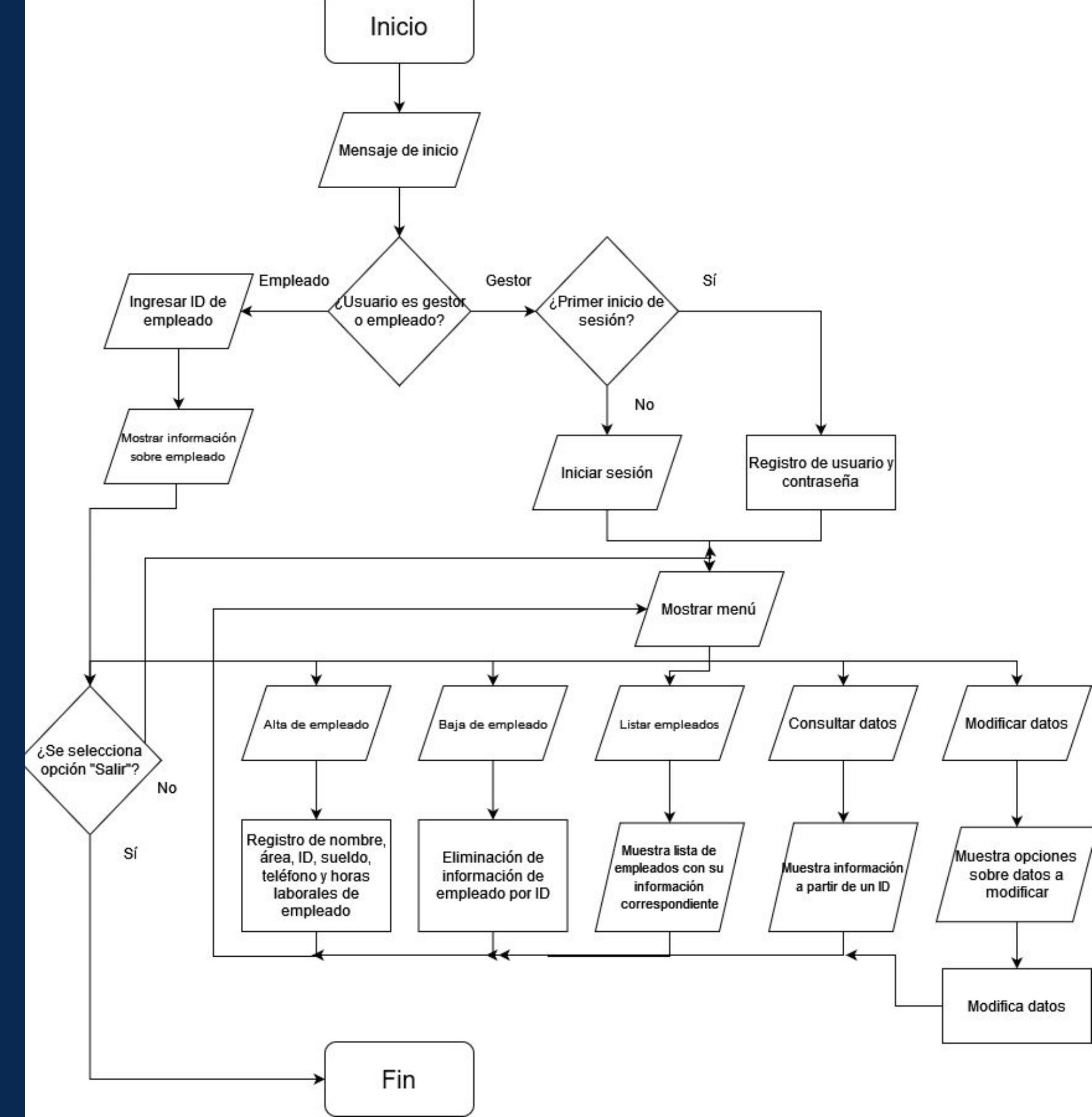
PLANTEAMIENTO DEL PROBLEMA

El problema se aborda en el momento que se utilizan conjuntos de datos que contienen la información de los empleados, como su nombre, el área en la que se desempeñan, su identificación única dentro de la empresa, su sueldo, el número de teléfono y las horas que laboran en la institución.

El manejo adecuado de dicha información resulta crucial para optimizar los procesos internos y garantizar llevar un registro actualizado de los empleados, al igual que asignar las correctas tareas dentro de la empresa, entre otros procesos.

DIAGRAMA DE FLUJO

El diagrama representa el proceso de cómo se ejecuta la secuencia de actividades del programa



CÓDIGO FUENTE

BIBLIOTECAS

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <locale.h>
4 #include <string.h>
```



CÓDIGO FUENTE

FUNCIONES DE CABECERA

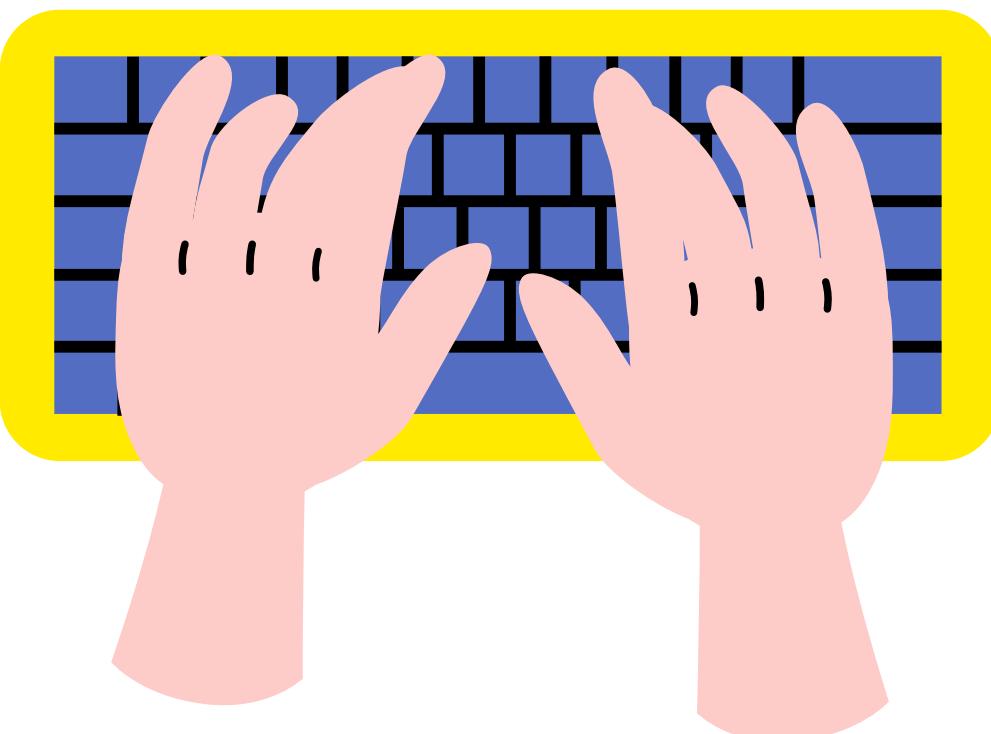


```
1 //PROTOTIPOS DE FUNCIONES
2 struct empleado{
3     char nombre[200];
4     char area[200];
5     int id;
6     float sueldo;
7     char num_cel[200];***_
8     int hrs_sem_trbj;
9 };
10 void mostrarmenu();
11 int cargarempleados(struct empleado empleados[100],int producc,int finan,int rechum,int ingen,int geren,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte); //FUNCION QUE DEVUELVE LOS DATOS DE LOS EMPLEADOS
12 void guardar_empleados(struct empleado empleados[100],int num_empleados);
13 void listar_empleados(struct empleado empleados [100],int num_empleados); //FUNCION QUE ENLISTA A LOS EMPLEADOS EXISTENTES
14 int alta_de_empleado(struct empleado empleados[100],int num_empleados,int producc,int finan,int rechum,int ingen,int geren,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte);
15 int baja_de_empleado(struct empleado empleados[100],int num_empleados,int producc,int finan,int rechum,int ingen,int geren,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte);
16 void consultar_empleado_gestor(struct empleado empleados[100],int num_empleados); //FUNCION QUE SIRVE PARA CONSULTAR LOS DATOS DE UN EMPLEADO EN ESPECIFICO PARA EL GESTOR
17 void consultar_empleado_empleado(struct empleado empleados[100],int num_empleados); //FUNCION QUE SIRVE PARA CONSULTAR LOS DATOS DEL EMPLEADO
18 void modificar_area(struct empleado empleados[100],int num_empleados,int producc,int finan,int rechum,int ingen,int geren,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte);
19 void modificar_sueldo(struct empleado empleados [100],int num_empleados); //FUNCION QUE MODIFICA EL SUELDO DE LOS EMPLEADOS
20 void modificar_num_cel(struct empleado empleados[100],int num_empleados); //FUNCION QUE MODIFICA EL NUMERO DE CELULAR DE LOS EMPLEADOS
21 void modificar_hrs_sem_trbj(struct empleado empleados [100],int num_empleados); //FUNCION QUE MODIFICA LAS HORAS DE SEMANA DE TRABAJO
22 int comprobacion_existencia_empleado(struct empleado empleados[100], int num_empleados, int id); //FUNCION QUE NOS AYUDARA PARA SABER SI EXISTE UN EMPLEADO
23 int lecturaCredenciales(char *inicusuAdm, char *iniclavAdm); // Funcion para leer las credenciales guardadas en archivo txt
24 void inicioSesion(char *inicusuAdm, char *iniclavAdm); // Funcion para inicio de sesion
25 void graficadora(int grafica[50][49],int produccion,int finanzas,int rechumanos,int ingenieria,int gerencia,int limpieza,int transporte);
26 int claveUsuario(void); // Funcion para registro de credenciales de gestor
```

FUNCIONES

FUNCIÓN PARA MOSTRAR EL MENÚ DEL GESTOR

```
● ● ●  
1 void mostrarmenu(){  
2     printf("\n      -----");  
3     printf("\n      | Menu del administrador.\t\t\t\t|\n");  
4     printf("      |\t[1]-Alta empleado\t\t\t\t|\n      |\t[2]-Baja empleado\t\t\t\t|\n      |\t[3]-Listar empleados\t\t\t\t|\n      |\t[4]-Consultar datos de un empleado\t\t\t|\n      |\t[5]-Modificar datos del empleado\t\t\t|\n      |\t[6]-Salir\t\t\t\t\t\t\t\t|\n");  
5     printf("      -----");  
6     printf("\tIntroduce la opcion que deseas realizar: ");//FUNCION PARA IMPRIMIR EL MENU  
7 }
```



FUNCIÓN PARA CARGAR LOS EMPLEADOS QUE SE ENCUENTRAN EN EL ARCHIVO

```
1 int cargarempleados(struct empleado empleados[100],int producc,int finan,int rechum,int ingen,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte){//FUNCION QUE DEVUELVE EL NUMERO DE EMPLEADOS QUE EXISTIRAN EN EL FICHERO
2     int num_empleados=0;//NUMERO DE EMPLEADOS
3     FILE *f;//VARIABLE DE TIPO FILE PARA LEER EL FICHERO
4     f=fopen("empleados.txt","r");//ABRE EL FICHERO EN MODO "READING" QUE SIRVE PARA PASAR AL FICHERO LOS DATOS, EN ESTE CASO LOS EMPLEADOS
5     char cadena[250];//CADENA QUE ALMACENA CADA LINEA LEIDA DEL FICHERO
6     char delimitador[]=";"//VARIABLE QUE DIVIDE EN TROZOS LAS LINEAS DEL FICHERO POR ; EN ESTE CASO
7
8     while(feof(f)==0){//LA FUNCION FEOF MANDA DOS RESULTANTES, 1 o 0: 1 SIGNIFICA QUE YA TERMINO DE LEER EL FICHERO Y 0 QUE AUN NO HA TERMINADO DE LEER EL FICHERO
9     //POSTERIORMENTE SE PROCESA CADA LINEA, CADA LINEA REPRESENTA UN EMPLEADO, SEGUIDO DE SUS DATOS
10    struct empleado e;//AQUI UTILIZAS LA ESTRUCTURA EMPLEADO PARA CARGAR LOS DATOS DEL EMPLEADO DE LA LINEA QUE SE ESTE LETYENDO
11    fgets(cadena,250,f);// CON FGETS OBTENES LA LINEA QUE SE GUARDA EN LA VARIABLE CADENA DE HASTA 200 CARACTERES, Y SI INGRESA EL FICHERO QUE SE LEE EN ESTE CASO ES f
12    char *token= strtok(cadena,delimitador);//LA FUNCION STRTOK DIVIDE LA CADENA EN PARTES POR EL DELIMITADOR QUE ES ; Y DEVUELVE LOS DIFERENTES CAMPOS QUE ESTAN SEPARADOS POR EL DELIMITADOR POR LO TANTO SE ALMACENA EN UNA VARIABLE DE TIPO CADENA EN ESTE CASO LLAMADA TOKEN, SE VA LLAMANDO LA FUNCION STRTOK HASTA QUE LLEGUE AL FINAL DE LA CADENA Y TOKEN OBTENGA EL VALOR NULL
13
14    if(token!=NULL){//POR LO TANTO SI TOKEN ES DISTINTO DE NULL, ESTA DENTRO DE LOS CAMPOS
15        int campo=1;//VARIABLE QUE SIRVE PARA VER EN QUE CAMPO TE ENCUENTRAS (SI ES EL NOMBRE, EL AREA, EL ID, ETC)
16        while (token!=NULL){//MIENTRAS TOKEN SEA DIFERENTE DE NULL, OSEA QUE NO HAYALLEGADO HASTA EL FINAL
17            if (campo==1){
18                strcpy(e.nombre,token);//SI VA EN EL CAMPO 1 SIGNIFICA QUE ES EL NOMBRE Y LO COPIA EN E.NOMBRE DE LA ESTRUCTURA
19            }
20            else if(campo==2){//SI VA EN EL CAMPO 2 SIGNIFICA QUE ES EL AREA Y LO COPIA EN E.AREA DE LA ESTRUCTURA
21                strcpy(e.area,token);
22                if(strcmp(e.area,"Produccion")==0){
23                    producc++;
24                }
25                if(strcmp(e.area,"Finanzas")==0){
26                    finan++;
27                }
28                if(strcmp(e.area,"Ingenieria")==0){
29                    ingen++;
30                }
31                if(strcmp(e.area,"R. Humanos")==0){
32                    rechum++;
33                }
34                if(strcmp(e.area,"Limpieza")==0){
35                    limp++;
36                }
37                if(strcmp(e.area,"Gerencia")==0){
38                    geren++;
39                }
40                if(strcmp(e.area,"Transporte")==0){
41                    transp++;
42                }
43            }
44            else if(campo==3){//SI VA EN EL CAMPO 3 SIGNIFICA QUE ES EL ID Y LO COPIA EN E.ID DE LA ESTRUCTURA, DEBIDO A QUE SON CARACTERES LOS QUE SE LEEN Y ESTAS EN EL ID SE CONVIERTEN A ENTEROS CON LA FUNCION ATOI SEGUIDO DE LA VARIABLE QUE DESEAS CONVERTIR A ENTERO, BASICAMENTE ES PARA CONVERTIR DE CADENA A ENTERO
45                e.id=atoi(token);
46            }
47            else if(campo==4){//SI VA EN EL CAMPO 4 SIGNIFICA QUE ES EL SUELDO Y LO COPIA EN E.SUELDO DE LA ESTRUCTURA, DEBIDO A QUE SON CARACTERES LOS QUE SE LEEN Y ESTAS EN EL SUELDO SE CONVIERTEN A FLOTANTES CON LA FUNCION ATOF SEGUIDO DE LA VARIABLE QUE DESEAS CONVERTIR A FLOTANTE, BASICAMENTE ES PARA CONVERTIR DE CADENA A FLOTANTE
48                e.sueldo=atof(token);
49            }
50            else if(campo==5){
51                strcpy(e.num_cel,token);//SI VA EN EL CAMPO 5 SIGNIFICA QUE ES EL NUMERO DE CELULAR Y LO COPIA EN E.NUM_CEL DE LA ESTRUCTURA
52            }
53            else if(campo==6){
54                e.hrs_sem_trbj=atoi(token);//SI VA EN EL CAMPO 6 SIGNIFICA QUE SON LAS HORAS DE TRABAJO SEMANALES Y LO COPIA EN E.HRS_SEM_TRBJ DE LA ESTRUCTURA, DEBIDO A QUE SON CARACTERES LOS QUE SE LEEN, SE CONVIERTEN A ENTEROS CON LA FUNCION ATOI SEGUIDO DE LA VARIABLE QUE DESEAS CONVERTIR A ENTERO, BASICAMENTE ES PARA CONVERTIR DE CADENA A ENTERO
55            }
56            campo++;//AUMENTAS CAMPO DE 1 EN 1 PARA QUE SE VAYA RECORRIENDO CADA ACAMPO
57            token=strtok(NULL,delimitador);//CUANDO LLEGUE AL ULTIMO TOKEN SERA NULL Y SALE DEL WHILE
58        }
59        empleados[num_empleados]=e;//SE ALMACENA EN EL ARRAY DE EMPLEADOS EN LA POSICION QUE MARQUE NUM_EMPLEADOS EL EMPLEADO QUE SE ACABA DE CARGAR
60        num_empleados++; //AUMENTAS EL NUMERO DE EMPLEADOS DE 1 EN 1
61    }
62 }
63 fclose(f); //CUANDO ALLA LEIDO TODAS LAS LINEAS SE CIERRA EL FICHERO Y DVUELVES EL NUMERO DE EMPLEADOS QUE SE HAN CARGADO
64 *produccion=producc;
65 *finanzas=finan;
66 *ingenieria=ingen;
67 *rechumanos=rechum;
68 *limpieza=limp;
69 *gerencia=geren;
70 *transporte=transp; //Devuelve el valor de los empleados registrados en el archivo
71 return num_empleados; //RETORNA EL NUMERO DE EMPLEADOS
72 }
```

FUNCIÓN PARA GUARDAR EMPLEADOS

```
● ● ●  
1 void guardar_empleados(struct empleado empleados[100], int num_empleados){  
2     FILE *f;  
3     f=fopen("empleados.txt","w");  
4  
5     for(int i=0;i<num_empleados-1;i++){  
6         fprintf(f,"%s;%s;%d%.2f;%s;%d\n",empleados[i].nombre,empleados[i].area,empleados[i].id,empleados[i].sueldo,empleados[i].num_cel,empleados[i].hrs_sem_trbj);  
7     }  
8     fprintf(f,"%s;%s;%d%.2f;%s;%d",empleados[num_empleados-1].nombre,empleados[num_empleados-1].area,empleados[num_empleados-1].id,empleados[num_empleados-1].sueldo,empleados[num_empleados-1].num_cel,empleados[num_empleados-1].hrs_sem_trbj);  
9     fclose(f);  
10 }
```

FUNCIÓN PARA LISTAR LOS EMPLEADOS

```
● ● ●  
1 void listar_empleados(struct empleado empleados[100], int num_empleados){//FUNCIONES QUE ENLISTA LOS EMPLEADOS  
2     int i;  
3     printf("LISTANDO LOS DATOS DEL EMPLEADO\n\n");  
4     for(i = 0; i < num_empleados; i++){  
5         printf("-----\n");  
6         printf("|Nombre del empleado\t | %s\n",empleados[i].nombre);  
7         printf("-----\n");  
8         printf("|Area de trabajo\t | %s\n",empleados[i].area);  
9         printf("-----\n");  
10        printf("|ID del empleado\t | %d\n",empleados[i].id);  
11        printf("-----\n");  
12        printf("|Sueldo percibido\t | %.2f\n",empleados[i].sueldo);  
13        printf("-----\n");  
14        printf("|Numero de telefono \t | %s\n",empleados[i].num_cel);  
15        printf("-----\n");  
16        printf("|Horas por semana\t | %d\n",empleados[i].hrs_sem_trbj);  
17        printf("-----\n");  
18        printf("\n");  
19    }  
20 }
```

FUNCIÓN PARA DAR DE ALTA EMPLEADOS

```
● ● ●
1 int alta_de_empleado(struct empleado empleados[100],int num_empleados,int producc,int finan,int rechum,int ingen,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte){//FUNCION QUE DA DE AL
2     int creado=0;//SE INICIALIZA LA VARIABLE EN 0 PARA INDICAR SI SE HA CREADO O NO
3     struct empleado nuevoemp;//ESTRUCTURA EMPLEADO DONDE SE VAN A CARGAR LOS NUEVOS DATOS DEL EMPLEADO A CREAR
4     fflush(stdin);//SE LIMPIA EL BUFFER PORQUE SE PEDIRAN COSAS
5     printf("\tIntroduzca nombre y apellido del nuevo empleado: ");//SE GUARDA TODO DEL NUEVO EMPLEADO DESDE NOMBRE HASTA HORAS DE TRABAJO
6     gets(nuevoemp.nombre);
7     fflush(stdin);
8     printf("\tIntroduzca el area del nuevo empleado: ");
9     gets(nuevoemp.area);
10    if(strcmp(nuevoemp.area,"Produccion")==0)
11        producc++;
12    else if (strcmp(nuevoemp.area,"Finanzas")==0)
13        finan++;
14    else if (strcmp(nuevoemp.area,"Ingenieria")==0)
15        ingen++;
16    else if (strcmp(nuevoemp.area,"R. Humanos")==0)
17        rechum++;
18    else if (strcmp(nuevoemp.area,"Limpieza")==0)
19        limp++;
20    else if (strcmp(nuevoemp.area,"Gerencia")==0)
21        geren++;
22    else if (strcmp(nuevoemp.area,"Transporte")==0)
23        transp++;
24    fflush(stdin);
25    printf("\tIntroduzca el ID del nuevo empleado: ");
26    scanf("%d",&nuevoemp.id);
27    fflush(stdin);
28    printf("\tIntroduzca el sueldo del nuevo empleado: ");
29    scanf("%f",&nuevoemp.sueldo);
30    fflush(stdin);
31    printf("\tIntroduzca el telefono del nuevo empleado: ");
32    gets(nuevoemp.num_cel);
33    fflush(stdin);
34    printf("\tIntroduzca las horas semanales a trabajar del empleado: ");
35    scanf("%d",&nuevoemp.hrs_sem_trbj);
36
37    int existe=comprobacion_existencia_empleado(empleados,num_empleados,nuevoemp.id);//SE COMPRUEBA SI EXISTE O NO EL ID DEL NUEVO EMPLEADO MANDANDO 1 SI EXISTE O MANDANDO 0 SI NO EXISTE, ASIGNANDOLA EN LA VARIABLE 'EXISTE'
38    if(num_empleados<100){//SI EL NUMERO DE EMPLEADOS ES MENOR AL LIMITE PROSIGUE
39        if(existe==0){//SI NO EXISTE SE CONTINUA
40            empleados[num_empleados]=nuevoemp;//PROcede a GUARDAR EN EL ARREGLO DE EMPLEADOS EN LA POSICION NUM_EMPLEADOS SE GUARDA EL NUEVO EMPLEADO QUE SE ACABA DE CREAR
41            creado=1;//CREADO SE CONVIERTEN EN VERDADERO
42            printf("Empleado registrado con exito.\n");//MENSAJE DE CONFIRAMCION
43        }
44        else{
45            printf("No se puede crear el empleado. El ID ingresado ya existe.\n");//SI EXISTE NO SE PUED Duplicar PORQUE TIENE EL MISMO ID
46        }
47    }
48    else{
49        printf("No se puede dar de alta el empleado. Limite de registros alcanzado.\n");//SI NO NO SE PUEDE PORQUE YA SE ALCANZO EL CUPO
50    }
51    *produccion=producc;
52    *finanzas=finan;
53    *ingenieria=ingen;
54    *rechumanos=rechum;
55    *limpieza=limp;
56    *gerencia=geren;
57    *transporte=transp;
58    return creado;
59 }
```

FUNCIÓN PARA DAR DE BAJA EMPLEADOS

```
● ● ●
1 int baja_de_empleado(struct empleado empleados[100],int num_empleados,int producc,int finan,int rechum,int ingen,int geren,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte){///
2     int eliminado=0;//VARIABLE QUE NOS INDICA SI PUDIMOS ELIMINAR O NO AL EMPLEADO
3     int id, i, j;
4     int indice;
5     printf("\tIntroduza el ID del empleado a dar de baja: ");
6     scanf("%d",&id);
7     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id);
8     if(existe==1){
9         for(i=0;i<num_empleados;i++){
10             if(empleados[i].id==id){
11                 indice=i;//NO SE PUEDE QUEDAR UN ESPACIO EN BLANCO SI SE ELIMINA UN EMPLEADO QUE SE ENCUENTRE ENTRE OTROS ENTONCES UTILIZAMOS LA VARIABLE INDICE IGUALADA A LA POSICION I
12             }
13         }
14         if(strcmp(empleados[indice].area,"Produccion")==0){
15             producc--;
16         }
17         if(strcmp(empleados[indice].area,"Finanzas")==0){
18             finan--;
19         }
20         if(strcmp(empleados[indice].area,"Ingenieria")==0){
21             ingen--;
22         }
23         if(strcmp(empleados[indice].area,"R. Humanos")==0){
24             rechum--;
25         }
26         if(strcmp(empleados[indice].area,"Limpieza")==0){
27             limp--;
28         }
29         if(strcmp(empleados[indice].area,"Gerencia")==0){
30             geren--;
31         }
32         if(strcmp(empleados[indice].area,"Transporte")==0){
33             transp--;
34         }
35         *produccion=producc;
36         *finanzas=finan;
37         *ingenieria=ingen;
38         *rechumanos=rechum;
39         *limpieza=limp;
40         *gerencia=geren;
41         *transporte=transp;
42         for(j=indice;j<num_empleados-1;j++){//CON ESTE CICLO SE EMPIEZA DESDE EL EMPLEADO QUE SE QUIERE ELIMINAR HACIA ADELANTE
43             empleados[j]=empleados[j+1];//PARA CADA EMPLEADO QUE SE ACCEDA EN ESA POSICION J SE GUARDA EL EMPLEADO QUE SE ENCUENTRA EN LA POSICION SIGUIENTE
44             struct empleado aux;//SE CREA UNA ESTRUCTURA EMPLEADO AUXILIAR
45             empleados[j+1]=aux;//LA ESTRUCTURA INICIALIZADA SE METE EN LA POSICION J+1 PARA BORRAR LO QUE TUVIERA LA POSICION J+1 Y METER LA ESTRUCTURA QUE ESTA INICIALIZADA POR DEFECTO
46             //CON ESTE FOR DESPLAZAMOS HACIA LA IZQUIERDA TODOS LOS EMPLEADOS
47         }
48         eliminado=1;
49         printf("Empleado eliminado con exito.\n");
50     }
51 }
52 else{
53     printf("No se puede dar de baja el empleado. No existe el ID ingresado.\n");
54 }
55 return eliminado;
56 }
```

FUNCIÓN PARA CONSULTAR EMPLEADO - GESTOR



```
1 void consultar_empleado_gestor(struct empleado empleados[100],int num_empleados){//funcion que consulta los empleados que el gestor necesita en especifico
2     int id;//variable del id a buscar
3     int i;
4     printf("Introduzca el ID del empleado a consultar porfavor: ");//se imprime en pantalla que ingrese el id del empleado que desea buscar
5     scanf("%d",&id);//se guarda el id a buscar en la variable
6     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id);//compruebas si existe el empleado con la funcion comprobacion_existencia_empleado
7     if(existe==1){//si existe continua
8         for(i=0;i<num_empleados;i++){//con el ciclo for recorres a los empleados existentes
9             if(empleados[i].id==id){//si de los empleados recorridos su id es igual al que se ingreso al incio
10                printf("\nCargando los datos del empleado con ID %d\n\n",empleados[i].id);//entonces imprime los dtos del empleado solicitado
11                printf("-----\n");
12                printf("|Nombre del empleado\t | %s\n",empleados[i].nombre);
13                printf("-----\n");
14                printf("|Area de trabajo\t | %s\n",empleados[i].area);
15                printf("-----\n");
16                printf("|ID del empleado\t | %d\n",empleados[i].id);
17                printf("-----\n");
18                printf("|Sueldo percibido\t | %.2f\n",empleados[i].sueldo);
19                printf("-----\n");
20                printf("|Numero de telefono \t | %s\n",empleados[i].num_cel);
21                printf("-----\n");
22                printf("|Horas por semana\t | %d\n",empleados[i].hrs_sem_trbj);
23                printf("-----\n");
24                printf("\n");
25        }
26    }
27 }
28
29 else{
30     printf("ID no reconocido.EMPLEADO INEXISTENTE\n\n");//si no existe entonces se envia este mensaje
31 }
32 }
```

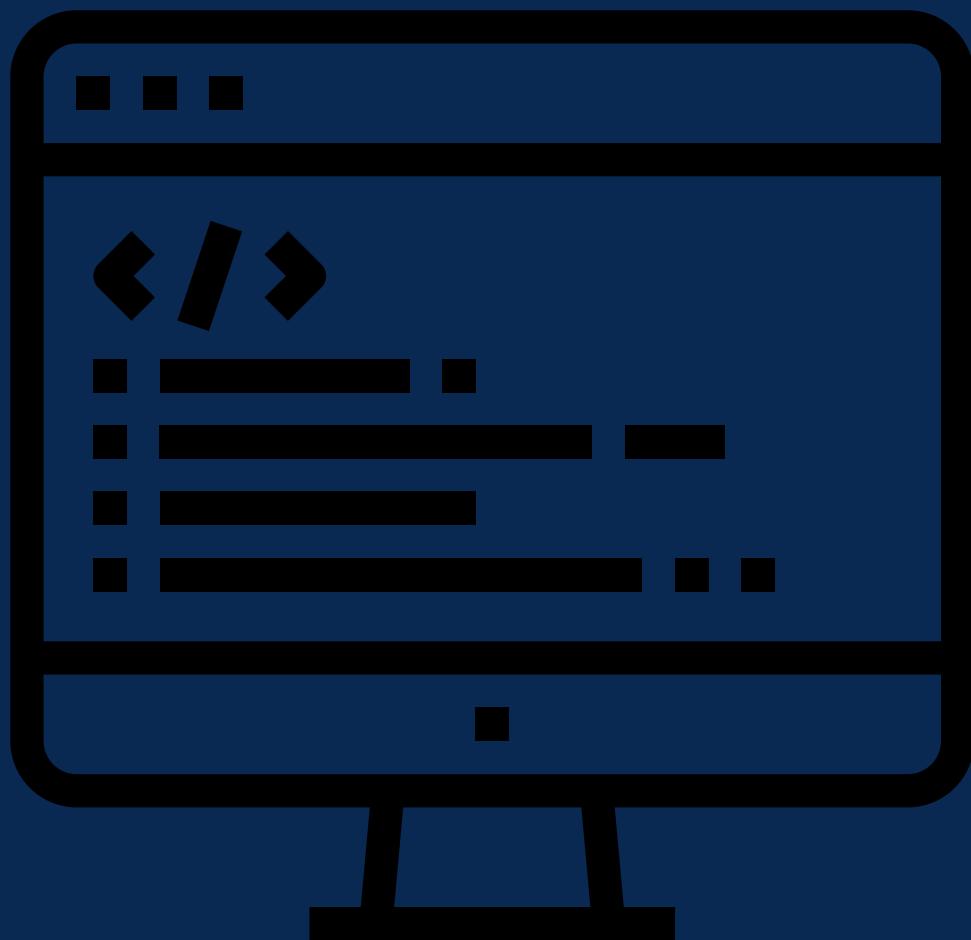
FUNCIÓN PARA CONSULTAR EMPLEADO - EMPLEADOS

```
● ● ●

1 void consultar_empleado_empleado(struct empleado empleados[100],int num_empleados){//funcion que consulta los empleados que el gesto necesita en especifico
2     int id;//variable del id a buscar
3     int i;
4     printf("Dame tu ID de la empresa porfavor: ");//se imprime en pantalla que ingrese el id del empleado que desea buscar
5     scanf("%d",&id);//se guarda el id a buscar en la variable
6     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id);//compruebas si existe el empleado con la funcion comprobacion_existencia_empleado
7     if(existe==1){//si existe continua
8         for(i=0;i<num_empleados;i++){//con el ciclo for recorres a los empleados existentes
9             if(empleados[i].id==id){//si de los empleados recorridos su id es igual al que se ingreso al incio
10                printf("\nCargando los datos del empleado con ID %d\n\n",empleados[i].id);//entonces imprime los dtos del empleado solicitado
11                printf("-----\n");
12                printf("|Nombre del empleado\t | %s\n",empleados[i].nombre);
13                printf("-----\n");
14                printf("|Area de trabajo\t | %s\n",empleados[i].area);
15                printf("-----\n");
16                printf "|ID del empleado\t | %d\n",empleados[i].id);
17                printf("-----\n");
18                printf "|Sueldo percibido\t | %.2f\n",empleados[i].sueldo);
19                printf("-----\n");
20                printf "|Numero de telefono \t | %s\n",empleados[i].num_cel);
21                printf("-----\n");
22                printf "|Horas por semana\t | %d\n",empleados[i].hrs_sem_trbj);
23                printf("-----\n");
24                printf("\n");
25
26        }
27    }
28 }
29 else{
30     printf("\nID no reconocido. Este empleado no esta registrado\n\n");//si no existe entonces se envia este mensaje
31 }
32 printf("Cualquier duda o aclaracion, favor de comunicarse con el administrador.\n");
33 }
```

FUNCTION PARA MODIFICAR ÁREA

```
1 void modificar_area(struct empleado empleados[100],int num_empleados,int producc,int finan,int rechum,int ingen,int limp,int transp,int *produccion,int *finanzas,int *rechumanos,int *ingenieria,int *gerencia,int *limpieza,int *transporte){//FU
2     int id;
3     int i;
4     printf("\tIntroduzca el ID del empleado del cual quiere modificar su numero de celular: ");
5     scanf("%d",&id);
6     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id);
7     if(existe==1){
8         char nueva_area[200];
9         printf("\tIntroduzca la nueva area de trabajo del empleado: ");
10        fflush(stdin);
11        gets(nueva_area);
12        for(i=0;i<num_empleados;i++){
13            if(empleados[i].id==id){
14                if(strcmp(empleados[i].area,"Produccion")==0){
15                    producc--;
16                }
17                if(strcmp(empleados[i].area,"Finanzas")==0){
18                    finan--;
19                }
20                if(strcmp(empleados[i].area,"Ingenieria")==0){
21                    ingen--;
22                }
23                if(strcmp(empleados[i].area,"R. Humanos")==0){
24                    rechum--;
25                }
26                if(strcmp(empleados[i].area,"Limpieza")==0){
27                    limp--;
28                }
29                if(strcmp(empleados[i].area,"Gerencia")==0){
30                    geren--;
31                }
32                if(strcmp(empleados[i].area,"Transporte")==0){
33                    transp--;
34                }
35
36                strcpy(empleados[i].area,nueva_area);
37
38                if(strcmp(empleados[i].area,"Produccion")==0){
39                    producc++;
40                }
41                if(strcmp(empleados[i].area,"Finanzas")==0){
42                    finan++;
43                }
44                if(strcmp(empleados[i].area,"Ingenieria")==0){
45                    ingen++;
46                }
47                if(strcmp(empleados[i].area,"R. Humanos")==0){
48                    rechum++;
49                }
50                if(strcmp(empleados[i].area,"Limpieza")==0){
51                    limp++;
52                }
53                if(strcmp(empleados[i].area,"Gerencia")==0){
54                    geren++;
55                }
56                if(strcmp(empleados[i].area,"Transporte")==0){
57                    transp++;
58                }
59                printf("Area de trabajo actualizada.\n");
60
61                *produccion=producc;
62                *finanzas=finan;
63                *ingenieria=ingen;
64                *rechumanos=rechum;
65                *limpieza=limp;
66                *gerencia=geren;
67                *transporte=transp;
68            }
69        }
70    }
71    else{
72        printf("ID no reconocido.EMPLEADO INEXISTENTE\n\n");
73    }
74 }
```



FUNCIÓN PARA MODIFICAR SUELDO



```
1 void modificar_sueldo(struct empleado empleados [100],int num_empleados){//FUNCION QUE MODIFICA EL SUELDO DE LOS EMPLEADOS
2     int id;
3     int i;
4     printf("\tIntroduzca el ID del empleado del cual quiere modificar su sueldo: ");
5     scanf("%d");//SE INTRODUCE EL ID DEL EMPLEADO
6     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id);//SE UTILIZA LA FUNCION COMPROBACION EXISTENCIA DEL EMPLEADO
7     if(existe==1){//SI LA FUNCION ANTERIOR DA 1 SIGNIFICA QUE SI EXISTE POR LO TANTO SI SE PUEDE MODIFICAR
8         float nuevo_sueldo;
9         printf("\tIntroduzca el nuevo sueldo del empleado: ");
10        scanf("%f");//SE INTRODUCE EL NUEVO SUELDO
11        for(i=0;i<num_empleados;i++){//SE RECORRE EL TOTAL DE EMPLEADOS
12            if(empleados[i].id==id){//CUANDO ENCUENTRA EL MISMO ID
13                empleados[i].sueldo=nuevo_sueldo;//A ESE EMPLEADO SE LE ASIGNA EL NUEVO SUELDO
14                printf("Sueldo actualizado.\n");//IMPRESION DE COMPROBACION QUE SI SE MODIFICO EL SUELDO
15            }
16        }
17    }
18    else{
19        printf("ID no reconocido.No se reconoce al empleado.\n\n");//DE LO CONTRARIO NO EXISTE ENTONCES NO SE PUEDE MODIFICAR
20    }
21 }
```

FUNCIÓN PARA MODIFICAR NÚMERO DE CELULAR



```
1 void modificar_num_cel(struct empleado empleados[100],int num_empleados){//FUNCION QUE MODIFICA EL NUMERO DE CELULAR DE LOS EMPLEADOS
2     int id;
3     int i;
4     printf("\tIntroduzca el ID del empleado del cual quiere modificar su numero de celular: ");
5     scanf("%d",&id);
6     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id);
7     if(existe==1){
8         char nuevo_num_cel[200];
9         printf("\tIntroduzca el nuevo numero de celular del empleado: ");
10        fflush(stdin);
11        gets(nuevo_num_cel);
12        for(i=0;i<num_empleados;i++){
13            if(empleados[i].id==id){
14                strcpy(empleados[i].num_cel,nuevo_num_cel);
15                printf("Numero de celular actualizado.\n");
16            }
17        }
18    }
19    else{
20        printf("ID no reconocido.No se reconoce al empleado.\n\n");
21    }
22 }
```

FUNCIÓN PARA MODIFICAR HORAS DE TRABAJO



```
1 void modificar_hrs_sem_trbj(struct empleado empleados [100],int num_empleados){//FUNCION QUE MODIFICA LAS HORAS DE SEMANA DE TRABAJO
2     int id;
3     int i;
4     printf("\tIntroduzca el ID del empleado del cual quiere modificar su sueldo: ");
5     scanf("%d",&id);//SE INTRODUCE EL ID DEL EMPLEADO
6     int existe=comprobacion_existencia_empleado(empleados,num_empleados,id); //SE UTILIZA LA FUNCION COMPROBACION EXISTENCIA DEL EPMPLEADO
7     if(existe==1){//SI LA FUNCION ANTERIOR DA 1 SIGNIFICA QUE SI EXISTE POR LO TANTO SI SE PUEDE MODIFICAR
8         int nuevo_hrs_sem_trbj;
9         printf("\tIntroduzca las nuevas horas semanales de trabajo : ");
10        scanf("%d",&nuevo_hrs_sem_trbj); //SE INTRODUCE EL NUEVO SUELDO
11        for(i=0;i<num_empleados;i++){//SE RECORRE EL TOTAL DE EMPLEADOS
12            if(empleados[i].id==id){//CUANDO ENCUENTRA EL MISMO ID
13                empleados[i].hrs_sem_trbj=nuevo_hrs_sem_trbj;//A ESE EMPLEADO SE LE ASIGNA EL NUEVO SUELDO
14                printf("Horas de trabajo actualizadas.\n");//IMPRESION DE COMPROBACION QUE SI SE MODIFICO EL SUELDO
15            }
16        }
17    }
18    else{
19        printf("ID no reconocido. No se reconoce al empleado.\n\n");//DE LO CONTRARIO NO EXISTE ENTONCES NO SE PUEDE MODIFICAR
20    }
21 }
```

FUNCIÓN PARA COMPROBAR EXISTENCIA DEL EMPLEADO



```
1 int comprobacion_existencia_empleado(struct empleado empleados[100], int num_empleados, int id){//FUNCION QUE NOS AYUDARA PARA SABER SI EXISTE UN EMPLEADO
2     int resultado=0;//SE DECLARA INICIALMENTE COMO SI NO HUBIERA
3     int i;
4     for(i=0;i<num_empleados;i++){//PASAS EL CICLO PARA RECORRER LOS EMPLEADOS
5         if(empleados[i].id==id){//SE COMPRUEBA SI ES EL MISMO ID EN EL ARREGLO DE ALGUN EMPLEADO CON LO QUE SE INTRODUCE IGUALMENTE QUE SU NOMBRE
6             resultado=1;//DE SER IGUAL RESULTADO ES IGUAL A 1
7         }
8     }
9     return resultado;//REGRESA EL RESULTADO
10 }
```



FUNCIÓN PARA REGISTRO DE CREDENCIALES DEL GESTOR

```
● ● ●
1 int claveUsuario(void){
2     // Variables de contraseña
3     char usuAdm[20], claveAdm[20], claveAdmconf[20], iniusuAdm[20], iniciclaveAdm[20];
4     char nombre[20];
5     int n, i, j, cc=0, lenUsu=0, lenClave=0, flagadmsesion=0;
6
7     // Registro de nuevos datos para administrador
8     // Pedir nombre de usuario
9     printf("\t***** Registro de administrador *****\n\nIngrese su nombre de usuario, debe contener por lo menos 5 caracteres.");
10    // Control de longitud de caracteres
11    while (cc==0){
12        printf("\n\tNombre de usuario: ");
13        scanf("%s", usuAdm);
14        if (strlen(usuAdm)<5){
15            printf("\nNombre de usuario demasiado corto, debe ser de al menos 5 caracteres\n");
16            printf("\nNombre de usuario: ");
17            scanf("%s", usuAdm);
18        } else{
19            cc++;
20        }
21    }
22    // Pedir contraseña y confirmar que sea correcta
23    cc = 0;
24    while (cc==0){
25        printf("\n\tNueva contraseña (Debe ser de 8 - 15 caracteres): ");
26        scanf("%s", claveAdm);
27        n = strlen(claveAdm);
28        while (lenClave==0){
29            if (n<8){
30                system("cls");
31                printf("\nContraseña demasiado corta, debe ser de 8 - 15 caracteres\n");
32                printf("\n\tNueva contraseña: ");
33                scanf("%s", claveAdm);
34                n = strlen(claveAdm);
35            }
36            else if (n>15){
37                system("cls");
38                printf("\nContraseña demasiado larga, debe ser de 8 - 15 caracteres");
39                printf("\n\tNueva contraseña: ");
40                scanf("%s", claveAdm);
41                n = strlen(claveAdm);
42            }
43            else if (n>8 && n<=15){
44                lenClave++;
45            }
46        }
47        printf("\tConfirmar contraseña: ");
48        scanf("%s", claveAdmconf);
49        for (i=0; i<n; i++){
50            if (claveAdm[i]==claveAdmconf[i]){
51                system("cls");
52                printf("\nxx Contraseña no es la misma xx");
53                lenClave=0;
54                break;
55            }
56        }
57        if (i==n){
58            cc++;
59        }
60    }
61    // Se abre archivo para escribir texto
62    FILE *archivo = fopen("credenciales.txt", "w");
63    if (archivo == NULL) {
64        printf("No se pudo abrir el archivo.");
65        return 1;
66    }
67    // Codifica usuario
68    for (i=0; usuAdm[i]!='\0'; i++){
69        char caracterCodificado = usuAdm[i] + 5;
70        fputc(caracterCodificado, archivo);
71    }
72    // Se agrega separación por ;
73    fputc(';', archivo);
74    // Codifica clave
75    for (i=0; claveAdm[i]!='\0'; i++){
76        char caracterCodificado = claveAdm[i] + 5;
77        fputc(caracterCodificado, archivo);
78    }
79    // Cerrar archivo
80    fclose(archivo);
81 }
```

FUNCIÓN PARA LECTURA DE CREDENCIALES DEL GESTOR

```
● ● ●

1 int lecturaCredenciales(char *inusuAdm, char *iniclaveAdm){
2     // Declarar variables
3     int caracterCodificado, i=0;
4     char *token, cadena[100], caracterOriginal;
5     // Lectura de archivo
6     FILE *archivo = fopen("credenciales.txt", "r");
7     if (archivo == NULL) {
8         printf("Error al abrir archivo");
9         return 0;
10    }
11    // Guardar informaci?n en una cadena y decodificarlo
12    while ((caracterCodificado = fgetc(archivo)) != EOF){
13        caracterOriginal = caracterCodificado - 5;
14        cadena[i++]=caracterOriginal;
15    }
16    // Separar informaci?n en diferentes cadenas
17    // Guarda usuario
18    token = strtok(cadena, ";");
19    if (token != NULL){
20        strcpy(inusuAdm, token);
21        // Guarda clave
22        token = strtok(NULL, ";");
23        if (token != NULL){
24            strcpy(iniclaveAdm, token);
25        }
26    }
27    // Se cierra el archivo
28    fclose(archivo);
29    return 0;
30 }
```

FUNCIÓN PARA EL INICIO DE SESIÓN DEL GESTOR



```
1 void inicioSesion(char *iniusuAdm, char *iniclaveAdm){
2     // Declarar variables
3     int i, j, flagadmsesion=0;
4     char usuAdm[50], claveAdm[50];
5     // Inicio de solicitar datos y comparar
6     system("cls");
7     while (flagadmsesion==0){
8         printf("\t--- Inicio de sesion ---\n");
9         printf("\n\tNombre de usuario: ");
10        scanf("%s", usuAdm);
11        printf("\n\tContrasena: ");
12        scanf("%s", claveAdm);
13        for (j=0; j<strlen(usuAdm) ;j++){
14            if (iniusuAdm[j]!=usuAdm[j]){
15                break;
16            }
17        }
18        for (i=0; i<strlen(claveAdm) ;i++){
19            if (iniclaveAdm[i]!=claveAdm[i]){
20                break;
21            }
22        }
23        if (i!=strlen(claveAdm) || j!=strlen(usuAdm)){
24            system("cls");
25            printf("Error: Nombre de usuario o contrasena incorrecto\n\n");
26        }
27        else if (i==strlen(claveAdm) && j==strlen(usuAdm)){
28            flagadmsesion++;
29        }
30    }
31 }
```

FUNCIÓN PARA GRAFICAR

```
● ● ●

1 void graficadora(int grafica[][],int produccion,int finanzas,int rechumanos,int ingenieria,int gerencia,int limpieza,int transporte){
2     for(int i=0;i<50;i++)
3         for(int j=0;j<49;j++)
4             grafica[i][j]=0;
5     for(int i=49;i>=0;i--){ //Inicia desde la fila de hasta abajo
6         for(int j=1;j<49;j++){//Inicia en la primera columna
7             if(j%7!=0){
8                 if(j>0 && j<7 && produccion!=0 && ((49-i)<produccion)){ //Evalua para poder imprimir una barra, llenando las barras con 1
9                     grafica[i][j]=1;
10                }else if (j>7 && j<14 && finanzas!=0 && ((49-i)<finanzas)){
11                    grafica[i][j]=1;
12                }else if (j>14 && j<21 && rechumanos!=0 && ((49-i)<rechumanos)){
13                    grafica[i][j]=1;
14                }else if (j>21 && j<28 && ingenieria!=0 && ((49-i)<ingenieria)){
15                    grafica[i][j]=1;
16                }else if (j>28 && j<35 && gerencia!=0 && ((49-i)<gerencia)){
17                    grafica[i][j]=1;
18                }else if (j>35 && j<42 && limpieza!=0 && ((49-i)<limpieza)){
19                    grafica[i][j]=1;
20                }else if (j>42 && j<49 && transporte!=0 && ((49-i)<transporte)){
21                    grafica[i][j]=1;
22                }
23            }
24        }
25    }
26    for(int i=0;i<50;i++){ //Impresion de la grafica
27        if(50-i>=10)
28            printf("\n\t %d|\t",50-i);
29        else
30            printf("\n\t %d|\t",50-i);
31        for(int j=0;j<49;j++){
32            if(grafica[i][j]!=0) //Impresion de la barra
33                printf("%c",219);
34            else
35                printf(" ");
36        }
37    }
38    printf("\n\t -----");
39    printf("\n\t Prod. Fnzs. R.Hum. Ing. Grnc. Limp. Trans.\n");
40    return;
41 }
42 }
```

CÓDIGO MAIN

VARIABLES & MENSAJE DE INICIO



```
1 int main(){
2     //INICIALIZACION DE VARIABLES
3     int elecc=0;//VARIABLE DE ELECCION PARA EL SWITCH
4     int grafica[50][49]={0}; //Matriz para poder graficar
5     int produccion=0,finanzas=0,rechumanos=0,ingenieria=0,gerencia=0,limpieza=0,transporte=0; //Contabiliza el numero de empleados por area
6     int opc=0;//VARIABLE DE OPCION PARA EL MENU
7     int num_empleados=0;//INCIALIZAS LA VARIABLE DE NUMERO DE EMPLEADOS A CERO PARA PSOTERIORMENTE IGUALARLO AL RESULTADO QUE DA LA FUNCION CARGAR EMPLEADOS
8     struct empleado empleados[100];//ESTRUCTURA DONDE ESTARAN TODOS LOS EMPLEADOS, MAX 100
9     num_empleados=cargarempleados(empleados,produccion,finanzas,rechumanos,ingenieria,gerencia,limpieza,transporte,&produccion,&finanzas,&rechumanos,&ingenieria,&gerencia,&limpieza,&transporte);//
10    int id_empleado;//VARIABLE QUE SE APLICA DENTRO DEL CASE 2 PARA QUE EL USUARIO PUEDA VER LOS DATOS CON SU ID
11    int opc_modificacion_empleado=0;//VARIABLE QUE SE APLICA DENTRO DEL CASE 1 EN LA OPCION 5 PARA QUE ELIGA QUE DESEA MODIFICAR
12    char cntclave, iniusuAdm[20], iniclavAdm[20]; // Variables para funcion de lectura de credenciales
13    FILE *sesion;
14    //MENSAJE DE INICIO
15    printf("      --SoftWork es un programa que brinda apoyo al gestor para tener un manejo dinamico y eficiente de los empleados--\n\n");
16    printf("\n      -----");
17    printf("\n      |Rol en la empresa\t\t\t\t|\n      |\t[1]-Gestor\t\t\t\t\t|\n      |\t[2]-Empleado\t\t\t\t|");
18    printf("\n      -----");
19    printf("\n\tPor favor, seleccione la opcion que desea realizar: ");
20    scanf("%d",&elecc);
21    system("cls");
```

CASO DEL GESTOR

```
1 switch(elecc){
2     case 1://CASO PARA EL GESTOR
3         // Inicio de sesi n
4         sesion = fopen("credenciales.txt", "r");
5         if (sesion == NULL){
6             claveUsuario();
7         }
8         else {
9             fclose(sesion);
10            lecturaCredenciales(iniusuAdm, iniclaveAdm);
11            inicioSesion(iniusuAdm, iniclaveAdm);
12        }
13        system("cls");
14        //MENSAJE DE BIENVENIDA AL GESTOR
15        printf("\tBienvenido a SoftWork!\n\n");
16        mostrarmenu();
17        scanf("%d",&opc);
18
19        while(opc!=6){
20            if(opc==1){
21                system("cls");
22                int creado=alta_de_empleado(empleados,num_empleados,produccion,finanzas,rechumanos,ingenieria,gerencia,limpieza,transporte,&produccion,&finanzas,&rechumanos,&ingenieria,&gerencia,&limpieza,&transporte);
23                if(creado==1){
24                    num_empleados++; //SI CREADO ES IGUAL A 1 SE INCREMENTA EL NUMERO DE EMPLEADOS EXISTENTES EN EL SISTEMA
25                }
26            }
27
28            else if(opc==2){
29                system ("cls");
30                int eliminado=baja_de_empleado(empleados,num_empleados,produccion,finanzas,rechumanos,ingenieria,gerencia,limpieza,transporte,&produccion,&finanzas,&rechumanos,&ingenieria,&gerencia,&limpieza,&transporte);
31                if(eliminado==1){
32                    num_empleados--; //SI ELIMINADO ES IGUAL A 1 SE HA ELIMINADO EL NUMERO DE EMPLEADOS EXISTENTES EN EL SITEMA
33                }
34            }
35
36            else if(opc==3){
37                system("cls");
38                listar_empleados(empleados,num_empleados);
39                graficadora(grafica,produccion,finanzas,rechumanos,ingenieria,gerencia,limpieza,transporte);
40            }
41
42            else if(opc==4){
43                system("cls");
44                consultar_empleado_gestor(empleados,num_empleados);
45            }
46
47            else if(opc==5){
48                system("cls");//NUEVO MENU PARA ELECCION DE MODIFICACION
49                printf("\n      -----\n      |\t[1]-Area\t\t|\t\t|\t\t|\t\t|\n      |\t[2]-Sueldo\t\t|\t\t|\t\t|\t\t|\n      |\t[3]-Celular\t\t|\t\t|\t\t|\t\t|\n      |\t[4]-Horas por semana\t\t|\t\t|\t\t|\n      -----");
50                printf("\tIntroduce la opcion que dese e realizar: ");
51                scanf("%d",&opc_modificacion_empleado);//SE LEE LA OPCION DESEADA
52                switch (opc_modificacion_empleado){//CON SWITCH MANDA A LLAMAR LAS FUNCIONES A ELEGIR
53                    case 1:
54                        modificar_area(empleados,num_empleados,produccion,finanzas,rechumanos,ingenieria,gerencia,limpieza,transporte,&produccion,&finanzas,&rechumanos,&ingenieria,&gerencia,&limpieza,&transporte);
55                        break;
56                    case 2:
57                        modificar_sueldo(empleados,num_empleados);
58                        break;
59                    case 3:
60                        modificar_num_cel(empleados,num_empleados);
61                        break;
62                    case 4:
63                        modificar_hrs_sem_trbj(empleados,num_empleados);
64                        break;
65                    default:
66                        printf("Opcion no valida.\n");
67                }
68            }
69        }
70    }
71
72    else{
73        printf("Opcion no valida.\n");
74    }
75    guardar_empleados(empleados,num_empleados);
76    printf("\n");
77    mostrarmenu();//VUELVE A MOSTRAR EL MENU
78    scanf("%d",&opc);//SE VUELVE A GUARDAR LA NUEVA OPCION DEL MENU
79}
80    system("cls");
81    printf("\tSoftWork le desea un excelente dia. Hasta pronto!\n");//MENSAJE DE SALIDA
82    break;
```

CASO DEL EMPLEADO



```
1 case 2://CASO PARA EL EMPLEADO
2         printf("Bienvenido a SoftWork!\n\n");
3         consultar_empleado_empleado(empleados,num_empleados);
4         break;
5
6     default:
7         printf("No existe la opcion ingresada\n");
8         break;
9     }
10    return 0;
11 }
```

EJEMPLOS DE FUNCIONAMIENTO

--SoftWork es un programa que brinda apoyo al gestor para tener un manejo dinamico y eficiente de los empleados--

```
|Rol en la empresa  
| [1]-Gestor  
| [2]-Empleado
```

Por favor, seleccione la opcion que desea realizar: |

***** Registro de datos del administrador *****

Ingrese su nombre de usuario, debe contener por lo menos 5 caracteres.

Nombre de usuario: Miguel

Nueva contrasenia (Debe ser de 8 - 15 caracteres): resistencia

Confirmar contrasenia: resistencia|

Bienvenido a SoftWork!

```
| Menu del administrador.  
| [1]-Alta empleado  
| [2]-Baja empleado  
| [3]-Listar empleados  
| [4]-Consultar datos de un empleado  
| [5]-Modificar datos del empleado  
| [6]-Salir
```

Introduce la opcion que desee realizar: |

LISTANDO LOS DATOS DEL EMPLEADO

| | |
|---------------------|----------------|
| Nombre del empleado | Miguel Flores |
| Area de trabajo | finanzas |
| ID del empleado | 1 |
| Sueldo percibido | 5000.00 |
| Numero de telefono | 55-25-20-89-97 |
| Horas por semana | 48 |
| Nombre del empleado | Oscar Ayala |
| Area de trabajo | mercadotecnia |
| ID del empleado | 3 |
| Sueldo percibido | 40000.00 |
| Numero de telefono | 55-23-45-77-88 |
| Horas por semana | 40 |
| Nombre del empleado | Angel Sanchez |
| Area de trabajo | ingenieria |
| ID del empleado | 4 |
| Sueldo percibido | 200.00 |
| Numero de telefono | 55-43-56-78-90 |
| Horas por semana | 40 |

Introduzca nombre y apellido del nuevo empleado: Deyanira
Introduzca el area del nuevo empleado: Cocina
Introduzca el ID del nuevo empleado: 9
Introduzca el sueldo del nuevo empleado: 35000
Introduzca el telefono del nuevo empleado: 55-23-12-45-78
Introduzca las horas semanales a trabajar del empleado: 40
Empleado registrado con exito.

Menu del administrador.
[1]-Alta empleado
[2]-Baja empleado
[3]-Listar empleados
[4]-Consultar datos de un empleado
[5]-Modificar datos del empleado
[6]-Salir

Introduce la opcion que deseé realizar: |

Introduza el ID del empleado a dar de baja: 9
Empleado eliminado con exito.

Menu del administrador.
[1]-Alta empleado
[2]-Baja empleado
[3]-Listar empleados
[4]-Consultar datos de un empleado
[5]-Modificar datos del empleado
[6]-Salir

Introduce la opcion que deseé realizar:

Introduzca el ID del empleado a consultar porfavor: 1

Cargando los datos del empleado con ID 1

|Nombre del empleado | Miguel Flores

|Area de trabajo | finanzas

|ID del empleado | 1

|Sueldo percibido | 5000.00

|Numero de telefono | 55-25-20-89-97

|Horas por semana | 48

- | [1]-Area
- | [2]-Sueldo
- | [3]-Celular
- | [4]-Horas por semana

Introduce la opcion que deseé realizar: 2

Introduzca el ID del empleado del cual quiere modificar su sueldo: 1

Introduzca el nuevo sueldo del empleado: 40000

Sueldo actualizado.

Error: Nombre de usuario o contrasenia incorrecto

--- Inicio de sesion ---

Nombre de usuario: roberto

Constrasenia: reciclar

SoftWork le desea un excelente dia. Hasta pronto!

Process returned 0 (0x0) execution time : 8.774 s

Press any key to continue.

CONCLUSIÓN

El programa utiliza una estructura de datos para almacenar la información de los empleados y los cargos existentes, lo que permite una gestión eficiente de los datos. Además, se realizan validaciones de los datos ingresados para asegurar su integridad y se brindan mensajes claros de confirmación o error al administrador.

Gracias a este programa, se simplifica y automatiza el proceso de administración de los empleados, facilitando tareas como la creación, modificación y eliminación de registros de sus datos, así como la visualización de información relevante. Esto puede contribuir a una gestión más eficiente de los recursos humanos, de la organización y a un mejor seguimiento de la información de los empleados.

```
Enter rows and columns for first matrix.  
Enter rows and columns for second matrix.  
Enter elements of matrix 1:  
10  
10  
10  
10  
10  
10  
10  
10  
10  
10  
Enter elements of matrix 2:  
10  
10  
10  
10  
10  
10  
10  
10  
10  
10  
Enter element a" << i + 1 << j + 1 << ":" << endl;  
cout << "Enter element b" << i + 1 << j * ,  
cout << a[i][j];  
cout << endl;
```

MUCHAS GRACIAS



```
render() {
  return (
    <React.Fragment>
      <div className="py-5">
        <div className="container">
          <Title name="our" title="product">
            <div className="row">
              <ProductConsumer>
                {(value) => {
                  console.log(value)
                }}
              </ProductConsumer>
            </div>
          </div>
        </div>
      <React.Fragment>
```