Universidad Tecnológica Nacional

Facultad Regional Tucumán

Algoritmo y Estructura de datos

Trabajo Grupal: Sistema de gestión

Integrantes	Legajo
Bonilla Valentina	49929
Navarro Carolina	50201

Comisión 1K5

Fecha de presentación

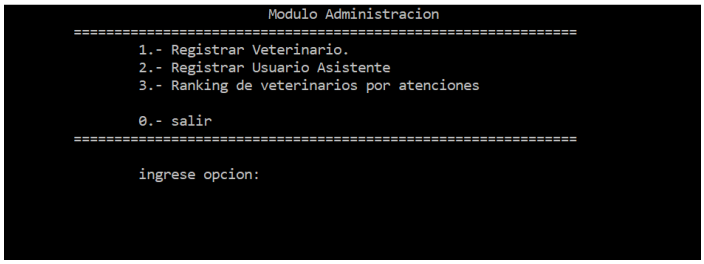
14/12/2020

Modulo Administración

¿En qué consiste el módulo?

Este módulo está dirigido a la gerencia encargada de registrar veterinarios y empleados. El sistema debía contar con las opciones de registrar veterinarios, usuarios asistentes y el ranking de veterinarios según la cantidad de atenciones.

¿Cómo queda nuestro menú de inicio?



¿Cómo codificamos este módulo?

Para la realización de este modulo usamos las siguientes liberarías.

- *#include <stdio.h>:* Contiene las constantes, las declaraciones de funciones de la biblioteca estándar y permite la manipulación de datos de entrada y salida
- *#include <stdlib.h>:* Tiene el prototipo de varias funciones y otras utilidades de uso general.
- *#include <string.h>:* Contiene un conjunto de funciones para manipular cadenas, copiar, cambiar caracteres, comparar cadenas, etc.
- *#include <ctype.h>:* Contiene funciones que permite realizar operaciones con cadenas de caracteres, como por ejemplo convertir un carácter a mayúscula.

Luego de declarar las librerías que necesitaríamos para la realización del programa, procedemos a declarar nuestro propio tipo de dato mediante la palabra clave typedef.

// typedef char cadena [60];

Con “struct” definimos nuestras estructuras: Veterinario, turno, mascota, usuario, entre otras que nos serán necesarias. Como se muestra a continuación estas serán algunas de las estructuras que usaremos.

```
struct Dia
{
    cadena NombreDia;
    bool trabaja;
    int turnos;
};


struct veterinario
{
    cadena apeynom;
    cadena usuarios;
    Dia DiaTrabajo[6];
    int matricula;
    cadena contrasenia;
    cadena Telefono;
};

struct usuarios
{
    cadena usuario;
    cadena contrasenia;
    cadena apeynom;
};
```

En una función con tipo definimos nuestro menú, con un do-while.

```
int menu()
{
    int opc;
    do
    {
        system("cls");
        printf("                    Modulo Administracion\n");
        printf("                ============================================================\n");
        printf("            1.- Registrar Veterinario.\n");
        printf("            2.- Registrar Usuario Asistente\n");

        printf("            3.- Ranking de veterinarios por atenciones\n\n");
        printf("            0.- salir\n");
        printf("                ============================================================\n");
        printf("\n");
        printf("                    ingrese opcion: ");
        scanf("%d", &opc);
    }while(opc<0 && opc>5);
    return opc;
}
```

Utilizaremos la función bool para validar el ingreso de sesión de un veterinario a través de la matricula.

```
87  // veterinario
88  bool buscarVeterinario(int objetivo)
89  {
90      veterinario regi;
91      FILE *archi;
92      bool resp=false;
93      archi=fopen("Veterinarios.dat","rb");
94      rewind(archi);
95      fread(&regi,sizeof(regi),1,archi);
96      while(!feof(archi))
97      {
98          if(objetivo==regi.matricula)
99          {
100             resp=true;
101         }
102         fread(&regi,sizeof(regi),1,archi);
103     }
104     fclose(archi);
105     return resp;
106 }
```

líneas

91. Declaramos el archivo

92. Declaramos como falsa la respuesta

93. Abrimos el archivo de veterinario, utilizamos “rb” porque lo abre un archivo en modo binario para lectura

94. Usamos rewind, que rebobina y sitúa el cursor de lectura/escritura al principio del archivo.

95- Utilizamos fread ya que esta función permite leer datos de un archivo abierto con la función fopen (), a partir de la posición indicada por el puntero de lectura/escritura.

95. Utilizamos un while, y con la función de feof que es de extracción de archivo.

Luego volveremos a utilizar una función bool para validar que la contraseña cumpla con los requisitos necesarios, también otra para validar usuario y para buscar el usuario.

A través de una función sin tipo, registraremos al veterinario y otra para registrar usuario asistente.

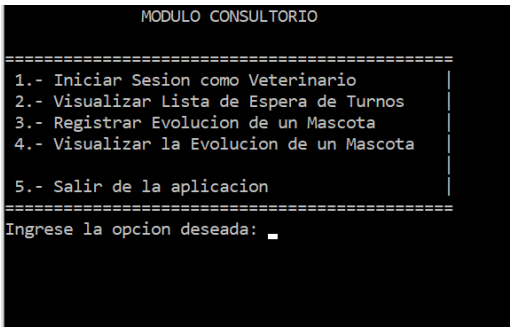
Luego a través de una función sin tipo, comprobaremos el ranking de veterinarios por atencion.

Modulo Consultorio

¿En qué consiste este módulo?

Este modulo esta dirigido a los profesionales que manejan la base de datos de la veterinaria. Se debe pedir un ingreso de sesión, la lista de turnos y la evolución de mascota.

¿Cómo queda el meno de este módulo?



¿Cómo codificamos este módulo?

- *#include <stdio.h>:* Contiene las constantes, las declaraciones de funciones de la biblioteca estándar y permite la manipulación de datos de entrada y salida
- *#include <stdlib.h>:* Tiene el prototipo de varias funciones y otras utilidades de uso general.
- *#include <string.h>:* Contiene un conjunto de funciones para manipular cadenas, copiar, cambiar caracteres, comparar cadenas, etc.
- #include <conio.h>: Contiene principalmente prototipos de funciones como getche () y getch ().

Luego de declarar las librerías que necesitaríamos para la realización del programa, procedemos a declarar nuestro propio tipo de dato mediante la palabra clave typedef.

// typedef char cadena [60];

Con “struct” definimos nuestras estructuras: Veterinario, turno, mascota, usuario, entre otras que nos serán necesarias. Como se muestra a continuación estas serán algunas de las estructuras que usaremos.

```
struct Dia
{
    cadena60 NombreDia;

};

struct RegSesionV
{
    cadena60 AyN;
    cadena60 Telefono;
    Dia dias[6];
    int matricula;
    cadena60 contrasenia;
};

struct Turnos
{
    int matricula;
    cadena60 dia;
    int dni;
    cadena60 Detalle;
    cadena60 usuario;
    bool activo;
};
```

A través de una función sin tipo declaramos nuestro menú.

```
void MostrarMenu()
{
    printf("              MODULO CONSULTORIO");
    printf("\n");
    printf("\n================================================");
    printf("\n 1.- Iniciar Sesion como Veterinario          |");
    printf("\n 2.- Visualizar Lista de Espera de Turnos     |");
    printf("\n 3.- Registrar Evolucion de un Mascota        |");
    printf("\n 4.- Visualizar la Evolucion de un Mascota    |");
    printf("\n                                              |");
    printf("\n 5.- Salir de la aplicacion                   |");
    printf("\n================================================");
}
```

A diferencia del módulo administración, en este módulo no usaremos funciones y la llamemos en los cases correspondientes del switch case. Si no, que desarrollaremos en cada case correspondiente.

Declaramos main (); y comenzamos a declarar archivos y variables necesarias. Abrimos los archivos que necesitaremos y construimos un if en caso de que los archivos no puedan abrirse.

```
archie=fopen("Evoluciones.dat", "a+b");
if(archie==NULL)
{
    printf("No se pudo abrir el archivo de Evoluciones...");
    printf("\n");
    system("pause");
    exit(1);
}
fclose(archie);
```

Con un switch case, empezamos a abrir los casos.

En el correspondiente al caso 1, en este case debemos comprobar el inicio de sesión. Con nuestras variables declaradas, usamos un while para habilitar el inicio de sesión, si la contraseña es correcta y damos 5 intentos para ingresar la contraseña.

En el caso 2, si antes no inicio sesión se le avisa al usuario que no puede proseguir hasta ingresar sesión de manera correspondiente. Una vez ingresado el usuario podrá ver los turnos próximos.

En el correspondiente al caso 3, si antes no inicio sesión se le avisa al usuario que no puede proseguir hasta ingresar sesión de manera correspondiente. El veterinario ingresara la evolución de la mascota, ingresando el DNI del dueño de la mascota, como se muestra a continuación.

```
while(!feof(archie))
{
    if(dnibuscar==mascota.dni)
    {
        banderadni=1;
        printf("Ingrese la evolucion de la mascotas: ");
        _flushall();
        mascota.dni=dnibuscar;
        gets(mascota.evolucion);
        fseek(archie, sizeof(mascota), SEEK_CUR);
        fwrite(&mascota, sizeof(mascota), 1, archie);
        rewind(archit);
        fread(&turno, sizeof(turno), 1, archit);
        while(!feof(archit))
        {
        if((dnibuscar==turno.dni) && (mat==turno.matricula))
        {
            turno.activo=false;
            fseek(archit, sizeof(turno), SEEK_CUR);
            fwrite(&turno, sizeof(turno), 1, archit);
            break;
        }
        fread(&turno, sizeof(turno), 1, archit);
        }
        printf("Se ha almacenado la evolucion del paciente!");
        printf("\n");
        system("pause");
```

En el caso 4, si antes no inicio sesión se le avisa al usuario que no puede proseguir hasta ingresar sesión de manera correspondiente. El veterinario podrá visualizar la evolución de la mascota.

Luego en caso de que el usuario no ingrese una acción correspondiente, se usa default:

Para terminar con este módulo, procedemos a cerrar los archivos.

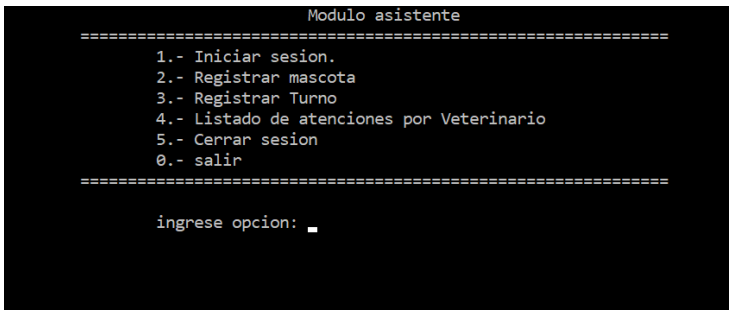
```
    default:
            {
            system("cls");
            printf("\nCODIGO NO EXISTENTE");
            printf("\n");
            system("pause");

            break;
            }
}
system("cls");
MostrarMenu();

printf("\nIngrese la opcion deseada: ");
scanf("%d", &cod);
```

Modulo recepción

¿En qué consiste el módulo de recepción?

Este módulo está dirigido para el personal que asiste a los veterinarios y se encarga de la atención al público. Se debe pedir el ingreso de sesión, los asistentes registrarán turnos y mascota y podrán ver el listado de atención de los veterinarios.

¿Cómo queda el menú principal?



¿Cómo codificamos el módulo?

Para la realización de este módulo usamos las siguientes liberarías.

- *#include <stdio.h>:* Contiene las constantes, las declaraciones de funciones de la biblioteca estándar y permite la manipulación de datos de entrada y salida
- *#include <stdlib.h>:* Tiene el prototipo de varias funciones y otras utilidades de uso general.
- *#include <string.h>:* Contiene un conjunto de funciones para manipular cadenas, copiar, cambiar caracteres, comparar cadenas, etc.
- *#include <ctype.h>:* Contiene funciones que permite realizar operaciones con cadenas de caracteres, como por ejemplo convertir un carácter a mayúscula.

Luego de declarar las librerías que necesitaríamos para la realización del programa, procedemos a declarar nuestro propio tipo de dato mediante la palabra clave typedef.

// typedef char cadena [60];

Con “struct” definimos nuestras estructuras: Veterinario, turno, mascota, usuario, entre otras que nos serán necesarias. Como se muestra a continuación estas serán algunas de las estructuras que usaremos.

```
struct fecha
{
    int dia;
    int mes;
    int anio;
};

struct usuarios
{
    cadena usuario;
    cadena contrasenia;
    cadena apeynom;
};


struct Mascotas
{
    cadena apeynom;
    cadena domicilio;
    int DNI;
    cadena localidad;
    fecha nacimiento;
    float peso;
    cadena Telefono;
};
```

En una función con tipo definimos nuestro menú, con un do-while.

```
int menu()
{
    int opc;
    do
    {
        system("cls");
        printf("                    Modulo asistente\n");
        printf("                ============================================================\n");
        printf("            1.- Iniciar sesion.\n");
        printf("            2.- Registrar mascota\n");
        printf("            3.- Registrar Turno\n");
        printf("            4.- Listado de atenciones por Veterinario\n");
        printf("            5.- Cerrar sesion\n");
        printf("            0.- salir\n");
        printf("                ============================================================\n");
        printf("\n");
        printf("                    ingrese opcion: ");
        scanf("%d", &opc);
    }while(opc<0 && opc>5);
    return opc;

}
```

Con una función bool, validaremos el inicio de sesión, también utilizaremos una para buscar la mascota y veterinario.

A través de una función sin tipo, registraremos las mascotas y los turnos.

```
    system("cls");
    printf("                Registrar Mascota\n");
    printf(" ============================================================\n");
    printf("DNI: ");
    scanf("%d", &regi.DNI);
}
while(buscarMascota(regi.DNI)==true);
printf("\nApellido y Nombre: ");
_flushall();
gets(regi.apeynom);
strlwr(regi.apeynom);
printf("\ndomicilio: ");
_flushall();
gets(regi.domicilio);
printf("\nlocalidad: ");
_flushall();
gets(regi.localidad);
printf("\nfecha de nacimiento(dd/mm/aa): \ndia: ");
do
{
    scanf("%d", &regi.nacimiento.dia);
}
while(regi.nacimiento.dia<1 && regi.nacimiento.dia>31);
printf("\nmes: ");
do
{
    scanf("%d", &regi.nacimiento.mes);
}
while(regi.nacimiento.mes<1 && regi.nacimiento.mes>12);
printf("\nanio: ");
do
{
    scanf("%d", &regi.nacimiento.anio);
}
while(regi.nacimiento.anio<0 && regi.nacimiento.anio>99);
printf("\npeso: ");
scanf("%f", &regi.peso);
printf("Telefono: ");
_flushall();
gets(regi.Telefono);
system("cls");
archi=fopen("Mascotas.dat", "a+b");
```

Volveremos a utilizar una función bool para validar las mascotas y para validar los días de turnos de los veterinarios.

Para poder ingresar al listado de atenciones, al igual que en las otras opciones pedimos el ingreso de un usuario y la verificación del mismo. Una vez ingresado, utilizamos un switch case:

```
system("cls");
printf("                Listado atenciones\n");
printf(" ============================================================\n");
printf("\n1.listado de atenciones por Veterinario");
printf("\n0.volver");
printf("\n\nopcion: ");
scanf("%d", &opc);
```

Y en el “case 1” se mostrarán la cantidad de turnos de los veterinarios.

Al igual que en el módulo de administración nuestro “main ()” está formado por un switch case y por funciones que serán llamadas en cada caso según le corresponda.

Y cerraremos el programa de la siguiente manera:

```
        case 5:
                if(IS==0)
                {
                    system("cls");
                    printf("no se inicio sesion aun!!\n\n");
                    system("pause");
                    break;
                }
                else
                {
                    system("cls");
                    IS=0;
                    printf("la sesion se cerro con exito\n\n");
                    system("pause");
                    break;
                }
        }
    }
    while(opc!=0);
}
```