Instituto Politécnico Nacional
Escuela Superior de Computo

“Reporte Proyecto Final”

Equipo ARMY EDD

-Cervantes Martínez José Alberto

-Guerrero Pérez Diana Estefanía

-Navarro Topete José Alberto

-Rodea Azcona Erick Alberto

-Santos Méndez Ulises Jesús

Estructura de Datos

1CM1

Índice

Introducción

A continuación en este documento vamos a explicar todo el procedimiento que hicimos para hacer nuestro proyecto, desde el por qué lo escogimos, lo que vamos a ocupar para poder programarlo, como funciona, las consideraciones que tiene nuestro programa y demostraremos el funcionamiento de esta.

Justificación

Elegimos este proyecto ya que nos pareció atractiva y un poco desafiante para programarlo, ya que para hacer este tipo de cosas si necesitas analizar bien las preguntas que se le va a hacer al usuario y que estas sean preguntas cerradas para que podamos hacer el promedio de las respuestas y que estas estén almacenadas y que cuando el usuario quiera ver los resultados, los pueda buscar rápido y sin ningún problema, también para ver qué tipos de estructura de datos servirían en este tipo de caso que nosotros planteamos y que también pueda servir en un caso de trabajo y que gracias a este proyecto tengamos la facilidad de hacer este tipo de programas.

Descripción

Nuestro proyecto consiste en hacer un programa que haga un estudio demográfico sobre diferentes usuarios para saber si prefieren gatos o perros.

Para hacer este programa vamos a ocupar *"pilas dinámicas y árbol binario"*, primero vamos a sacar información que necesitamos del usuario y para obtener esa información vamos a hacerle un cuestionario y las preguntar para la información son las siguientes:

- Nombre.
- Sexo.
- Edad.
- Trabaja.
- ¿Tiene hijos?
- ¿De dónde es?
- ¿Qué prefiere gatos o perros?
- ¿Está cuidando de un gato o un perro?

Estas preguntas son obligatorias que tiene que contestar la persona que lo haga y en la última pegunta, si el usuario responde que si está cuidando a un gato o perro, podrá contestar las siguientes preguntas:

- ¿El gato o perro es comprado o es adoptado?
- ¿El gato o perro es de raza o es mestizo?
- ¿Cuántos gatos o perros está cuidando?

Entonces después de que el usuario ingreso sus respuestas, estas las vamos a almacenar en un "*árbol binario*" y dependiendo de su respuesta esta se acomodara en lado izquierdo o derecho del árbol y con la "*pila dinámica*" vamos a obtener la información que necesitamos y a hacer el promedio de: edad, de cuantos prefieren gatos o los perros, si tienen hijos o no, de cuantos están al cuidado de un gato o un perro y cuantos están cuidando, de cuantos son de raza o mestizos y de cuantos son comprados o adoptados.

Para hacer la demostración de que funciona nuestro proyecto, vamos a hacer por lo menos 200 datos, estos los vamos a hacer tanto manual como radom.

Funcionamiento

Primero declaramos nuestras estructuras que vamos a ocupa en nuestro programa

```
typedef struct REGISTRO{
    char nombre[20];
    char apellido[20];
    char sexo[9];
    int edad;
    char estado[50];
    char trabaja[3];
    char hijos[3];
    char preferencia_perros_gatos[8];
    char esta_cuidando[3];
    char adoptado_comprado[9];
    char raza_mestizo[8];
    int cantidad_mascotas;
}registro;
```

```
typedef struct NODO_PILA{
    registro *regis;
    struct NODO_PILA *nodo_abajo;
}nodo_pila;

typedef struct PILA{
    nodo_pila *nodo_tope;
}pila;

typedef struct DPST{
    registro *dato;
    struct DPST *ds;
    struct DPST *is;
}Nodo;

typedef struct INR{
    int nelementos;
    struct DPST *raiz;
}Indicador;
```

A continuación le explicare las siguientes funciones que ocupa nuestro programa:

```
registro* PedirDatos(){

    int res;

    registro *nuevo_registro = (registro*)malloc(sizeof(registro));

    if(nuevo_registro == NULL){
        printf("ERROR, ya no hay memoria para un registro nuevo\n");
    }else{
        printf("Ingrese el nombre: "); scanf(" %s", nuevo_registro->nombre);

        printf("Ingrese el apellido: "); scanf(" %s", nuevo_registro->apellido);

        printf("Ingrese el sexo: "); scanf(" %s", nuevo_registro->sexo);

        printf("Ingrese su edad: "); scanf(" %d",&nuevo_registro->edad);

        printf("De donde es ? : "); scanf(" %s",nuevo_registro->estado);

        printf("Trabaja ? : "); scanf(" %s", nuevo_registro->trabaja);

        printf("Tiene hijos ? : "); scanf(" %s", nuevo_registro->hijos);

        printf("Prefiere perros o gatos ? : "); scanf(" %s", nuevo_registro->preferencia_perros_gatos);

        printf("Esta cuidando a algun perro o gato ? : "); scanf(" %s", nuevo_registro->esta_cuidando);


        if(strcmp(nuevo_registro->esta_cuidando, "Si") == 0){
            system("cls");
            printf("Es adoptado o comprado?: "); scanf(" %s",nuevo_registro->adoptado_comprado);
            printf("Es de raza o mestizo? : "); scanf(" %s", nuevo_registro->raza_mestizo);
            printf("Cuantos perros o gatos estas cuidando en total? : "); scanf(" %d",&nuevo_registro->cantida
        }else{
            nuevo_registro->cantidad_mascotas = 0;
        }

        return nuevo_registro;
    }
}
```

Esta función nos permite pedir los datos al usuario que nosotros ocupamos para hacer el promedio y para guardar la información en el árbol, y también hay una condición, que si esta persona está cuidando un perro o un gato, y si responde que "si" (en minúsculas) ,le lanzara las siguientes preguntas.

```
void imprimirRegistro(registro *reg){
    printf("Nombre -> %s",reg->nombre);
    printf("\nApellido -> %s",reg->apellido);
    printf("\nSexo -> %s",reg->sexo);
    printf("\nEdad -> %d",reg->edad);
    printf("\nEstado -> %s",reg->estado);
    printf("\nTrabaja -> %s",reg->trabaja);
    printf("\nHijos -> %s",reg->hijos);
    printf("\nPreferencia perros o gatos -> %s",reg->preferencia_perros_gatos);
    printf("\nEsta cuidando a algun perro o gato -> %s", reg->esta_cuidando);

    if(strcmp(reg->esta_cuidando, "Si") == 0){
        printf("\nAdoptado o comprado -> %s",reg->adoptado_comprado);
        printf("\nRaza o mestizo -> %s",reg->raza_mestizo);
        printf("\nNumero de mascotas -> %d",reg->cantidad_mascotas);
    }

    printf("\n\n");
}
```

Esta función nos ayuda a imprimir los datos previamente ingresados por el usuario para que este vea que se ingresó correctamente.

```
int ValorRandom(int n, int mi, int ma){
    srand(n);
    return (rand()%(ma-mi))+mi;
}
```

En esta función nos permite generar aleatoriamente los unos datos previamente ingresados en el programa, que a continuación le mostraremos como usamos esta función.

```
registro* Automatico(int num1, int num2){
    registro *nuevo_registro = (registro*)malloc(sizeof(registro));
    srand(time(NULL));
    int ma = 20;
    int mi = 0;
    char *nombres[20]={
        "Jose", "Juan", "Pablo", "Edgar", "Alejandro", "Esteban", "Erick", "Victor", "Alberto", "Joaquin",
        "Maria", "Lucia", "Mercedes", "Lorena", "Jimena", "Veronica", "Ester", "Johana", "Guadalupe", "Jessica"
    };
    char *apellidos[20]={
        "Rodriguez", "Hernandez", "Garcia", "Martinez", "Lopez", "Gonzales", "Sanchez", "Ramirez", "Perez",
        "Gomez", "Fernandez", "Diaz", "Torres", "Ruiz", "Sosa", "Alvarez", "Benites", "Acosta", "Herrera", "Aguirre"
    };
    int n = rand()%num1;
    //nuevo_registro->nombre = nombres[ValorRandom(n, mi, ma)];
    strcpy(nuevo_registro->nombre,nombres[ValorRandom(n, mi, ma)]);
    //printf("%s\n",nuevo_registro->nombre);
```

En esta función vamos a generar los datos aleatorios que ya vienen en el programa y como se muestra en la imagen primero ponemos los datos en las variables y después se va a generar el nombre.

```
if(ValorRandom(n, mi, ma) >= 0 && ValorRandom(n, mi, ma) <= 9)
{
    //nuevo_registro->sexo = "Masculino";
    strcpy(nuevo_registro->sexo,"Masculino");
}
else
{
    //nuevo_registro->sexo = "Femenino";
    strcpy(nuevo_registro->sexo,"Femenino");
}
```

Después se va a generar si es masculino o femenino. Y si el nombre sale que esta en el rango de 0 a 9, significa que es masculino y si es mayor o igual a 9 significa que es femenino.

```
n = rand()%num2;
//nuevo_registro->apellido = apellidos[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->apellido,apellidos[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->apellido);
```

En esta parte se generará el apellido.

```
ma = 70;
mi = 18;
n = rand()%num2;
nuevo_registro->edad = ValorRandom(n, mi, ma);
//printf("%d\n",nuevo_registro->edad);
mi = 0;
```

En esta parte se generara la edad, pero tuvimos que hacerle de un rango para que no fuera tan descabellado la edad de la persona que crea el programa, que este rango es de 18 a 70 año, que está definida en las variables "mi" y "ma".

```
ma = 32;
char *estados[32]={
    "Aguascalientes", "Baja California", "Baja California Sur", "Campeche", "Coahuila", "Colima", "Chiapas", "Chihuahua"
    "Cuidad de Mexico", "Durango", "Guanajuato", "Guerrero", "Hidalgo", "Jalisco", "Estado de Mexico", "Michoacan",
    "Morelos", "Nayarit", "Nuevo Leon", "Oaxaca", "Puebla", "Queretaro", "Quintana Roo", "San Luis Potosi", "Sinaloa",
    "Sonora", "Tabasco", "Tamaulipas", "Tlaxcala", "Veracruz", "Yucatan", "Zacatecas"
};
n = (rand()%num2)+ma;
//nuevo_registro->estado = estados[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->estado,estados[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->estado);
```

En esta parte se genera los estados de donde proviene la persona que crea el programa, pusimos los 32 estados de la República Mexicana.

```
ma = 2;
char *respuesta[2]={"Si", "No"};
n = rand()%num2;
//nuevo_registro->trabaja = respuesta[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->trabaja,respuesta[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->trabaja);
```

En este parte se genera si la persona que crea el programa trabaja.

```
ma = 2;
char *hijos[2]={"Si", "No"};
n = rand()%num2;
//nuevo_registro->hijos = hijos[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->hijos,hijos[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->hijos);
```

En esta parte solo genera si tiene hijo o no la persona.

```
ma = 2;
char *prefer[2]={"Perros", "Gatos"};
n = rand()%num2;
//nuevo_registro->preferencia_perros_gatos = prefer[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->preferencia_perros_gatos,prefer[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->preferencia_perros_gatos);
```

En esta parte el programa va a decir si la persona que creo prefiere a los gatos o perros.

```
ma = 2;
char *cuid[2]={"Si", "No"};
n = rand()%num2;
//nuevo_registro->esta_cuidando = cuid[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->esta_cuidando,cuid[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->esta_cuidando);
```

En esta parte cuando de un perro o un gato.

```
ma = 2;
char *ad[2] = {"Adoptado", "Comprado"};
n = rand()%num2;
//nuevo_registro->adoptado_comprado = ad[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->adoptado_comprado,ad[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->adoptado_comprado);
```

En esta parte elegirá si es adoptado o comprado, que anteriormente explicamos de la condición cuando pedimos los datos a la persona.

```
ma = 2;
char *rame[2] = {"Raza", "Mestizo"};
n = rand()%num2;
//nuevo_registro->raza_mestizo = rame[ValorRandom(n, mi, ma)];
strcpy(nuevo_registro->raza_mestizo,rame[ValorRandom(n, mi, ma)]);
//printf("%s\n",nuevo_registro->raza_mestizo);
```

En esta parte se usa para ver si el gato o el perro es mestizo o de raza.

```
ma = 4;
mi = 1;
n = rand()%num2;
nuevo_registro->cantidad_mascotas = ValorRandom(n, mi, ma);
//printf("%d\n",nuevo_registro->cantidad_mascotas);
//sleep(10);
return nuevo_registro;
```

Y por último este se dirá cuántas mascotas está cuidando la persona creada por el programa.

```c
void ini_stack(pila *stack){
    stack->nodo_tope = NULL;
}

void push_pila(pila *stack, registro *nuevo_registro){

    nodo_pila *nuevo_nodo = (nodo_pila*)malloc(sizeof(nodo_pila));

    if(nuevo_nodo != NULL){
        nuevo_nodo->regis = nuevo_registro;
        nuevo_nodo->nodo_abajo = stack->nodo_tope;
        stack->nodo_tope = nuevo_nodo;
    }else{
        printf("ERROR. Ya no hay memoria para insertar mas registros.\n");
    }

}
```

En esta parte hay dos funciones, la primera función sirve para ver crear la pila, y la segunda función sirve para añadir los registros que previamente se ingresaron.

```c
nodo_pila* pop_pila(pila *stack){

    nodo_pila *nodo_fuera; nodo_pila *aux;

    if(es_vacia(stack) == 1){
        printf("ERROR. No hay elementos en la pila\n");
    }else{
        nodo_fuera = stack->nodo_tope;
        aux = stack->nodo_tope;
        stack->nodo_tope = stack->nodo_tope->nodo_abajo;
        free(aux);
    }
    return nodo_fuera;
}
```

EL funcionamiento de esta función es que lee el último dato ingresado y lo saca de la pila.

```c
int es_vacia(pila *stack){
    if(stack->nodo_tope == NULL){
        return 1;
    }else{
        return 0;
    }
}
```

Esta función solo nos indica si la pila esta vacía.

```c
void imprimir_stack(pila *stack){

    if(es_vacia(stack) == 1){
        printf("ERROR. No hay registros en la pila para imprimir.\n");
    }else{
        nodo_pila *aux = stack->nodo_tope;
        while(aux != NULL){
            imprimirRegistro(aux->regis);
            aux = aux->nodo_abajo;
        }
    }
}
```

En esta función nos imprimirá todo el contenido que tenga la pila almacenada en ella.

```c
void eliminar_pila(pila *stack){
    ini_stack(stack);
}
```

Y esta función solo sirve para eliminar la pila cuando esta se le pide que lo haga.

```c
void llenarDatos(pila *stack, int cantidad){
    int control;
    int num1 = 400;
    int num2 = 400;
    for(control = 1; control <= cantidad; control++){
        push_pila(stack, Automatico(num1,num2));
        num1--;
        num2 = num2 - 2;
        //printf("registro %d\n",control);
    }
}
```

Y por último en cuestión de pilas, esta función lo que hace es llenar la pila con los datos radoms que genero el programa.

En esta parte vamos a crear el árbol y los elementos que va a llevar

```
Indicador* CrearArbol(Indicador *tree){
    tree=(Indicador*)malloc(sizeof(Indicador));
    tree->nelementos=0;
    tree->raiz=NULL;
    return tree;
}

Nodo* CrearNodo(registro *elemento){
    Nodo *rama;
    rama=(Nodo*)malloc(sizeof(Nodo));
    rama->dato=elemento;
    rama->ds=NULL;
    rama->is=NULL;
    return rama;
}
```

En esta parte la primera función de creamos el árbol y en la segunda función vamos a crear los nodos u hojas, que al principio está vacío. Pero más adelante vamos a ocupar la función.

```
void InsertarElemento(Indicador *tree, registro *elemento){
    Nodo *rama, *comp;
    if(tree->nelementos==0){
        rama=CrearNodo(elemento);
        tree->raiz=rama;
        tree->nelementos++;
    }
    else{
        rama=CrearNodo(elemento);
        comp=tree->raiz;
        if(strcmp((rama->dato)->preferencia_perros_gatos,(comp->dato)->preferencia_perros_gatos)!=0){
            if(comp->ds!=NULL){
                comp=comp->ds;
                do{
                    if(strcmp((rama->dato)->sexo,(comp->dato)->sexo)!=0){
                        if(comp->ds!=NULL){
                            comp=comp->ds;
                        }
                        else{
                            comp->ds=rama;
                            tree->nelementos++;
                            comp=NULL;
                        }
                    }
                    else{
                        if(comp->is!=NULL){
                            comp=comp->is;
                        }
```

```
                    else{
                        comp->is=rama;
                        tree->nelementos++;
                        comp=NULL;
                    }
                }
            }while(comp!=NULL);
        }
        else{
            comp->ds=rama;
            tree->nelementos++;
        }
    }
    else{
        if(comp->is!=NULL){
            comp=comp->is;
            do{
                if(strcmp((rama->dato)->sexo,(comp->dato)->sexo)!=0){
                    if(comp->ds!=NULL){
                        comp=comp->ds;
                    }
                    else{
                        comp->ds=rama;
                        tree->nelementos++;
                        comp=NULL;
                    }
                }
                else{
                    if(comp->is!=NULL){
                        comp=comp->is;
                    }
                    else{
                        comp->is=rama;
                        tree->nelementos++;
                        comp=NULL;
                    }
                }
            }while(comp!=NULL);
        }
        else{
            comp->is=rama;
            tree->nelementos++;
        }
    }
}
return;
}
```

En esta parte de la función Insertamos los datos de la pila al árbol, que la primera condición compara la preferencia entre gato y perro del "nodo padre" en el árbol con respecto al nuevo nodo que se desea introducir, si la preferencia es la misma, es decir que al nodo padre y el nuevo nodo prefieren ambos gato o perro, el nuevo nodo será hijo izquierdo del nodo padre, en caso contrario será hijo derecho. Eso sí solo contamos el primer nivel del árbol, es decir, la raíz y sus respectivos hijos. En dado caso que los hijos de la raíz se encuentren definidos, ahora se compara el sexo del nuevo nodo a introducir y el nodo padre que en este caso puede ser cualquiera que se encuentre después de la raíz, si el sexo del nodo padre y el nuevo nodo es el mismo, será hijo izquierdo pero en caso contrario será hijo derecho.

Las siguientes imágenes son funciones de búsqueda de datos en el árbol, pero algunas funciones tienen un contador para cuando el usuario pida una información de un cierto dato, vea todos los registros que esta almacenada en el árbol.

```
void busquedaNombreApellido(Nodo *tree, char nombre[], char apellido[]){
    if(tree!=NULL){
        busquedaNombreApellido(tree->is,nombre,apellido);
        if(strcmp(tree->dato->nombre,nombre) == 0 && strcmp(tree->dato->apellido,apellido) == 0){
            printf("\n"); imprimirRegistro(tree->dato);
        }
        busquedaNombreApellido(tree->ds,nombre,apellido);
    }
    return;
}
```

```
void busquedaSexo(Nodo *tree, char sexo[]){
    if(tree!=NULL){
        busquedaSexo(tree->is,sexo);
        if(strcmp(tree->dato->sexo,sexo) == 0){
            printf("\n"); imprimirRegistro(tree->dato);
        }
        busquedaSexo(tree->ds,sexo);
    }
    return;
}
```

t

```
void busquedaEdad(Nodo *tree, int edad){
    int cont = 0;
    if(tree!=NULL){
        busquedaEdad(tree->is,edad);
        if(tree->dato->edad == edad){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaEdad(tree->ds,edad);
    }

    return;
}
```

```c
void busquedaEstado(Nodo *tree, char estado[]){
    int cont = 0;
    if(tree!=NULL)
        busquedaEstado(tree->is,estado);
        if(strcmp(tree->dato->estado,estado) == 0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaEstado(tree->ds,estado);
    }

    return;
}
```

```c
void busquedaTrabaja(Nodo *tree, char trabaja[]){
    int cont = 0;
    if(tree!=NULL){
        busquedaTrabaja(tree->is,trabaja);
        if(strcmp(tree->dato->trabaja,trabaja) == 0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaTrabaja(tree->ds,trabaja);
    }

    return;
}
```

```c
void busquedaHijos(Nodo *tree, char hijos[]){
    int cont = 0;
    if(tree!=NULL){
        busquedaHijos(tree->is,hijos);
        if(strcmp(tree->dato->hijos,hijos) == 0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaHijos(tree->ds,hijos);
    }

    return;
}
```

```c
void busquedaPreferencias(Nodo *tree, char preferencias[]){
    int cont = 0;
    if(tree!=NULL){
        busquedaPreferencias(tree->is,preferencias);
        if(strcmp(tree->dato->preferencia_perros_gatos,preferencias) == 0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaPreferencias(tree->ds,preferencias);
    }

    return;
}
```

```c
void busquedaCuidando(Nodo *tree, char cuidando[]){
    int cont = 0;
    if(tree!=NULL){
        busquedaCuidando(tree->is,cuidando);
        if(strcmp(tree->dato->esta_cuidando,cuidando) == 0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaCuidando(tree->ds,cuidando);
    }

    return;
}
```

```c
void busquedaAdopComp(Nodo *tree, char ac[]){
    int cont = 0;
    if(tree!=NULL){
        busquedaAdopComp(tree->is,ac);
        if(strcmp(tree->dato->adoptado_comprado,ac) == 0 && strcmp(tree->dato->esta_cuidando,"Si")==0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaAdopComp(tree->ds,ac);
    }

    return;
}
```

```c
void busquedaRM(Nodo *tree, char rm[]){
    int cont = 0;
    if(tree!=NULL){
        busquedaRM(tree->is,rm);
        if(strcmp(tree->dato->raza_mestizo,rm) == 0 && strcmp(tree->dato->esta_cuidando,"Si")==0){
            printf("\n"); imprimirRegistro(tree->dato); cont++;
        }
        busquedaRM(tree->ds,rm);
    }

    return;
}
```

```c
void busquedaCM(Nodo *tree, int cm){
    if(tree != NULL){
        busquedaCM(tree->is,cm);
        if(tree->dato->cantidad_mascotas == cm && strcmp(tree->dato->esta_cuidando,"Si")==0){
        //printf("%d",tree->dato->cantidad_mascotas);
            printf("\n"); imprimirRegistro(tree->dato);
        }
        busquedaCM(tree->ds,cm);
    }
    return;
}
```

En las siguientes imágenes la primera función sirve para buscar los datos que necesite la persona y como ya había mencionado le mostrara todos los registro que se ingresó al árbol y la otra función es solo el menú que se le mostrara el usuario al momento de que se termine de ingresar los datos y que le llevara a la función de buscar datos.

```c
void buscarDatosGeneral(Indicador *tree, int opcion){
    switch(opcion){
        case 1: printf("Opcion 1 -> Busqueda por nombre y apellido\n");
                char nombre[20]; char apellido[20];
                printf("Introduce el nombre -> "); scanf(" %s",nombre);
                printf("Introduce el apellido -> "); scanf(" %s",apellido);
                busquedaNombreApellido(tree->raiz,nombre,apellido);
                break;
        case 2: printf("Opcion 2 -> Busqueda por sexo\n"); char sexo[9];
                printf("Se mostraran todos los registros segun el sexo que ingreses\n");
                printf("Introduce el sexo a buscar -> "); scanf(" %s",sexo);
                busquedaSexo(tree->raiz,sexo);
                break;
        case 3: printf("Opcion 3 -> Edad\n"); int edad;
                printf("Se mostraran todos los registros segun la edad que ingreses\n");
                printf("Introduce la edad a buscar -> "); scanf(" %d",&edad);
                busquedaEdad(tree->raiz,edad);
                break;
        case 4: printf("Opcion 4 -> Estado\n"); char estado[50];
                printf("Se mostraran todos los registros segun el estado que ingreses\n");
                printf("Introduce el estado a buscar -> "); scanf(" %s",estado);
                busquedaEstado(tree->raiz,estado);
                break;
        case 5: printf("Opcion 5 -> Trabaja ?\n"); char trabaja[3];
                printf("Se mostraran todos los registros segun si la persona trabaja o no.\n");
                printf("Introduzca 'No' o 'Si' segun quiera obtener los registros -> "); scanf(" %s",trabaja);
                busquedaTrabaja(tree->raiz,trabaja);
                break;
```

```
        case 6: printf("Opcion 6 -> Hijos\n"); char hijos[3];
                printf("Se mostraran todos los registros segun si la persona tiene hijos o no.\n");
                printf("Introduzca 'No' o 'Si' segun quiera obtener los registros -> "); scanf(" %s",hijos);
                busquedaHijos(tree->raiz,hijos);
                break;
        case 7: printf("Opcion 7 -> Preferencia\n"); char preferencias[8];
                printf("Se mostraran todos los registros segun la preferencia de animal de la persona.\n");
                printf("Introduzca 'Perro' o 'Gato' segun quiera obtener los registros -> "); scanf(" %s",preferencias);
                busquedaPreferencias(tree->raiz,preferencias);
                break;
        case 8: printf("Opcion 8 -> Esta Cuidadndo\n"); char cuidando[3];
                printf("Se mostraran todos los registros segun si la persona esta o no cuidando a un perro o gato.\n");
                printf("Introduzca 'No' o 'Si' segun quiera obtener los registros -> "); scanf(" %s",cuidando);
                busquedaCuidando(tree->raiz,cuidando);
                break;
        case 9: printf("Opcion 9 -> Adoptado o Comprado\n"); char ac[9];
                printf("Se mostraran todos los registros segun si la persona adopto o compro a su mascota.\n");
                printf("Introduzca 'Adoptado' o 'Comprado' segun quiera obtener los registros -> "); scanf(" %s",ac);
                busquedaAdopComp(tree->raiz,ac);
                break;
        case 10: printf("Opcion 10 -> Raza o Mestizo\n"); char rm[8];
                 printf("Se mostraran todos los registros segun si la mascota de la persona es de raza o mestizo.\n");
                 printf("Introduzca 'Adoptado' o 'Comprado' segun quiera obtener los registros -> "); scanf(" %s",rm);
                 busquedaRM(tree->raiz,rm);
                 break;
        case 11: printf("Opcion 11 -> Cantidad de mascotas\n"); int cm;
                 printf("Se mostraran todos los registros segun la cantidad de mascotas de la persona.\n");
                 printf("Introduzca el numero de mascotas -> "); scanf(" %d",&cm);
                 busquedaCM(tree->raiz,cm);
                 break;
        default: printf("Esa opcion no esta disponible\n");
    }
}
```

```
void menubuscarDatos(Indicador *tree){

    int opcion;

    printf("\nCon respecto a que atributo del registro desea buscar ?\n");
    printf("1.- Nombre y Apellido\n");;
    printf("2.- Sexo\n");
    printf("3.- Edad\n");
    printf("4.- Estado\n");
    printf("5.- Trabaja ?\n");
    printf("6.- Hijos\n");
    printf("7.- Preferencias perros o gatos\n");
    printf("8.- Esta cuidando ?\n");
    printf("9.- Adoptado o comprado\n");
    printf("10.- Raza o mestizo\n");
    printf("11.- Cantidad de mascotas\n");
    scanf("%d",&opcion); buscarDatosGeneral(tree,opcion);
}
```

```c
void funcionar(pila *stack){

    char op = 's';

    while(op == 's'){
        system("cls");
        push_pila(stack,PedirDatos());
        printf("Continuas -> "); scanf(" %c",&op);
    }
    system("cls");
    imprimir_stack(stack);
}
```

Esta función solo sirve para que podamos usar la pila que anteriormente vimos sus elementos.

```c
void insertar(pila *stack, Indicador *Arbol){
    nodo_pila *aux = pop_pila(stack); int cont = 0;
    while(aux != NULL){
        InsertarElemento(Arbol,aux->regis);
        cont++; printf("%d\n",cont);
        aux = pop_pila(stack);
    }

}
```

En esta función nos servirá para contar los registros que hay en la pila y el arbol.

```
void promediar(pila *stack, Indicador *Arbol, int ref){

    nodo_pila *aux;
    float prom=0;
    float sum_adop=0;
    float sum_comp=0;
    float sum_qmas=0;
    float sum_pprros=0;
    float sum_pgatos=0;
    float sum_pmas=0;
    float sum_raza=0;
    float sum_mest=0;
    char a_c= 'A';
    char a_R= 'R';
    char a_P= 'P';
    float prom_adop=0;
    float prom_comp=0;
    float prom_pprros=0;
    float prom_pgatos=0;
    float prom_qprros=0;
    float prom_qgatos=0;
    float prom_raza=0;
    float prom_mestizo=0;
    int cont = 0;

    aux = pop_pila(stack);
```

La siguiente función nos servirá para poder promediar las respuestas que las personas hayan contestado el cuestionario, primero declaramos las variables donde se guarda el promedio de cada dato.

```
    if(ref>185){
        while(cont != 185){

            if(strchr(aux->regis->adoptado_comprado,a_c) != NULL){
                sum_adop++;
            }else{
                sum_comp++;
            }

            if(strchr(aux->regis->raza_mestizo,a_R) != NULL){
                sum_raza++;
            }else{
                sum_mest++;
            }

            if(strchr(aux->regis->preferencia_perros_gatos,a_P) != NULL){
                sum_pprros++;
            }else{
                sum_pgatos++;
            }
            InsertarElemento(Arbol,aux->regis);
            cont++;
            //printf("%d\n",cont);
            aux = pop_pila(stack);
            //printf("%d\n",cont);
        }
    }
```

```
    else{
        while(cont != ref){

            if(strchr(aux->regis->adoptado_comprado,a_c) != NULL){
                sum_adop++;
            }else{
                sum_comp++;
            }

            if(strchr(aux->regis->raza_mestizo,a_R) != NULL){
                sum_raza++;
            }else{
                sum_mest++;
            }

            if(strchr(aux->regis->preferencia_perros_gatos,a_P) != NULL){
                sum_pprros++;
            }else{
                sum_pgatos++;
            }
            InsertarElemento(Arbol,aux->regis);
            cont++;
            //printf("%d\n",cont);
            aux = pop_pila(stack);
            //printf("%d\n",cont);
        }
    }
```

Las imágenes anteriores son condiciones para comprobar si este es el dato coincide con la condición declarada, este se estará contando.

```
    prom_adop=(sum_adop/cont)*(100.00);
    printf("\n El promedio de mascotas adoptados: %.2f",prom_adop);
    prom_comp=(sum_comp/cont)*(100.00);
    printf("\n El promedio de mascotas compradas: %.2f",prom_comp);
    prom_raza=(sum_raza/cont)*(100.00);
    printf("\n El promedio de mascotas de raza: %.2f",prom_raza);
    prom_mestizo=(sum_mest/cont)*(100.00);
    printf("\n El promedio de mascotas mestizas: %.2f",prom_mestizo);
    prom_pprros=(sum_pprros/cont)*(100.00);
    printf("\n El promedio de preferencia por perros: %.2f",prom_pprros);
    prom_pgatos=(sum_pgatos/cont)*(100.00);
    printf("\n El promedio de preferencia por gatos: %.2f",prom_pgatos);
    return;
}
```

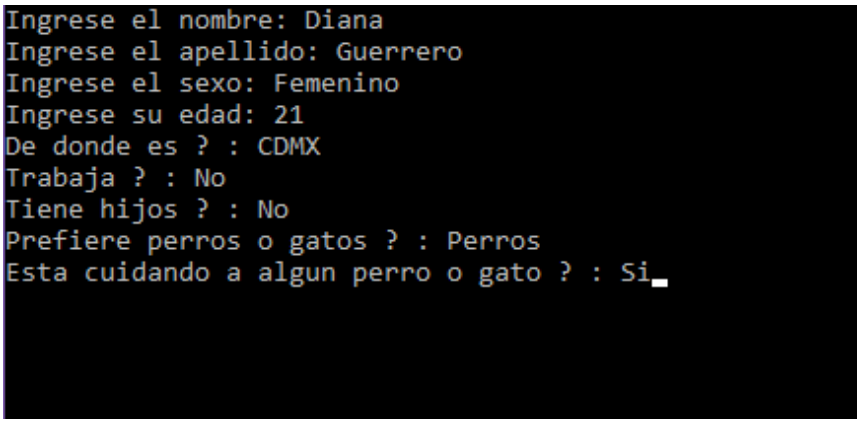
Después de que el programa haya revisado todos los registros, este le lanzara en pantalla los promedios.

```c
int main(){
    pila stack; char opcion = 's';
    Indicador *Arbol;
    Arbol=CrearArbol(Arbol);
    ini_stack(&stack);
    //funcionar(&stack);
    llenarDatos(&stack,147);
    funcionar(&stack);
    imprimir_stack(&stack);

     //insertar(&stack,Arbol);
    promediar(&stack,Arbol,150);
    //Inorden(Arbol->raiz);
    //Inorden(Arbol->raiz);
    while(opcion == 's'){
        menubuscarDatos(Arbol);
        printf("Deseas seguir haciendo busquedas ? -> "); scanf(" %c",&opcion);
    }


    return 0;
}
```

Y por último la función Main, nos servirá para que nuestras funciones que creamos puedan correr al momento de ejecutemos nuestro programa y al momento de que se termine de ejecutar todo el programa le preguntara si quiere hacer más búsqueda y si la persona pone "s", este le volverá a mostrar el menú de buscar los datos.



En la imagen que se mostro anterior mente ya es el programa ejecutado, la persona contestara las preguntas que se muestran (Se recomienda que al momento de ingresar los datos la primera letra sea en Mayúscula y lo demás sea en minúscula y en caso de que sea de la Ciudad de México poner CDMX) y cuando llegue a la última pregunta.

```
Es adoptado o comprado?: Adoptado
Es de raza o mestizo? : Mestizo
Cuantos perros o gatos estas cuidando en total? : 2
Continuas -> _
```

Si la respuesta es "Sí" (se recomienda poner mayúscula en la primera letra y lo demás en minúscula), le lanzara las preguntas que se muestran en la imagen (al momento de ingresar los datos se recomienda poner mayúscula en la primera letra y lo demás en minúscula), y la ultima pregunta si el usuario contesta que si (cuando este esta parte se recomienda que solo ponga la letra s en minúscula), este lo que hará es que vuelva aparecer las preguntas que les mostramos al principio, y en caso de que no quiere poner más datos tendrá que poner "n" (cuando este esta parte se recomienda que solo ponga la letra n en minúscula).

```
Nombre -> Diana
Apellido -> Guerrero
Sexo -> Femenino
Edad -> 21
Estado -> CDMX
Trabaja -> No
Hijos -> No
Preferencia perros o gatos -> Perros
Esta cuidando a algun perro o gato -> Si
Adoptado o comprado -> Adoptado
Raza o mestizo -> Mestizo
Numero de mascotas -> 2

Nombre -> Mercedes
Apellido -> Hernandez
Sexo -> Femenino
Edad -> 59
Estado -> Nuevo Leon
Trabaja -> Si
Hijos -> No
Preferencia perros o gatos -> Gatos
Esta cuidando a algun perro o gato -> No

Nombre -> Veronica
Apellido -> Herrera
Sexo -> Femenino
Edad -> 66
Estado -> Michoacan
Trabaja -> Si
Hijos -> Si
Preferencia perros o gatos -> Perros
Esta cuidando a algun perro o gato -> Si
Adoptado o comprado -> Adoptado
Raza o mestizo -> Raza
Numero de mascotas -> 3
```

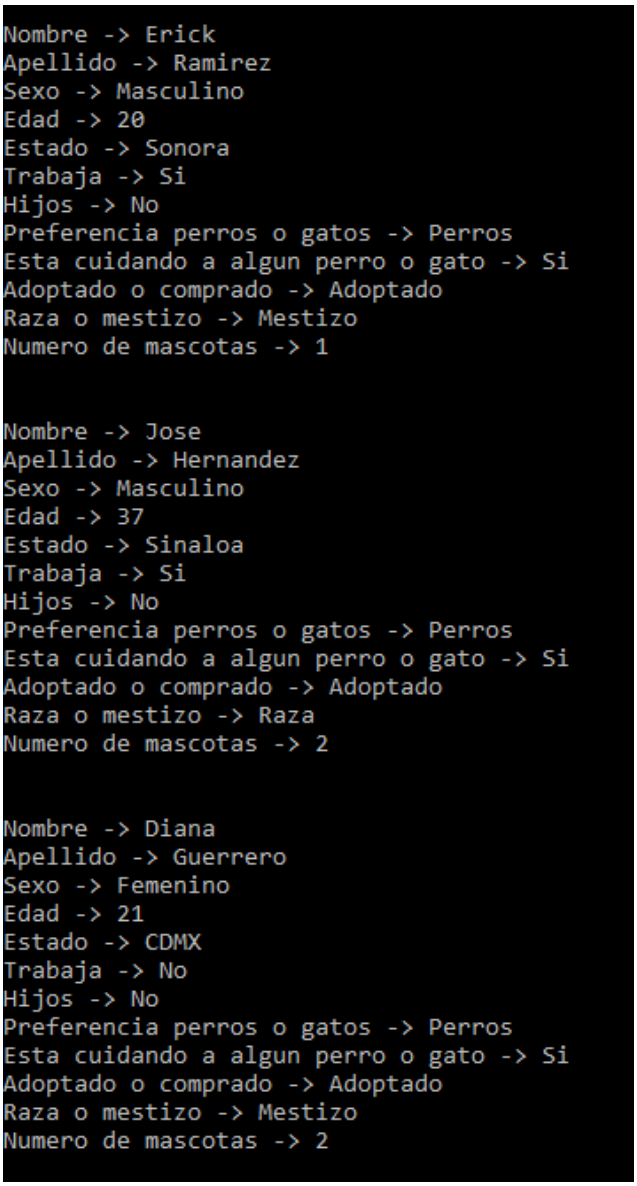
Lo que hará es mostrarle todos los datos que la persona ingreso y los datos aleatorios que creo el programa.

```
Promedios:
 El promedio de mascotas adoptados: 56.76
 El promedio de mascotas compradas: 43.24
 El promedio de mascotas de raza: 56.76
 El promedio de mascotas mestizas: 43.24
 El promedio de preferencia por perros: 49.73
 El promedio de preferencia por gatos: 50.27
```

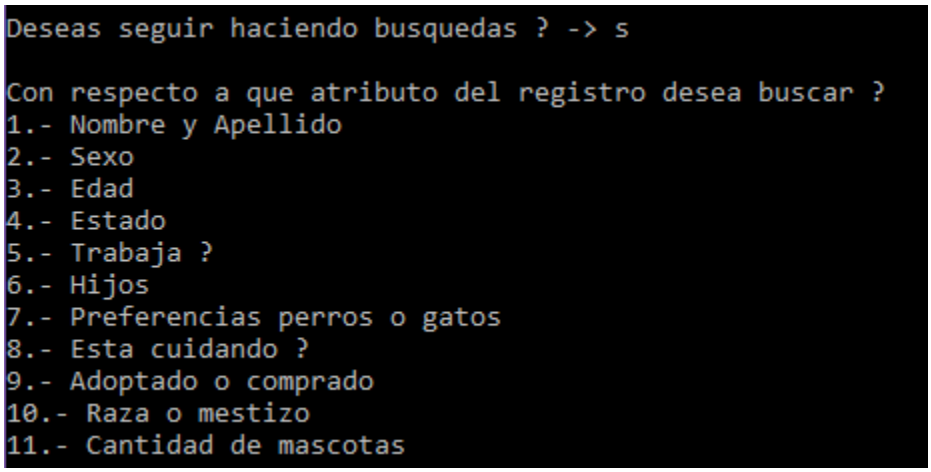
Cuando se termina de crear los datos de las personas, este se mostrará los promedios de las preguntas que hicimos.

```
Con respecto a que atributo del registro desea buscar ?
1.- Nombre y Apellido
2.- Sexo
3.- Edad
4.- Estado
5.- Trabaja ?
6.- Hijos
7.- Preferencias perros o gatos
8.- Esta cuidando ?
9.- Adoptado o comprado
10.- Raza o mestizo
11.- Cantidad de mascotas
7
Opcion 7 -> Preferencia
Se mostraran todos los registros segun la preferencia de animal de la persona.
Introduzca 'Perro' o 'Gato' segun quiera obtener los registros -> _
```

En esta parte se mostrará después de que se muestre los promedios, lo que hará que busque los datos que necesite la persona, escoge la opción y enseguida le preguntara si lo que quiere es una opción u otra que en este caso se eligió la 7 y me dice que si quiere Perro o Gato (al momento de ingresar la opción que desea se recomienda que la primera letra sea en Mayúscula y lo demás en minúscula y que lo copie tal cual está en el programa)

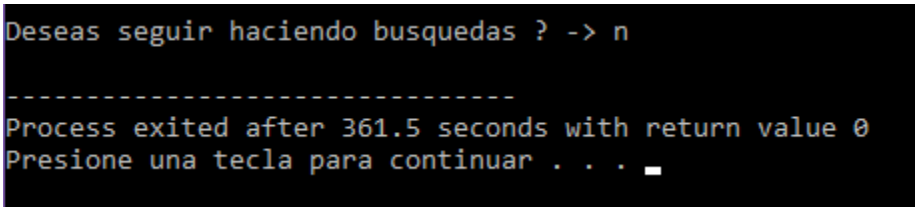


Y al momento de poner la respuesta esta le lanzara todos los datos que desea ver.



Y cuando termine de ejecutar los datos este le preguntará que si quiere hacer otra búsqueda, en caso de que si quiera tendrá que poner "s"

y le salta de nuevo el menú para ver qué datos está buscando.



En caso de que ya no quiera buscar nada lo único que tiene que poner es "n" y este lo sacara del programa

Conclusiones

Al trabajar con este proyecto nos dimos cuenta de algunas cosas que si bien en un principio se presentaron algunos problemas con lo que respecta a la aplicación de alguna estructura para obtener los datos de cada participante, el promedio de la encuesta y el respaldo de los datos obtenidos, en base al conocimiento adquirido por parte de la clase de teoría y el consejo de nuestro profesor, logramos identificar dos estructuras cuyas características cumplen de forma eficiente las demandas presentes en el desarrollo del programa.

Las estructuras que utilizamos para este proyecto muestran nuestro conocimiento para crearlas, usarlas y aplicarlas a estructurar datos. El uso de una pila para la transición de datos a otra estructura y el análisis de estos en el intermedio de esta acción, asimismo el uso de un árbol binario para la organización de la información y desglose de está en base a un dato en específico, en este caso la edad.

En conclusión se hizo manejo de distintos TAD que se lograron visualizar en clase, se hace uso de distintas funciones dedicadas a generar diversos datos así como el de ordenarlos, también se hizo uso de un TAD eficiente de árbol para la búsqueda de los datos generados, entonces se hizo una aplicación diversa de los temas vistos con éxito.

También gracias a este proyecto nos pudimos dar cuenta de que podemos lograr cualquier cosa que nos propagamos y algunas veces es mejor trabajar en equipo para que salga adelante el proyecto que tengamos ya que una sola persona puede tener dificultad en algunos temas y con más persona se pueda resolver rápido y con mayor facilidad las problemáticas.