

Manual de programador

Programación 1

Cuenca Esquivel Ana Karen – Esquivel Correa Juan Francisco

Índice

Introducción	2
Objetivo	2
Fundamento teórico	2
Requerimientos del Sistema	3
Estructura del código	3
Función main o principal.....	5
void gotoxy(int x, int y)	5
int selector(int x,int y,int n)	5
void bordes(int opc)	6
void gatos(void).....	6
void cls(void)	6
void encabezado()	7
void presentación ()	7
void menu()	8
int NumeroDatos(float nums[]).....	12
void lectura(float nums[],int n)	13
void ordenar(float nums[], int n)	15
float mediana(float nums[],int n).....	17
float media(float nums[],int n)	18
float moda(float nums[],int n).....	19
float rango(float nums[],int n).....	20
float varianza(float med,float nums[],int n)	21
float desvar(float var)	22
float coefvar(float med, float desvari)	22
int numInt(int n).....	23
int anchInt(int inter, float rang).....	23
void grafica(float nums[],int n).....	23

Introducción

La estadística es la Estudio que reúne, clasifica y recuenta todos los hechos que tienen una determinada característica en común, para poder llegar a conclusiones a partir de los datos numéricos extraídos.

Una vez que se tienen los datos clasificados y organizados se pueden calcular medidas de tendencia central básicas como la media, mediana y moda; medidas de variabilidad como el rango, varianza, desviación estándar y coeficiente de variabilidad. En su conjunto estas medidas nos indican diversas características de la muestra de datos con lo que podemos inferir en base a probabilidades el comportamiento de estos.

Los cálculos necesarios para tener estas medidas pueden ser muy laboriosos, sobre todo si el número de los datos es masivo, dificultando los cálculos y haciendo el margen de error de estos más grande.

El almacenamiento de estos datos y el cálculo de los estadísticos se puede llevar a cabo de manera muy práctica utilizando las estructuras, arreglos, funciones, archivos, sentencias de control y operadores matemáticos de un lenguaje de propósito general como lenguaje C.

Objetivo

Aplicar los conceptos de programación estructurada de lenguaje C en el cálculo de medidas de tendencia central, medidas de variabilidad y grafica de frecuencias para una muestra de datos almacenada en un archivo de texto (archivo plano). De este archivo deberá obtener los datos y realizar los cálculos mediante programación.

Fundamento teórico

Descripción de las medidas de tendencia central

Media aritmética. - Es el promedio de datos de una muestra de n datos, se puede calcular de la siguiente manera:

Mediana. -La mediana es un valor de la variable que deja por debajo de sí a la mitad de los datos, una vez que éstos están ordenados de menor a mayor. En caso de un número par de datos, la mediana no correspondería a ningún valor de la variable, por lo que se conviene en tomar como mediana el valor intermedio entre los dos valores centrales.

Moda. - La moda es el dato más repetido de la encuesta, el valor de la variable con mayor frecuencia absoluta. Su cálculo es extremadamente sencillo, pues solo necesita un recuento. En variables continuas, expresadas en intervalos, existe el denominado intervalo modal o, en su defecto, si es necesario obtener un valor concreto de la variable, se recurre a la interpolación.

Descripción de las medidas de variabilidad

Rango. - El rango o recorrido estadístico es la diferencia entre el valor máximo y el valor mínimo en un grupo de números aleatorios. Se le suele simbolizar con R . Sus únicos dos requisitos son: ordenamos los números según su tamaño y restamos el valor mínimo del valor máximo

Varianza. - La varianza es una medida estadística que mide la dispersión de los valores respecto a un valor central (media), es decir, es el cuadrado de las desviaciones.

Desviación estándar. - La varianza a veces no se interpreta claramente, ya que se mide en unidades cuadráticas. Para evitar ese problema se define otra medida de dispersión, que es la desviación típica, o desviación estándar, que se halla como la raíz cuadrada positiva de la varianza. La desviación típica informa sobre la dispersión de los datos respecto al valor de la media; cuanto mayor sea su valor, más dispersos estarán los datos.

Coeficiente de variación. - En estadística, cuando se desea hacer referencia a la relación entre el tamaño de la media y la variabilidad de la variable, se utiliza el coeficiente de variación.

Su fórmula expresa la desviación estándar como porcentaje de la media aritmética, mostrando una mejor interpretación porcentual del grado de variabilidad que la desviación típica o estándar.

Armado de la gráfica frecuencias para datos agrupados

Cuando los datos contienen una gran cantidad de elementos, para facilitar los cálculos es necesario agruparlos, a estos grupos se los llama intervalos o clases. Un intervalo es una serie de números incluidos entre dos extremos.

Las reglas generales para formas distribuciones de frecuencias para datos agrupados en intervalos son:

- 1) Calcule el Rango (R). - También se llama recorrido o amplitud total. Es la diferencia entre el valor mayor y el menor de los datos.
- 2) Seleccione el Número de Intervalos de Clase (ni).- No debe ser menor de 5 y mayor de 12, ya que un número mayor o menor de clases podría oscurecer el comportamiento de los datos.
- 3) Calcule el Ancho del Intervalo (i). - Se obtiene dividiendo el Rango para el número de intervalos. Cuando el valor de i no es exacto, se debe redondear al valor superior más cercano. Esto altera el valor de rango por lo que es necesario efectuar un ajuste así:4) Calcule las Frecuencias y realiza una gráfica en donde el eje vertical son los intervalos y el eje horizontal es la frecuencia de cada intervalo.

Requerimientos del Sistema

- Procesador core i3 o ryzen 3 como mínimo
- Sistema Operativo Windows 7 en adelante
- Compilador dev c instalado o VSCode

Estructura del código

Nuestro trabajo fue realizado mediante diversas funciones, dedicadas al cálculo de las medidas de tendencia central y de varianza, generador de gráfica de frecuencia, y las funciones para la fácil manipulación del programa para el usuario. A continuación, se explicará con detalle cada función o cabecera.

```

#include <locale.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <conio.h>
#include <Windows.h>

```

```

//DECLARACION DE FUNCIONES
void menu();
void presentacion();
void encabezado();
void lectura(float nums[],int n);
void despliegue (float calif[],int n);
int NumeroDatos(float nums[]);
int MenuPrimeraOpcion(void);
//tendencia central
float media(float nums[],int n);
void ordenar(float nums[], int n);
float mediana(float nums[],int n);
float moda(float nums[],int n);
//variabilidad
float rango(float nums[],int n);
float varianza(float med,float nums[],int n);
float desvar(float var);
float coefvar(float x, float desvar);
//grafica
int numInt(int n);
int anchInt(int inter, float rang);
void cls(void);
void grafica(float nums[],int n);
//"Estetica" del programa
float timer;
void gotoxy(int x, int y) ;
int selector(int x,int y,int n);
void bordes(int opc);
void gatos(void);

```

Fuera del main se tiene declarado un puntero a un archivo llamado Archivo y este servirá para la el ingreso y manipulación de datos, al igual que una variable global llamada timer, que ayudara a la impresión de elementos.

Función main o principal

Es una función tipo int, donde se invoca las funciones setlocale(LC_ALL, "Spanish"), encabezado() y menu(), siendo la primera encargada de poder utilizar el teclado en lenguaje español, y las otras dos dan inicio a la manipulación del usuario dentro del programa.

A screenshot of a code editor with a dark background and a blue border. The code is written in C and shows the main function. It includes a declaration for a FILE pointer, a comment for the main function, and the function body which calls setlocale, encabezado, menu, and returns 0.

```
FILE *Archivo;

//FUNCION PRINCIPAL
int main(){
    setlocale(LC_ALL, "Spanish");
    encabezado();
    menu();
    return 0;
}
```

void gotoxy(int x, int y)

Función encargada de la ubicación de elementos mediante coordenadas, por esto mismo se ocupa de parámetros las variables x y.

int selector(int x,int y,int n)

Función encargada de la selección de opciones dentro de la función menu. Dentro tiene declaradas las variables char tecla='\0' y int opc=1, se invoca la funcion para darle el color blanco, innovamos función gotoxy, e imprimimos un triángulo con el código ASCII, siendo el numero 16. Después entramos a un ciclo anidado de do-while con un condicional if para ir cambiando de posición el triangulo mediante las teclas de las flechas abajo o arriba. Si la tecla es la flecha hacia arriba, se le resta una coordenada en y al igual que un valor en la variable opc, en caso contrario que sea la flecha hacia abajo, se le suma coordenada y valor a opc. Para salir del ciclo do-while, se tiene como condicion de que no acabe hasta que la tecla enter sea presionada, siendo en código ASCII el numero 13.

```
int selector(int x,int y,int n){
    char tecla='\0'; //variables
    int opc;
    opc = 1;
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15); //colores
    gotoxy(x,y);
    printf("%c",16); //triangulito
    do{
        if(kbhit()) {
            tecla=getch();

            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),0);
            gotoxy(x,y);
            printf("%c",16);
            if(opc>1&&tecla==72){ //si opcion es mayor que uno y la tecla es hacia arriba
                y--; //se resta coordenada
                opc--; //se resta opcion
            }
            if(opc<8&&tecla==80){ //si opcion es menor que el tope y la tecla es hacia abajo
                y++; //se suma coordenada
                opc++; //se suma opcion
            }
            SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
            gotoxy(x,y);
            printf("%c",16); //bara que parpadea
        }
    }while(tecla!=13); //mientras la tecla no sea enter
    return(opc);
}
```

void bordes(int opc)

Esta función es encargada de los márgenes de cada opción que pueda seleccionar el usuario, por esto mismo se usó un switch, declarando las variables x, y, i, j para las coordenadas, timer e invocando la función para darle color rojo con el numero 12. Igualmente invocamos dos funciones, gotoxy y sleep. Con la primera funcion invocada, se marca la coordenada de cada carácter, ya sea "|" o "-" y esta coordenada va cambiando de acuerdo varios ciclos for. La funcion sleep es la encargada de imprimir con un destiempo estos caracteres, dándole el tiempo con la variable timer, siendo su valor 1.

void gatos(void)

Esta función es por estética de la portada, siendo este un ASCII art que encontramos en la pagina <https://www.asciiart.eu/animals/cats>, esta está ubicada por la función gotoxy y la de color.

void cls(void)

Es una función encargada de limpiar la terminal de salida, perfectamente se puede usar un system("clear"), pero uno de los integrantes del equipo trabajo el proyecto base desde el sistema Linux, siendo esta la razón por la que se dejó esta función.

void encabezado()

Es una función encargada de mostrar una pantalla de inicio, dando como salida la impresión del nombre del proyecto y los integrantes del proyecto, al igual que de entrada tiene la opción para continuar con el programa. Primero llama a la función bordes (comentada en los siguientes párrafos), para poder aplicar un marco a los datos, después se llama a la función

SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15) para darle color blanco y a la función gotoxy para ubicar los datos. Después se llama a la función gatos para finalizar con la portada.

```
void encabezado(){
    bordes(1);

    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15)
    ;
    gotoxy(30,4);
    printf("177932 - Cuenca Esquivel Ana Karen");
    gotoxy(27,5);
    printf("177520 - Esquivel Correa Juan Francisco\n");
    gotoxy(24,6);
    printf("MEDIDAS DE TENDENCIA CENTRAL, VARIANZA Y
    GRAFICA");
    gatos();
    gotoxy(32,8);

    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15
    );
    printf("PRESIONE ENTER PARA CONTINUAR ");
}
```

void presentación ()

Función encargada de seleccionar la opción para el menu. Se invoca la función getch y cls para limpiar la terminal al darle enter, al igual que se manda a llamar la función bordes en la opción 2, se cambia el color del texto a blanco y con la función gotoxy se ubican los elementos.

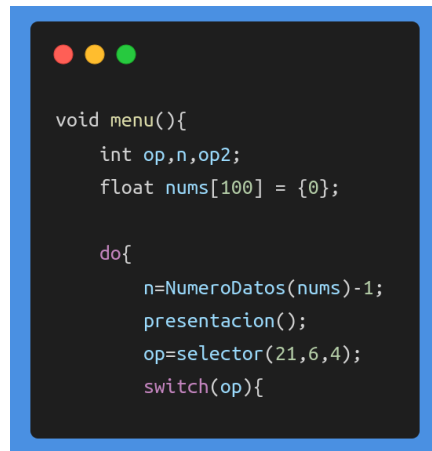

```

void presentacion(){
    getch();
    cls();
    bordes(2);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
    gotoxy(30,4);
    printf("Selecciona una opcion: ");
    gotoxy(24,6);
    printf("1.-Consultar datos o modificacion de datos");
    gotoxy(24,7);
    printf("2.-Desplegar estadisticas");
    gotoxy(24,8);
    printf("3.-Ver grafica");
    gotoxy(24,9);
    printf("4-.SALIR\n");
}

```

void menu()

El objetivo de esta función es mandar a llamar las funciones que se ocupan de acuerdo a la opción seleccionada por el usuario, por esto mismo se ocupa un ciclo do-while y un switch dentro del ciclo, por esto mismo, es la función más importante dentro del programa. Aquí están declaradas las variables int op, n, op2 y el float nums[100] = {0}, siendo n y nums lo más importante, porque serán los parámetros que se mandarán a las demás funciones. Dentro del do-while y fuera del switch, se tiene igualado la variable n con la función NumeroDatos con parámetro el arreglo nums disminuido en 1, esto para saber cuántos números se van a manipular y despreciando el índice 0; después se manda a llamar la función presentación, explicamos anteriormente, y la variable op se iguala a la función selector con valores que ocupara la función para ubicar el triángulo.



```
void menu(){
    int op,n,op2;
    float nums[100] = {0};

    do{
        n=NumeroDatos(nums)-1;
        presentacion();
        op=selector(21,6,4);
        switch(op){
```

Pasamos al switch con su respectivo caso:

En caso de seleccionar la primera opción, se borrará la presentación, se mandará a llamar la función `borde` con la opción 3, se imprimirá los subtítulos con `gotoxy` y se tendrán otras dos opciones para entrar a un submenu, por lo que la variable `op2` se igualara a la función `selector` con nuevos parámetros para la ubicación del triángulo. En el primer caso, se limpia la terminal y se podrá ver los datos y ordenarlos, esto con la función `lectura(nums,n)` y `ordenar(nums,n)`, se explicarán posteriormente. En el segundo caso se podrá manipular los datos del archivo `txt`, invocando a la función `system("notepad Datos.txt")`, al guardar los datos modificados saldrá una leyenda diciendo que están actualizados y se regresara el valor de `op` a 1. NOTA: en los datos modificados por el usuario, no se podrá dejar líneas vacías dentro del `txt`, puesto que marcará ciertos errores.

```
case 1:

    cls();
    bordes(3);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
    gotoxy(35,4);
    printf("ELEGISTE LA PRIMERA OPCION");
    gotoxy(24,6);
    printf("Deseas ver los datos o modificar los datos andentro del archivo?");
    gotoxy(24,7);
    printf("1.Ver datos");
    gotoxy(24,8);
    printf("2.Editar datos");
    op2=selector(21,7,2);
    switch(op2){
        case 1:
            gotoxy(24,10);
            printf("Escogiste: Ver datos");
            getch();
            cls();
            lectura(nums, n);
            cls();
            ordenar(nums,n);
            break;
        case 2:
            system("notepad Datos.txt");
            gotoxy(24,10);
            printf("Perfecto, datos actualizados :)");
            break;
            op=1;
    }
    break;
```

En la segunda opción esta encargada del despliegue de las estadísticas, puesto que se llaman a las funciones encargadas de cada una de ellas; la impresión de cada función se ubica mediante la famosa función gotoxy, dándole color blanco y llamando a la función bordes en el caso 4 (para las medidas de tendencia central) y en el caso 5 (medidas de varianza).

```

case 2: //desplegar estadísticas
    cls();
    ordenar(nums,n);
    getch();
    cls();
    bordes(4);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
    gotoxy(20,4);
    printf("ESTADISTICAS");
    gotoxy(20,6);
    printf("La media es: %.2f",media(nums,n));
    gotoxy(20,8);
    printf("La mediana es: %.2f",mediana(nums,n));
    gotoxy(20,10);
    printf("La moda es: %.2f",moda(nums,n));
    getch();
    cls();
    bordes(5);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
    gotoxy(20,4);
    printf("MEDIDAS DE VARIANZA");
    gotoxy(20,6);
    printf("El rango es: %.2f",rango(nums,n));
    float med=media(nums,n);
    gotoxy(20,8);
    printf("La varianza es: %.2f",varianza(med,nums,n));
    float var=varianza(med,nums,n);
    gotoxy(20,10);
    printf("La desviación estandar %.2f",desvar(var));
    float desvari=desvar(var);
    gotoxy(20,12);
    printf("El coeficiente de varianza: %.2f",coefvar(med,desvari));
    break;

```

En la opción tres, podremos observar la gráfica, realizada por la función gráfica, explicada a continuación.

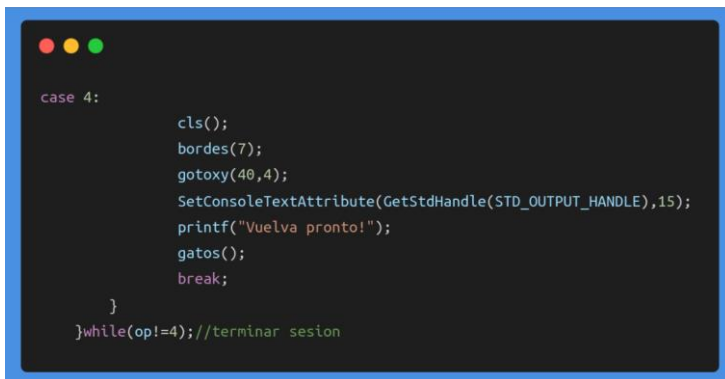
```

case 3:

    cls();
    gotoxy(30,4);
    printf("ELEGISTE LA TERCERA OPCION");
    grafica(nums,n);
    break;

```

En la última opción se hará lo mismo de limpiar y ubicar con color la despedida del programa, al igual que se vuelve a llamar la función `gatos`. En el `do-while` tenemos que se finalizara el ciclo hasta que `op` sea igual a cero, puesto que sería el fin de la interacción con el usuario.



```
case 4:
    cls();
    bordes(7);
    gotoxy(40,4);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
    printf("Vuelva pronto!");
    gatos();
    break;
}
}while(op!=4); //terminar sesion
```

`int NumeroDatos(float nums[])`

Esta función es importante para saber cuántos datos hay en el fichero, hace la lectura del documento. Usa las variables locales `i` y `n`. abre el archivo con la función `fopen("Datos.txt", "r")`. Después confirma si el archivo esta creado, mediante un `if`, siendo que si `Archivo` es igual a `NULL`, quiere decir que no esta creado. Después tenemos un ciclo `while`, para que mientras no sea el final del archivo, se escanee cada línea del `txt` y cuenta cuantos números hay en el documento. Termina cerrando el archivo y regresa el valor de la variable `n`.

```

int NumeroDatos(float nums[]){
    int i,n;
    Archivo = fopen("Datos.txt", "r");
    if(Archivo == NULL){
        printf("\nFichero no existe! \nPor favor creelo");
    }

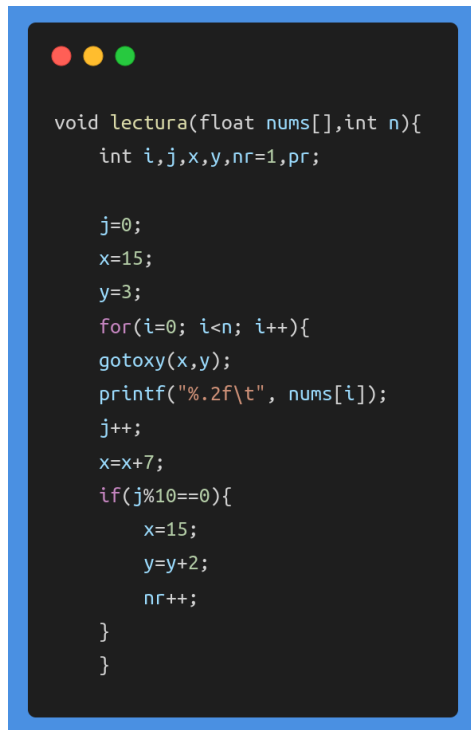
    n=0;
    i=0;
    while (feof(Archivo) == 0){
        fscanf(Archivo,"%f", &nums[i]);
        n++;
        i++;
    }

    fclose(Archivo);
    return (n);
}

```

void lectura(float nums[],int n)

Se declara las variables int i,j=0, x=15, y=3, nr=1, pr. Entramos a un ciclo for para que hasta el número de datos que hay en documento (calculado previamente en la función NumeroDatos) se vayan imprimiendo los datos, al igual que se va sumando una unidad en j y 7 unidades en la coordenada x, después entramos a un condicional, si la división de j entre 10, da cero, ubica x con valor en 15, y se agregan 2 unidades, y en nr se agrega por unidad.



```
void lectura(float nums[],int n){
    int i,j,x,y,nr=1,pr;

    j=0;
    x=15;
    y=3;
    for(i=0; i<n; i++){
        gotoxy(x,y);
        printf("%.2f\t", nums[i]);
        j++;
        x=x+7;
        if(j%10==0){
            x=15;
            y=y+2;
            nr++;
        }
    }
}
```

Estos últimos cálculos del if, sirven para el despliegue de los bordes, están dentro de esta función puesto que dependen de nr para adaptar las dimensiones de la tabla.

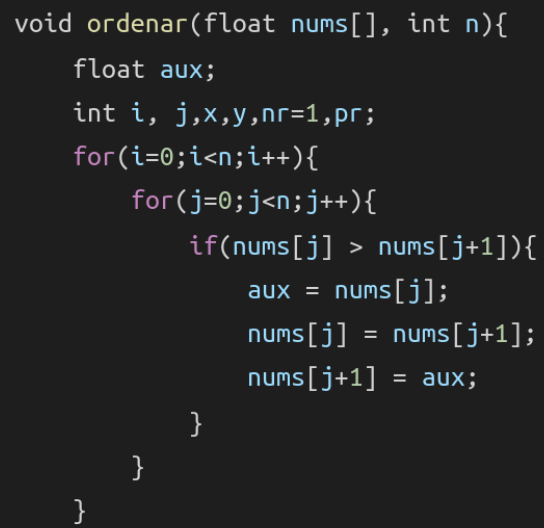
```

gotoxy(43,1);
printf("CALIFICACIONES: ");
SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),12);
y=2;
j=4;
for(pr=nr;pr>0;pr=pr-1){
    i=92;
    for(x=10;x<93;x++){
        gotoxy(x,y);
        printf("-"); Sleep(timer);
        i--;
    }
    j=j+2;
    y=y+2;
}

```

void ordenar(float nums[], int n)

Tenemos las variables float aux; int i, j,x,y,nr=1, pr; para entrar a un ciclo anidado de for, usando el método burbuja para ordenar el vector. Después se realiza lo mismo en los bordes de la función lectura.



```
void ordenar(float nums[], int n){  
    float aux;  
    int i, j,x,y,nr=1,pr;  
    for(i=0;i<n;i++){  
        for(j=0;j<n;j++){  
            if(nums[j] > nums[j+1]){  
                aux = nums[j];  
                nums[j] = nums[j+1];  
                nums[j+1] = aux;  
            }  
        }  
    }  
}
```

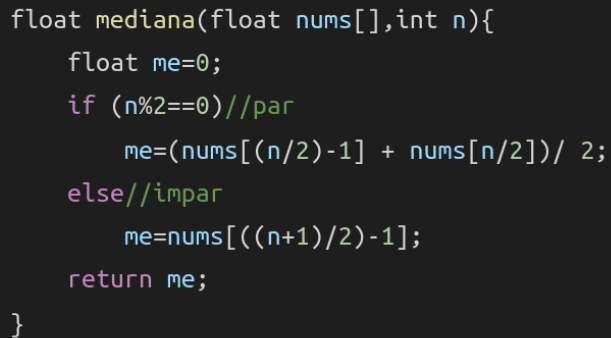
```

gotoxy(37,1);
printf("CALIFICACIONES ORDENADAS: ");
j=0;
x=15;
y=3;
for(i=0; i<n; i++){
gotoxy(x,y);
printf("%.2f\t", nums[i]);
j++;
x=x+7;
if(j%10==0){
    x=15;
    y=y+2;
    nr++;
}
}

```

float mediana(float nums[],int n)

Recibe como parámetros el arreglo de tipo float llamado nums y la variable de tipo entera llamada n. Antes de invocar esta función, se debió invocar la función ordenar, para que así sea más fácil encontrar la media. Dentro de la función, se declara una variable local igualada a cero llamadas me, en donde entra a una condicional, donde la variable n se divide entre dos, si el residuo de la operación es cero, lo que significa que es un numero par, por lo que la variable me será igual al índice del arreglo, dividido entre dos y restado en una unidad, más el mismo arreglo, pero con índice dividido entre dos, todo esto, dividido nuevamente entre dos. En caso de que el residuo de la división no sea cero, significara que es impar, por lo que la variable me, tomara el valor del índice de nums más 1 dividido entre dos y todo esto restado en 1 unidad. Al finalizar con la condición, se regresará el valor de la variable me, la mitad del arreglo.



```
float mediana(float nums[],int n){  
    float me=0;  
    if (n%2==0)//par  
        me=(nums[(n/2)-1] + nums[n/2])/ 2;  
    else//impar  
        me=nums[((n+1)/2)-1];  
    return me;  
}
```

float media(float nums[],int n)

Al igual que con la función mediana, ocupamos el mismo tipo de función y los mismos parámetros. Dentro de esta función, se declararon dos variables, x de tipo float e i de tipo entero. Al calcular la media, se ocupó sumar todos los números que hay dentro del arreglo, por lo que se prefirió usar un ciclo for, usando como contador la variable i, y agregando este valor en la variable x, al finalizar el ciclo for (que va de 0 hasta n), se divide x entre n, dando de regreso el valor de la variable x, siendo el valor de la media.

```

float media(float nums[],int n){
    float x=0;
    int i;
    for(i=0;i<n;i++){
        x+=nums[i];
    }
    x/=n;
    return x;
}

```

float moda(float nums[],int n)

La principal función de esta función es realizar un conteo de los valores que se repiten. Se sigue usando el mismo tipo de función y parámetros, aunque dentro de esta se ocupan más variables locales, siendo may y mod de tipo float, e i y j de tipo entero. Se uso un ciclo anidado de for con condicionales, el primero for se realiza de i=0 hasta la variable n, se declara la función cont=0 y empieza otro for, empezando desde i=0 hasta j<n, si nums[i] es igual a nums[j], significa que es el mismo número y le suma una unidad al contador. Fuera de este segundo for, se tiene la condicional sobre si el contador es mayor a la variable may, puesto que irá cambiando los valores hasta que se encuentre el contador con mayor cantidad de repetidos, entonces ese valor estará en el índice del arreglo nums. Al finalizar el ciclo anidado, se regresará el valor de mod, la moda.

Commented [CK1]: incompleta,no me acuerdo como la hice

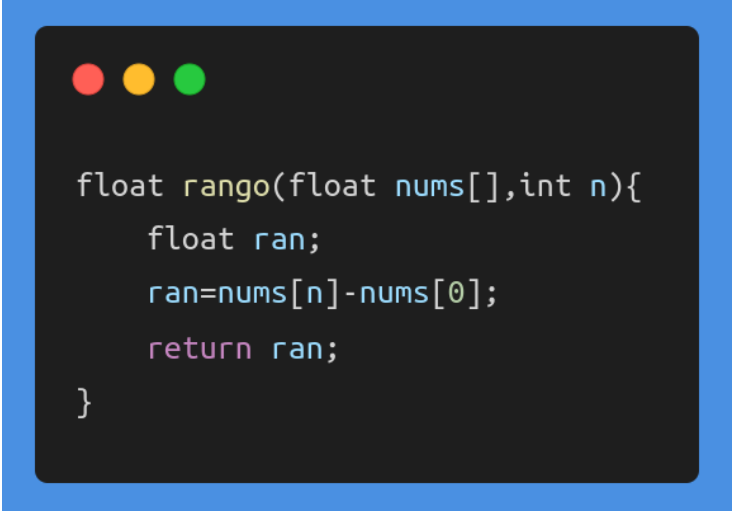
```

float moda(float nums[],int n){
    float may=0;
    float mod=0;
    int i,j;
    for (i=0; i<(n); i++){
        int cont=0;
        for (j=i+1; j<n; j++){
            if (nums[i]==nums[j])
                cont++;
        }
        if (cont>may)
        {
            may=cont;
            mod=nums[i];
        }
    }
    return mod;
}

```

float rango(float nums[],int n)

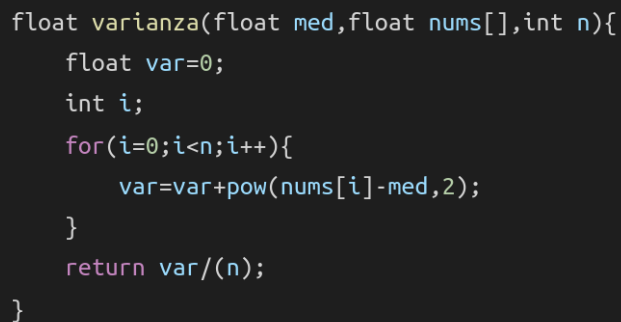
Se declara la variable ran, después se iguala al valor mayor del arreglo, nums[n](está en la última posición del arreglo después de ordenarlo, por eso es n) restado a la primera posición de nums (nums[0]). Regresas el valor de ran, el rango.



```
float rango(float nums[],int n){
    float ran;
    ran=nums[n]-nums[0];
    return ran;
}
```

float varianza(float med,float nums[],int n)

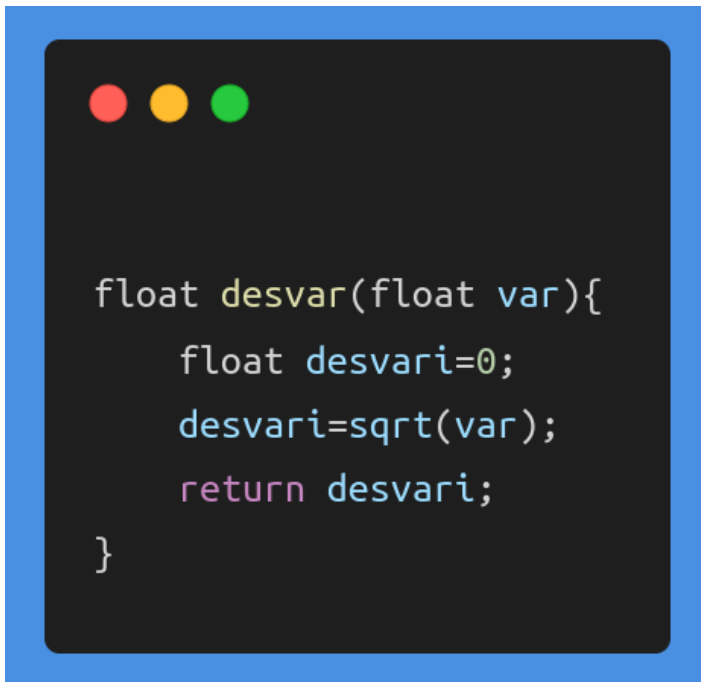
En esta función se le agrega otro parámetro, que sería la variable med, donde se guardó el valor de la media. Se declara float var=0 e int i, para usarlos en un ciclo for, que ira de i=0 hasta i<n, dentro de este ciclo la variable var será igual a esta misma variable más el cuadrado de nums[i] menos la media. Regresa el valor de var dividido entre n.



```
float varianza(float med,float nums[],int n){
    float var=0;
    int i;
    for(i=0;i<n;i++){
        var=var+pow(nums[i]-med,2);
    }
    return var/(n);
}
```

float desvar(float var)

Recibe el valor de la variable var, después se saca la raíz de var y esto será igual a la variable desvari, siendo esto la desviación estándar.



```
float desvar(float var){  
    float desvari=0;  
    desvari=sqrt(var);  
    return desvari;  
}
```

float coefvar(float med, float desvari)

Usaos los parámetros de med y desvari, puesto que esos datos calculados previamente, son los que se ocupa para calcular el coeficiente de varianza, dividiendo desvari/med. Regresamos el valor de esto en la variable cv.

```
float coefvar(float med, float desvari){
    float cv=0;
    cv=desvari/med;
    return cv;
}
```

int numInt(int n)

Calcula número de intervalos de acuerdo a la formula proporcionada, siendo $1+3.32*\log(n)$;

int anchInt(int inter, float rang)

Función que imprime ahí mismo el valor del ancho de los intervalos, dividiendo el rango entre el valor de los intervalos.

```
int numInt(int n){
    return 1+3.32*log(n);
}
int anchInt(int inter, float rang){
    return printf("El ancho de los intervalos es: %f",rang/inter);
}
```

void grafica(float nums[],int n)

Se declaran x,y, i, j z,u,r para la ubicación de las coordenadas, al igual que la variable global timer=80. Invocamos la función bordes con la opción 6. Damos color blanco con el SetConsoleTextAttribute. Hacemos unos pequeños cálculos, siendo el rangografica=rango(nums,n), para guardar el valor calculado en la función rango, al igual hacemos con la función intervalos, guardada en la variable numeinte, y el ancho es igual al rangografica/numeinter. Después de esto ubicamos los valores de las coordenadas y el limite inicial. Usamos un ciclo anidado con for, siendo el primero de 0 hasta el numerinte, dentro de este primer for, igualamos la variable segundo a primero + ancho, esto para calcular el limite superior, llamamos a la función gotoxy e imprimimos los límites. Entramos al segundo for que va de 0 hasta n, dentro esta una condicional, siendo que si la posición n dentro del arreglo es mayor o igual al limite

inferior y menor al límite superior, imprime, así podemos saber la frecuencia entre los límites. También se tienen unos ciclos para el borde de la gráfica

```
void grafica(float nums[],int n){
    int x,y,i,j,z,u,r;
    timer=80;
    float rangografica;
    float numerint,ancho,primero,segundo;

    rangografica=rango(nums,n);
    numerint=intervalos(n);
    ancho=rangografica/numerint;
    gotoxy(25,4);
    printf("Rango: %.2f",rangografica);
    gotoxy(25,5);
    printf("Ancho de intervalos: %.2f", ancho);
    gotoxy(25,6);
    printf("Numero de intervalos: %.2f", numerint);

    x=15;
    y=10;
```

```

r=10;
prInero=nums[0];

for (l=0; l<numerInt; l++){
    segundo=prInero+ancho;
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),12);
    for (z=12; z<74; z++){
        gotoxy(z,y-1);
        printf("-");
        gotoxy(z,y+1);
        printf("-");
    }

    u=r+2;
    for (r; r<u; r++){
        gotoxy(12,r);
        printf("|");
        gotoxy(35,r);
        printf("|");
        gotoxy(73,r);
        printf("|");
    }
    gotoxy(x,y);
    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),15);
    printf("%.2f" ,prInero); Sleep(timer); printf(" --"); Sleep(timer); printf(" %.2f",segundo); Sleep(timer);
    gotoxy(40,y);
    for (j=0; j<n; j++){
        if (nums[j]==prInero && nums[j]<=segundo){
            printf("%c",4); Sleep(timer);
        }
    }
    prInero=segundo;
    y=y+2;
}

}

```