|  |  |  |
| --- | --- | --- |
| Resultado de imagen para ipn escudo | **Instituto Politécnico Nacional** | Resultado de imagen para escom escudo |
| Escuela Superior de Cómputo | | |
| *Práctica 0: Manejo de vectores, implementando su uso para la manipulación de un Conjunto de Muestras de Aprendizaje* | | |
| Becerril Saldivar Luis Alejandro | | |
| 2CM10 | | |
| Pattern Recognition | | |
| Docente: M. en C. Ma. Elena Cruz Meza | | |

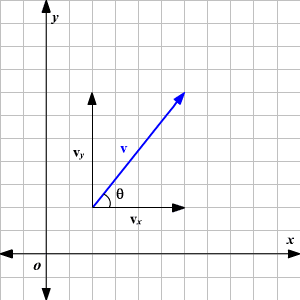
**Marco teórico:**

**Componentes de un vector**

En un sistema coordenado de dos dimensiones, cualquier vector puede separarse en el componente x y el componente y.

https://www.varsitytutors.com/assets/vt-hotmath-legacy/hotmath_help/spanish/topics/components-of-a-vector/image001.gif

Por ejemplo, en la figura siguiente mostrada, el vector se separa en dos componentes, y .



A través de estos vectores se pueden realizar operaciones, como lo son: suma, resta y multiplicación por un número escalar.

**Suma de vectores**

Conociendo las componentes cartesianas de los vectores a sumar, el vector resultante tendrá como componentes cartesianos la suma, eje a eje, de cada vector.

Siendo y el resultado al sumar las componentes quedaría:

**Resta de vectores**

Conociendo las componentes cartesianas de los vectores a sumar, el vector resultante tendrá como componentes cartesianos la resta, eje a eje, de cada vector.

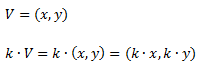
Siendo y el resultado al sumar las componentes quedaría:

**Producto de un escalar por un vector**

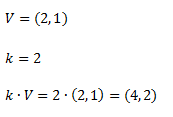
El producto de un escalar por un vector da por resultado otro vector, con la misma dirección que el primero. Al hacer la multiplicación, el escalar cambia el módulo del vector (gráficamente el largo) y en caso de ser negativo cambia también el sentido. La dirección del vector resultado es siempre la misma que la del vector original.

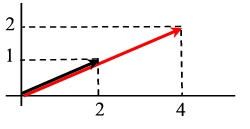
Matemáticamente se realiza multiplicando al escalar por cada una de las componentes del vector.

Si por ejemplo el vector V tiene 2 coordenadas:



Ejemplo:





Código fuente del programa:

/\*

Becerril Saldivar Luis Alejandro

Practica 0

Reconocimiento de patrones

\*/

/\*

Este programa utiliza archivos de texto

a continuación se muestra un ejemplo del contenido de un archivo

4 2

5 6

7 8

El archivo de texto solo contendran Caracteres que en su valor ANSCII corresponde a números

La primera columna corresponde a los elementos de la coordenada x y la segunda columna corresponde a los elementos de coordenada y

\*/

// Librerías correspondientes para la implementación de

//funciones dentro del programa

#include <stdio.h>

#include <string.h>

#include <dirent.h>

#include <windows.h>

#include <conio2.h>

#include <time.h>

#include <stdlib.h>

//Funciones Personalizadas

/\*

Función: CrearClases

Recibe un parámetro entero, dicha función te permite crear la clase

y se almacena en un archivo de texto plano txt

\*/

void CrearClases(int Numclases);

/\*

Función: Listar

Muestra una lista de clases creadas en el directorio donde este corriendo el

programa

\*/

void Listar();

/\*Función: Menu

Desglosa las opciones con las cuales puede interactuar el usuario

retornando un valor entero

\*/

int Menu();

/\*Función: NumClases

Función que retoñar cuentas clases se crean

\*/

int NumClases();

/\*Función: MenuOp

Desglosa el menú de las operaciones que se pueden realizar entre las clases

y retorna un valor entero

\*/

int MenuOp();

/\*Función: Operaciones

Función que realiza las 3 operaciones resta, suma y multiplicación.

Solo se aplican a una clase individual

\*/

void Operaciones();

/\*Función: Plano

Dibujo en consola un plano XY

\*/

void Plano();

/\*Función: Setpoint

Dibuja un punto en el plano con sus coordenadas XY.

\*/

void Setpoint(int x,int y);

/\*Función: SetpointArray

Dibuja en el plano más de un vector con sus coordenadas XY.

\*/

void SetpointArray(int vectores[], int indice);

/\*Función: SetpointArray

Función que permite graficar clases independientes

\*/

void Graficar();

/\*Función principal del programa\*/

int main()

{

int opc;

/\*

Menu principal donde interactúa el usuario

\*/

Inicio:

switch (Menu())

{

case 1: CrearClases(NumClases()); goto Inicio;break;

case 2: Listar(); goto Inicio; break;

case 3: Operaciones();goto Inicio;break;

case 4: Graficar();goto Inicio; break;

case 5: goto Salir; break;

default: printf("\n opcion incorrecta");goto Inicio; break;

}

Salir:

return 0;

}

/\*

Función: CrearClases

Recibe un parámetro entero, dicha función te permite crear la clase

y se almacena en un archivo de texto plano txt

\*/

void CrearClases(int Numclases)

{

//Limpiamos buffer para que el apuntador del archivo no presente problemas

fflush( stdin );

char nombre[]="default.txt";

//Ciclo que se repite N veces de acuerdo a las clases que vamos a crear

for(int i=0;i<Numclases;i++)

{

char \*puntero=nombre;

printf("\n Ingresa el nombre de la clase %d --> ",(i+1));

gets(puntero); //Asignamos el nombre al archivo que represetara la clase

strcat(puntero,".txt");//Concatenamos la extension para el archivo

FILE \*fichero; //Punto para el archivo

fichero = fopen(puntero,"a"); // Creamos el fichero en modo de escritura y si no está lo crea

if (fichero == NULL){ //En caso de que no se pueda crear mandamos el

//mensaje de error

printf("\n No se creó la clase");

}else //Caso contrario soliciarmos cuantos patrones tendra la clase

{ int Nump=0;

printf(" Cuantos patrones tendra esta clase? --> ");

scanf("%d",&Nump);

fflush( stdin );

for(int i=0;i<Nump;i++) //dentro del fichero colocamos los patrones

{

int x1=0,x2=0;

printf("\n Ingresa el par numero %d ",(i+1));

printf("\n Ingresa x1 ->");scanf("%d",&x1);

fflush( stdin );

printf(" Ingresa x2 ->");scanf("%d",&x2);

fflush( stdin );

if(i==0){fprintf(fichero,"%d %d",x1,x2);}

else{fprintf(fichero,"\n%d %d",x1,x2);}

}

}

fclose(fichero); //Cerramos el fichero

printf(" \n\n\t##### Clase(s) creada(s) con sus patrones con exito #####\n\n\n");

Sleep(500); //Confirmamos que el fichero se creó correctamente

}

}

void Listar()

{

DIR \*d; // Se crea un punto a una estructura de tipo directorio para listar los archivos

struct dirent \*dir;

d = opendir(".");

if (d) // si se localiza directorio listamos los mismos

{

printf(" \n\n\t\t ###### Lista de clases creadas ###### \n\n ");

while ((dir = readdir(d)) != NULL)

{

//string[position+c-1];

if(strlen(dir->d\_name)>4)

{

char aux[strlen(dir->d\_name)];

strcpy(aux,dir->d\_name);

for(int i=0;i<strlen(dir->d\_name);i++)

{

if(aux[i]=='.')

{

if( aux[i+1]=='t' && aux[i+2]=='x' && aux[i+3]=='t') //Aplicamos un filtro y solo

//mostrara archivos txt

{

for(int a=0;a<strlen(dir->d\_name);a++)

{

if(aux[a]!='.'){printf("%c",aux[a]);}

else {printf("\n ");break;};

}

}

}

}

//printf("%s\n", dir->d\_name);

}

}

closedir(d);

}

}

/\*

Función men que no brinda las opciones en las que puede interacturar el usuario

\*/

int Menu ()

{

int opc;

textcolor(15);

printf("\n\t\t\t\t #### Menu ####\n\n");

printf("\n\t\t\t 1- Crear clase(s)");

printf("\n\t\t\t 2- Ver clase(s) creadas");

printf("\n\t\t\t 3- Operaciones con clases");

printf("\n\t\t\t 4- Graficar clase(s)");

printf("\n\t\t\t 5- Salir");

printf("\n\t\t\t --> ");scanf("%d",&opc);

return opc;

}

/\*

Función NumClases, solo los regresa un valor entero que son las clases que se crearan

\*/

int NumClases()

{

int opc;

printf("\n\t\t Ingresa cuentas clases deseas crear -> ");

scanf("%d",&opc);

return opc;

}

void Operaciones () // Operaciones que puede realizar el usuario

{

char nombre[]="default.txt";

char \*puntero=nombre; //Creamos un puntero para almacenar el nombre de la clase

//Variables auxiliares para las operaciones

int x1=0,x2=0;

int aux=0;

int a=0;

//Puntero para abrir el archivo

FILE \*archivo;

char caracter;

char auxcaracter;

int vectores[30];

int escalar=0,numv=0;

//Se lista las clases creadas

Listar();

TryAgain: //Declaramos el ounto donde regresara la función goto

switch (MenuOp())

{

case 1: //Cuando ingresa a la opción uno se suman los valores de una misma clase

fflush( stdin );

printf("\n Ingresa el nombre de la clase a operar--> "); //Ingresamos en nombre de la clase a operar

gets(puntero);

strcat(puntero,".txt"); //Concatenamos el nombre con al extenci󮍊

/////// Suma

archivo = fopen(puntero,"r"); //Abrimos el archivo solo en modo lectura

if (archivo == NULL)

{

printf("\n Error de apertura de la clase. \n\n");

}

else //Si el archivo no tiene problemas en abrir lo recorremos hasta encontrar el final del archivo EOF

{

//Ingreso a la clase con exito

while((caracter = fgetc(archivo)) != EOF)

{

//printf("\n--> %c <--",caracter);

if(caracter!=' ' && '\n' != caracter && aux==0) //Aplicamos un filtro para tomar solo los cárteres que corresponde a los números

{

x1=x1+(caracter - '0'); //Aplicamos la conversión ara pasar del valor ANSCII al valor decimal y lo almacenamos en la bar aux x1

aux=aux+1; // Este variable detecta si es coordenada x o y

}

//Se aplica el mismo proceso de la parte de arriba

else if(caracter!=' ' && '\n' != caracter && aux==1)

{

x2=x2+(caracter - '0');

aux=aux-1;

}

}

} //Cerramos el archivo y mostramos el resultado y damos la opción al usuario de graficar

fclose(archivo);

printf(" \n\t\t Resultado X1:%d ; x2:%d ",x1,x2);

fflush( stdin );

printf(" \n\t\t Graficar el punto? Si-1 No-0 -->");int z;scanf("%d",&z);

if(z==1){Sleep(500);system("cls");Plano();Setpoint(x1,x2);getch();}

////// ##### Fin archivo

break;

case 2:

fflush( stdin ); //Seaplica el mismo proceso que de la suma, exeptuando que ahora se restan los vectores

printf("\n Ingresa el nombre de la clase a operar--> ");

gets(puntero);

strcat(puntero,".txt");

/////// #### Archvio resta

archivo = fopen(puntero,"r");

if (archivo == NULL)

{

printf("\n Error de apertura de la clase. \n\n");

}

else

{

//Ingreso a la clase con exito

while((caracter = fgetc(archivo)) != EOF)

{

//printf("\n--> %c <--",caracter);

if(caracter!=' ' && '\n' != caracter && aux==0)

{

x1=x1-(caracter - '0');

if(a==0){x1=x1\*-1;a++;}

aux=aux+1;

//printf("\n valor x1: %d",(caracter - '0'));

}

else if(caracter!=' ' && '\n' != caracter && aux==1)

{

x2=x2-(caracter - '0');

if(a==1){x2=x2\*-1;a++;}

aux=aux-1;

//printf("\n valor x2: %d",(caracter - '0'));

}

}

}

fclose(archivo);

printf(" \n\t\t Resultado X1:%d ; X2:%d ",x1,x2);

fflush( stdin );

printf(" \n\t\t Graficar el punto? Si-1 No-0 -->");int zz;scanf("%d",&zz);

if(zz==1){Sleep(500);system("cls");Plano();Setpoint(x1,x2);getch();}

////// ##### Fin archivo

break;

case 3:

//Sea aplica el mismo proceso que la suma y resta para obtener los datos

printf("\n Ingresa un escalar ");scanf("%d",&escalar);

fflush( stdin );

printf("\n Ingresa el nombre de la clase a operar--> ");

gets(puntero);

strcat(puntero,".txt");

/////// #### Archvio escalar

archivo = fopen(puntero,"r");

if (archivo == NULL)

{

printf("\n Error de apertura de la clase. \n\n");

}

else

{

//Ingreso a la clase con exito

while((caracter = fgetc(archivo)) != EOF) //Se recorre todo el archivo

{

//esta vez como se multiplica por un escalar se obtiene el valor de las coordenadas

//Se multiplica y se almacena su valor en un arreglo

if(caracter!=' ' && '\n' != caracter && aux==0)

{

x1=(caracter - '0')\*escalar;

aux=aux+1;

//aux == 1 indica que es coordenada x y aux == 0 indica que es coordenada y

vectores[numv]=x1;

numv++;

printf("\n\t\t X1: %d",x1);

}

else if(caracter!=' ' && '\n' != caracter && aux==1)

{

x2=(caracter - '0')\*escalar;

aux=aux-1;

//aux == 1 indica que es coordenada x y aux == 0 indica que es coordenada y

vectores[numv]=x2;

numv++;

printf("\n\t\t X2: %d",x2);

}

}

}

fclose(archivo); //Cerramos el archivo

//Graficamos el resultado

system("cls");

Plano();

SetpointArray(vectores,numv);

getch();

system("cls");

////// ##### Fin archivo

break;

default: printf("\n opcion incorrecta");goto TryAgain; break;

}

}

/\*

Función que retorna un valor entero, dicho valor corresponde a la opción que selecciona el usuario

\*/

int MenuOp()

{

int opc;

textcolor(10);printf("\n\t\t\t\t #### Menu ####\n\n");

printf("\n\t\t\t 1- Suma");

printf("\n\t\t\t 2- Resta");

printf("\n\t\t\t 3- Multiplicacion por escalar");

printf("\n\t\t\t --> ");scanf("%d",&opc);

return opc;

}

/\*

Función que plasma en consola un plano carteciano en dos dimenciones xy

tomando como referencia el centro de la consola

\*/

void Plano()

{

for(int i=0;i<40;i++)

{

gotoxy(60,i);printf("%c\n",197);

}

for(int i=0;i<120;i++){gotoxy(i,20);printf("%c\n",197);}

//gotoxy(80,20);printf("%c",197);

}

/\*

Función que recibe dos valores enteros, dichos valores se plasman en el plano carteciano como coordenadas

\*/

void Setpoint(int x,int y)

{

if(x>=1 && y>=1)//Primer cuadrante

{

textcolor(14);gotoxy((60+x),(20-y));printf("%c",254);

}

if(x<0 && y>=1)//Segundo cuadrante

{

textcolor(14);gotoxy((60+x),(20+y));printf("%c",254);

}

if(x<0 && y<0)//Tecer cuadrante

{

textcolor(14);gotoxy((60+x),(20+y));printf("%c",254);

}

if(x>=1 && y<0)//Cuarto cuadrante

{

textcolor(14);gotoxy((60+x),(20+y));printf("%c",254);

}

}

/\*

Función que recibe como parametros una arreglo de coordenada del tipo xy

La función hace un recorrido del arreglo y plasma las coordenadas en el plano que se dibuja en consola

\*/

void SetpointArray(int vectores[], int indice)

{

int x=0,y=0;

for(int i=0;i<indice;i=i+2)

{

x=vectores[i];

y=vectores[i+1];

if(x>=1 && y>=1)//Primer cuadrante

{

textcolor(14);gotoxy((60+x),(20-y));printf("%c",254);

}

if(x<0 && y>=1)//Segundo cuadrante

{

textcolor(14);gotoxy((60+x),(20+y));printf("%c",254);

}

if(x<0 && y<0)//Tecer cuadrante

{

textcolor(14);gotoxy((60+x),(20+y));printf("%c",254);

}

if(x>=1 && y<0)//Cuarto cuadrante

{

textcolor(14);gotoxy((60+x),(20+y));printf("%c",254);

}

if(x==0 && y==0)//Centro

{

textcolor(14);gotoxy((60),(20));printf("%c",254);

}

}

}

/\*

Función que permite graficar todos los puntos que esten dentro de una clase

\*/

void Graficar()

{

//Se declara una varibale del tipo char para abrir el carchivo donde estan los datos

char nombre[]="default.txt";

//Puntero dirijido utulizar el nombre y poder abrir el archivo de texto

char \*puntero=nombre;

//Variables auxliares para graficar

int x1=0,x2=0;

int aux=0;

int a=0;

//Se declara un puntero de tipo FILE para abrir el archivo

FILE \*archivo;

//Variables donde se almacenara los valores temporales al abrir el archivo

char caracter;

char auxcaracter;

int vectores[30];

int escalar=1,numv=0;

//Función que lista las clases ya creadas

Listar();

fflush(stdin); //Limpiamos el buffer para no tener problemas en los datos de entrada

printf("\n Ingresa el nombre de la clase a operar--> ");

gets(puntero);

strcat(puntero,".txt");

//A la variable del nombre le concatenamos la extensi󮠤el archivo de txto plano

//Abrimos el archivo en modo lectura "read"

archivo = fopen(puntero,"r");

if (archivo == NULL) //En caso de no poder abrir el archivo se regresa un valor NULL y nos avisa que la clase no se encontro

{

printf("\n Error de apertura de la clase. \n\n");

}

else

{

//Ingreso a la clase con exito

//Al ingresar a la clase recorremos todo el archivo hasta encontrar en final EOF "End of file"

while((caracter = fgetc(archivo)) != EOF)

{

//printf("\n--> %c <--",caracter);

if(caracter!=' ' && '\n' != caracter && aux==0) //Aplicamos un filtro para solo tomar los caracteres "numeros"

{

x1=(caracter - '0')\*escalar; //Derivado a que se obtienen caracteres del archivo se aplica una converci󮠥n ASCCI

//para tener el valor del nmero correspondiente

aux=aux+1; //Ocupamos la variable aux para detectar si la coordenada es x o y

// aux = 0 indica que es coordenada x

// aux = 1 indica que es coordenada y

vectores[numv]=x1;

//Almacenamos el valor en el arreglo

numv++; //Incrementamos el valor de indice de arreglo

}

//Se realiza el mismo proceso que en la parte de arriba

else if(caracter!=' ' && '\n' != caracter && aux==1)

{

x2=(caracter - '0')\*escalar;

aux=aux-1;

vectores[numv]=x2;

numv++;

}

}

}

fclose(archivo); //Cerramos el archivo

system("cls");

Plano(); //Llamamos la función para pintar el plano en consola

SetpointArray(vectores,numv); //Colocamos los puntos en el plano

getch();

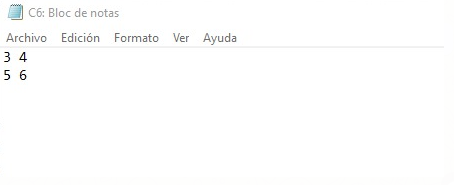
system("cls");

}

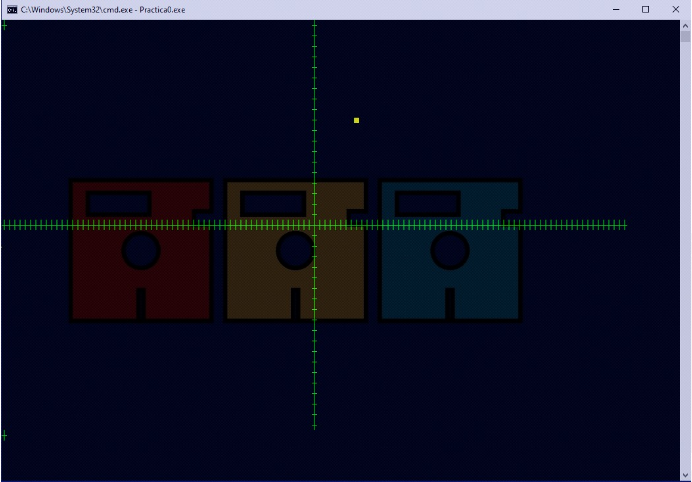
**Capturas de pantalla de la ejecución del programa**











**Referencia bibliográficas:**

Anónimo. (2014). Componentes de un vector. 06/10/2019, de Sitio web: <https://www.varsitytutors.com/hotmath/hotmath_help/spanish/topics/components-of-a-vector>

Anónimo. (2015). OPERACIONES CON VECTORES. 06/10/2019, de Sitio web: <https://www.universoformulas.com/fisica/vectores/operaciones-vectores/>

F. Noguez. (2015). Producto de un escalar por un vector. 06/10/2019, de Sitio web: <https://www.fisicapractica.com/producto-escalar-vector.php>