/\*\*Este Procedimiento Aborda el Tema de Estructuras o STRUCTS\*\*/

#include <stdio.h>

#include <stdlib.h>

int main()

{

    /\* defineInitializeStructs();

    inputAndPrintDate();

    nextDate();

    initializeStruct();

    readAndPrintCoordinate();

    staticArrayOfStruct();

    structWithArrayAsElement();

    usingLogicalOperatorsWithStructs();

    menuToOperateUponFractions();\*/

    usingStructWithinStruct();

    return 0;

}

/\*\*Este Procedimiento Define e Inicializa una Tipo de Dato Struct\*\*/

#include<stdio.h>

struct date{

    int iDay;

    int iMonth;

    int iYear;

};

void defineInitializeStructs()

{

    struct date date1;

    printf("Ingresa los datos de la fecha. \n");

    printf("\tIngresa el dia: ");

    scanf("%i", &date1.iDay);

    fflush(stdin);

    printf("\tIngresa el mes: ");

    scanf("%i", &date1.iMonth);

    fflush(stdin);

    printf("\tIngresa el a�o: ");

    scanf("%i", &date1.iYear);

    fflush(stdin);

    printf("La fecha ingresada es %02i/%02i/%02i. ", date1.iDay, date1.iMonth, date1.iYear);

    printf("\n \n");

}

/\*\*Este Procedimiento Ense�a las Distintas Maneras de Inicializar un Estructura\*\*/

typedef struct coordinate{

    int x;

    int y;

}point;

void initializeStruct()

{

    ///Declarar una estructura sin inicializar

    point p1;

    printf("Punto sin inicializar.\n\tX:%i\n\tY:%i\n", p1.x, p1.y);

    ///Inicializamos una estructura usando valores en orden

    point p2 = { 1, 2 };

    printf("Punto inicializado con valor en orden.\n\tX:%i\n\tY:%i\n", p2.x, p2.y);

    ///Inicializamos una estructura usando los campos correspondiente

    point p3 = { .x = 3, .y = 11 };

    printf("Punto inicializado usando sus campos correspondientes.\n\tX:%i\n\tY:%i\n", p3.x, p3.y);

    ///Inicializamos una estructura usando los campos correspondientes (sin orden)

    point p4 = { .y = 4, .x = 13};

    printf("Punto inicializado usando sus campos correpondientes (sin orden).\n\tX:%i\n\tY:%i\n", p4.x, p4.y);

    ///Inicializamos una estructura usando solo un campo (el otro campo se inicaliza automaticamente a cero)

    point p5 = { .x = 5 };

    printf("Punto inicializado usando solo uno de sus dos campos.\n\tX:%i\n\tY:%i\n", p5.x, p5.y);

    printf("\n \n");

}

/\*\*Este Procedimiento Muestra el Uso del Comando TYPEDEF\*\*/

#include<stdio.h>

typedef struct myDate{

    int iDay;

    int iMonth;

    int iYear;

} Date;

Date inputDate()

{

    Date dt;

    printf("Ingreso de la informacion de la fecha. \n");

    printf("Ingresa el dia: ");

    scanf("%i", &dt.iDay);

    fflush(stdin);

    printf("Ingresa el mes: ");

    scanf("%i", &dt.iMonth);

    fflush(stdin);

    printf("Ingresa el ano: ");

    scanf("%i", &dt.iYear);

    fflush(stdin);

    return dt;

}

void printDate(Date outputDate)

{

    printf("La fecha ingresada es %02i/%02i/%02i. ", outputDate.iDay, outputDate.iMonth, outputDate.iYear);

    printf("\n \n");

}

void inputAndPrintDate()

{

    Date Date1 = inputDate();

    printDate(Date1);

}

/\*\*Este Procedimiento, Dada una Fecha, Calcula la Siguiente Fecha\*\*/

#include<stdio.h>

typedef struct myDate{

    int iDay;

    int iMonth;

    int iYear;

} date;

void nextDate()

{

    date firstDate;

    char    cContinua[10];

    do {

        printf("Ingresa el dia de la fecha: ");

        scanf("%i", &firstDate.iDay);

        fflush(stdin);

        printf("Ingresa el mes de la fecha: ");

        scanf("%i", &firstDate.iMonth);

        fflush(stdin);

        printf("Ingresa el ano de la fecha: ");

        scanf("%i", &firstDate.iYear);

        fflush(stdin);

        printf("\n%02i-%02i-%02i", firstDate.iDay, firstDate.iMonth, firstDate.iYear);

        printf("\n \n");

        /\*Cubrimos todas las Condiciones por las Cuales una Fecha Puede ser Invalida\*/

        if ( ((firstDate.iMonth == 4 || firstDate.iMonth == 6 || firstDate.iMonth == 9 || firstDate.iMonth == 11) && (firstDate.iDay > 30)) || (!(firstDate.iMonth == 2 || firstDate.iMonth == 4 || firstDate.iMonth == 6 || firstDate.iMonth == 9 || firstDate.iMonth == 11) && (firstDate.iDay > 31)) || (firstDate.iMonth > 12) || (firstDate.iMonth < 1) || (firstDate.iDay < 1) || (((firstDate.iYear % 4 == 0) && (!(firstDate.iYear % 100 == 0) || (firstDate.iYear % 100 == 0 && firstDate.iYear % 400 == 0))) && (firstDate.iMonth == 2) && (firstDate.iDay > 29)) || (!((firstDate.iYear % 4 == 0) && (!(firstDate.iYear % 100 == 0) || (firstDate.iYear % 100 == 0 && firstDate.iYear % 400 == 0))) && (firstDate.iMonth == 2) && (firstDate.iDay > 28)))

            printf("La fecha no es valida. ");

        /\*Calculamos la Fecha que Procede a la Fecha Ingresada\*/

        else

        {

            if (((firstDate.iMonth == 4 || firstDate.iMonth == 6 || firstDate.iMonth == 9 || firstDate.iMonth == 11) && firstDate.iDay < 30) || (!(firstDate.iMonth == 2 ||firstDate.iMonth == 4 || firstDate.iMonth == 6 || firstDate.iMonth == 9 || firstDate.iMonth == 11) && firstDate.iDay < 31) || ((firstDate.iMonth == 2) && firstDate.iDay < 28) || (((firstDate.iYear % 4 == 0) && (!(firstDate.iYear % 100 == 0)|| (firstDate.iYear % 100 == 0 && firstDate.iYear % 400 == 0)) ) && (firstDate.iDay < 29)) || (!((firstDate.iYear % 4 == 0) && (!(firstDate.iYear % 100 == 0)|| (firstDate.iYear % 100 == 0 && firstDate.iYear % 400 == 0)) ) && firstDate.iDay < 28) )

            {

                firstDate.iDay = firstDate.iDay + 1;

            }

            else

            {

                firstDate.iDay = 1;

                if (firstDate.iMonth == 12)

                {

                    firstDate.iMonth = 1;

                    firstDate.iYear = firstDate.iYear + 1;

                }

                else

                    firstDate.iMonth = firstDate.iMonth + 1;

            }

            printf("Siguiente fecha: \n%02i-%02i-%02i", firstDate.iDay, firstDate.iMonth, firstDate.iYear);

        }

        printf("\n \n");

        printf("Deseas Continuar ingresando fechas (si, no)?: ");

        scanf("%s", &cContinua);

    } while (strcmp(cContinua, "si") == 0);

    printf("\n \n");

}

/\*\*Este Procedimiento Lee una Estructura del Usuario y Procede a Imprimir Dicha Estructura\*\*/

#include<stdio.h>

typedef struct coordinate{

    float y;

    float x;

}point;

point readCoordinateFromUser()

{

    point pInput;

    printf("Por favor ingresa la siguiente coordenada. \n");

    printf("\tIngresa X: ");

    scanf("%f", &pInput.x);

    fflush(stdin);

    printf("\tIngresa Y: ");

    scanf("%f", &pInput.y);

    fflush(stdin);

    return pInput;

}

void printCoordinateToUser(point pOutput)

{

    printf ("Coordenada ingresada --> (%0.2f, %0.2f). ", pOutput.x, pOutput.y);

}

void readAndPrintCoordinate()

{

    point p1 = readCoordinateFromUser();

    printCoordinateToUser(p1);

    printf("\n \n");

}

/\*\*Este Procedimiento Ense�a los Vectores Estaticos de Tipo ESTRUCTURA O STRUCT\*\*/

#include<stdio.h>

#define iSIZE 5

typedef struct coordinate{

    int x;

    int y;

}point;

void staticArrayOfStruct()

{

    int f;

    point pArray[iSIZE];

    printf("Ingreso de la Informacion de %i Coordenada(s). \n", iSIZE);

    for (f = 0; f < iSIZE; f++)

    {

        printf("Informacion de la Coordenada #%i. \n", f + 1);

        printf("\tIngresa X: ");

        scanf("%i", &pArray[f].x);

        fflush(stdin);

        printf("\tIngresa Y: ");

        scanf("%i", &pArray[f].y);

        fflush(stdin);

    }

    printf("Impresion de la Informacion de %i Coordenada(s). \n", iSIZE);

    for (f = 0; f < iSIZE; f++)

    {

        printf("Informacion de la Coordenada #%i --> (%i, %i) \n", f + 1, pArray[f].x, pArray[f].y);

    }

    printf("\n \n");

}

/\*\*Este Procedimiento Ense�a como una Estructura Puede Tener un Vector como Elemento\*\*/

#include <stdio.h>

#include <string.h>

typedef struct employee{

    char cName[15];

    int  iAge;

}emp;

void structWithArrayAsElement()

{

    emp e1, e2 = {"Daniel", 20};

    ///NOTA: Puedes usar el operador de asignacion (=) para copiar la informacion de una estructura a otra

    // Esta copia es real y no es una comparticion de memoria

    // Incluso si la estructura incluye un vector, dicho vector es copiado a su vector correspondiente de la estructura de destino

    e1 = e2;

    printf("Empleado 1.\n\tNombre: %s\n\tEdad: %i\n", e1.cName, e1.iAge);

    printf("Empleado 2.\n\tNombre: %s\n\tEdad: %i\n", e2.cName, e2.iAge);

    printf("\n \n");

}

/\*\*Este Procedimiento Ense�a como Podemos Simular Operadores Logicos para Usar con Estructuras\*\*/

///Nota: No se pueden usar operadores logicos con estructuras como si fuesen cualquier otro tipo de variable

#include<stdio.h>

typedef struct coordinate{

    int x;

    int y;

}point;

int equalPoint(point p1, point p2) // ==

{

    if (p1.x == p2.x && p1.y == p2.y)

        return 1;

    else

        return 0;

}

int notEqualPoint(point p1, point p2) // !=

{

    if (p1.x != p2.x || p1.y != p2.y)

        return 1;

    else

        return 0;

}

void usingLogicalOperatorsWithStructs()

{

    int f;

    point pInput[2];

    printf("Ingreso de la informacion de la coordenada. \n");

    for (f = 0; f < 2; f++)

    {

        printf("\tIngresa X(%i): ", f + 1);

        scanf("%i", &pInput[f].x);

        fflush(stdin);

        printf("\tIngresa Y(%i): ", f + 1);

        scanf("%i", &pInput[f].y);

        fflush(stdin);

    }

    if (equalPoint(pInput[0], pInput[1]) == 1) //pInput[0] == pInput[1]

        printf("Las dos coordenadas ingresadas son equivalentes. ");

    else if (notEqualPoint(pInput[0], pInput[1]) == 1) //pInput[0]  != pInput[1]

        printf("Las dos coordenadas ingresadas son distintas. ");

    printf("\n \n");

}

/\*\*Este Procedimiento Explora el Uso de Operadores Matematicos con Estructuras\*\*/

///No se pueden usar los operadores matematicos de forma habitual con las variables de tipo estructura

#include<stdio.h>

#include<string.h>

typedef struct rationalNumber{

    int numerator;

    int denominator;

}fraction;

fraction incrementFraction (fraction frIncrement) // ++

{

    printf("\t%i/%i + 1\n\n", frIncrement.numerator, frIncrement.denominator);

    frIncrement.numerator = frIncrement.numerator + frIncrement.denominator;

    return frIncrement;

}

fraction decrementFraction (fraction frDecrement) // --

{

    printf("\t%i/%i - 1\n\n", frDecrement.numerator, frDecrement.denominator);

    frDecrement.numerator = frDecrement.numerator - frDecrement.denominator;

    return frDecrement;

}

fraction subtractFraction (fraction frMinuend, fraction frSubtrahend) // -

{

    printf("\t%i/%i - %i/%i\n\n", frMinuend.numerator, frMinuend.denominator, frSubtrahend.numerator, frSubtrahend.denominator);

    fraction frDifference;

    int iCommonDenominator = frMinuend.denominator;

    if (frMinuend.denominator != frSubtrahend.denominator)

        iCommonDenominator = calculateCommonDenominator(frMinuend.denominator, frSubtrahend.denominator);

    frMinuend.numerator = frMinuend.numerator \* (iCommonDenominator / frMinuend.denominator);

    frSubtrahend.numerator = frSubtrahend.numerator \* (iCommonDenominator / frSubtrahend.denominator);

    //frMinuend.denominator = iCommonDenominator;

    //frSubtrahend.denominator = iCommonDenominator;

    frDifference.numerator = frMinuend.numerator - frSubtrahend.numerator;

    frDifference.denominator = iCommonDenominator;

    return frDifference;

}

fraction addFraction (fraction frAddend1, fraction frAddend2) // +

{

    printf("\t%i/%i + %i/%i\n\n", frAddend1.numerator, frAddend1.denominator, frAddend2.numerator, frAddend2.denominator);

    fraction frSum;

    int iCommonDenominator = frAddend1.denominator;

    if (frAddend1.denominator != frAddend2.denominator)

        iCommonDenominator = calculateCommonDenominator(frAddend1.denominator, frAddend2.denominator);

    frAddend1.numerator = frAddend1.numerator \* (iCommonDenominator / frAddend1.denominator);

    frAddend2.numerator = frAddend2.numerator \* (iCommonDenominator / frAddend2.denominator);

    frSum.numerator = frAddend1.numerator + frAddend2.numerator;

    frSum.denominator = iCommonDenominator;

    return frSum;

}

fraction multiplyFraction (fraction frFactor1, fraction frFactor2) // \*

{

    printf("\t%i/%i \* %i/%i\n\n", frFactor1.numerator, frFactor1.denominator, frFactor2.numerator, frFactor2.denominator);

    fraction frProduct;

    frProduct.numerator = frFactor1.numerator \* frFactor2.numerator;

    frProduct.denominator = frFactor1.denominator \* frFactor2.denominator;

    return frProduct;

}

fraction divideFraction (fraction frDividend, fraction frDivisor) // /

{

    printf("\t%i/%i / %i/%i\n\n", frDividend.numerator, frDividend.denominator, frDivisor.numerator, frDivisor.denominator);

    fraction frQuotient;

    frQuotient.numerator = frDividend.numerator \* frDivisor.denominator;

    frQuotient.denominator = frDividend.denominator \* frDivisor.numerator;

    return frQuotient;

}

int calculateCommonDenominator (int iDenominator1, int iDenominator2)

{

    int iTemp, f;

    if (iDenominator1 < iDenominator2)

    {

        iTemp = iDenominator1;

        iDenominator1 = iDenominator2;

        iDenominator2 = iTemp;

    }

    for (f = iDenominator1; f <= iDenominator1 \* iDenominator2; f = f + iDenominator1)

    {

        if (f % iDenominator2 == 0)

            break;

    }

    return f;

}

int biggerFraction (fraction fr1, fraction fr2) // >

{

    printf("\t%i/%i > %i/%i ?\n\n", fr1.numerator, fr1.denominator, fr2.numerator, fr2.denominator);

    int iCommonDenominator = fr1.denominator;

    if (fr1.denominator != fr2.denominator)

        iCommonDenominator = calculateCommonDenominator(fr1.denominator, fr2.denominator);

     fr1.numerator = fr1.numerator \* (iCommonDenominator / fr1.denominator);

     fr2.numerator = fr2.numerator \* (iCommonDenominator / fr2.denominator);

     if (fr1.numerator > fr2.numerator)

            return 1;

    else

        return 0;

}

int smallerFraction (fraction fr1, fraction fr2) // <

{

    printf("\t%i/%i < %i/%i ?\n\n", fr1.numerator, fr1.denominator, fr2.numerator, fr2.denominator);

    int iCommonDenominator = fr1.denominator;

    if (fr1.denominator != fr2.denominator)

        iCommonDenominator = calculateCommonDenominator(fr1.denominator, fr2.denominator);

    fr1.numerator = fr1.numerator \* (iCommonDenominator / fr1.denominator);

    fr2.numerator = fr2.numerator \* (iCommonDenominator / fr2.denominator);

    if (fr1.numerator < fr2.numerator)

        return 1;

    else

        return 0;

}

int equivalentFraction (fraction fr1, fraction fr2)

{

    printf("\t%i/%i = %i/%i ?\n\n", fr1.numerator, fr1.denominator, fr2.numerator, fr2.denominator);

    if (fr1.numerator \* fr2.denominator == fr2.numerator \* fr1.denominator)

        return 1;

    else

        return 0;

}

int notEquivalentFraction (fraction fr1, fraction fr2)

{

    printf("\t%i/%i != %i/%i ?\n\n", fr1.numerator, fr1.denominator, fr2.numerator, fr2.denominator);

    if (fr1.numerator \* fr2.denominator != fr2.numerator \* fr1.denominator)

        return 1;

    else

        return 0;

}

void inputFractionFromUser(fraction \*frInput)

{

        printf("\nIngresa una fraccion. \n");

        printf("\tIngresa el numerador: ");

        scanf("%i", &(\*frInput).numerator);

        fflush(stdin);

        printf("\tIngresa el denominador: ");

        scanf("%i", &(\*frInput).denominator);

        fflush(stdin);

        printf("Fraccion ingresada --> %i/%i \n\n", (\*frInput).numerator, (\*frInput).denominator);

}

void printFractionResult(fraction frOutput)

{

    printf("El resultado de la operacion es %i/%i \n\n", frOutput.numerator, frOutput.denominator);

}

void menuToOperateUponFractions()

{

    fraction frInput, frNumber, frResult;

    char cBackToMenu[3] = {' ', ' ', '\0'}, cBackToStart[3] = {' ', ' ', '\0'};

    int iOption;

    do{

        inputFractionFromUser(&frInput);

        do{

            printf("\t---> MENU <---\n");

            printf("1.  %i/%i +  1 \n", frInput.numerator, frInput.denominator);

            printf("2.  %i/%i -  1 \n", frInput.numerator, frInput.denominator);

            printf("3.  %i/%i -  Y/X \n", frInput.numerator, frInput.denominator);

            printf("4.  %i/%i +  Y/X \n", frInput.numerator, frInput.denominator);

            printf("5.  %i/%i x  Y/X \n", frInput.numerator, frInput.denominator);

            printf("6.  %i/%i /  Y/X \n", frInput.numerator, frInput.denominator);

            printf("7.  %i/%i >  Y/X ? \n", frInput.numerator, frInput.denominator);

            printf("8.  %i/%i <  Y/X ? \n", frInput.numerator, frInput.denominator);

            printf("9.  %i/%i =  Y/X ? \n", frInput.numerator, frInput.denominator);

            printf("10. %i/%i != Y/X ? \n", frInput.numerator, frInput.denominator);

            printf("\n");

            printf("Ingresa la opcion seleccionada: ");

            scanf("%i", &iOption);

            fflush(stdin);

            switch (iOption)

            {

            case 1:

                frResult = incrementFraction(frInput);

                break;

            case 2:

                frResult = decrementFraction(frInput);

                break;

            case 3:

                inputFractionFromUser(&frNumber);

                frResult = subtractFraction(frInput, frNumber);

                break;

            case 4:

                inputFractionFromUser(&frNumber);

                frResult = addFraction(frInput, frNumber);

                break;

            case 5:

                inputFractionFromUser(&frNumber);

                frResult = multiplyFraction(frInput, frNumber);

                break;

            case 6:

                inputFractionFromUser(&frNumber);

                frResult = divideFraction(frInput, frNumber);

                break;

            case 7:

                inputFractionFromUser(&frNumber);

                frResult.numerator = biggerFraction(frInput, frNumber);

                break;

            case 8:

                inputFractionFromUser(&frNumber);

                frResult.numerator = smallerFraction(frInput, frNumber);

                break;

            case 9:

                inputFractionFromUser(&frNumber);

                frResult.numerator = equivalentFraction(frInput, frNumber);

                break;

            case 10:

                inputFractionFromUser(&frNumber);

                frResult.numerator = notEquivalentFraction(frInput, frNumber);

                break;

            default:

                printf("ERROR: El numero de opcion ingresada no existe. Por favor escoge otra opcion. \n");

                break;

            }

            if (iOption >= 1 && iOption <=10)

            {

                if (iOption >= 1 && iOption <= 6)

                    printFractionResult(frResult);

                else

                {

                    if (iOption == 7 || iOption == 8)

                        printf("La inecualidad ");

                    else

                        printf("La ecuacion ");

                    if (frResult.numerator == 1)

                        printf("es verdadera. \n\n");

                    else

                        printf("es falsa. \n\n");

                }

                printf("Deseas regresar al menu (si/no)?: ");

                scanf("%2s", &cBackToMenu);

                fflush(stdin);

            }

            else

                strcpy(cBackToMenu, "si");

        } while (strcmp(cBackToMenu, "si") == 0);

        printf("\nDeseas ingresar otra fraccion (si/no)?: ");

        scanf("%s", &cBackToStart);

        fflush(stdin);

    } while (strcmp(cBackToStart, "si") == 0);

    printf("\nGracias por usar el sistema. Hasta luego!");

    printf("\n \n");

}

/\*\*Este Procedimiento Ense�a como Crear un Elemento de Tipo Estructura dentro de una Estructura\*\*/

#include <stdio.h>

typedef struct coordinate{

    int x;

    int y;

}point;

typedef struct circumference{

    point center;

    int radius;

}circle;

void usingStructWithinStruct()

{

    circle circle1;

    printf("Ingreso de la Informacion de un Circulo. \n");

    printf("\tIngresa el centro del circulo: \n");

    printf("\t\tIngresa el punto X: ");

    scanf("%i", &circle1.center.x);

    fflush(stdin);

    printf("\t\tIngresa el punto Y: ");

    scanf("%i", &circle1.center.y);

    fflush(stdin);

    printf("\tIngresa el radio del circulo: ");

    scanf("%i", &circle1.radius);

    fflush(stdin);

    printf("\nImpresion de la Informacion del Circulo. \n");

    printf("\tCentro del circulo: (%i, %i) \n", circle1.center.x, circle1.center.y);

    printf("\tRadio del circulo: %i", circle1.radius);

    printf("\n \n");

}