Proceso determina\_si\_el\_numero\_es\_par\_o\_impar

Definir a Como Real;

Escribir Sin Saltar"ingrese un numero; ";

Leer a;

//determina si es entero al restar al número la parte entera y

// determinar si la parte decimal es distinta de cero

Si (a-trunc(a)) <> 0 Entonces

Escribir "el numero no es entero";

SiNo

//saca el resto de la división del número por 2

// si es impar, el resto da como resultado 1

//en cambio para los pares el resto siempre es 0

Si a MOD(2) = 0 Entonces

Escribir "El número es par";

SiNo

Escribir "el número es impar";

FinSi

FinSi

FinProceso

Proceso serie\_de\_fibonacci

// Se trata de una secuencia infinita de números naturales;

// a partir del 0 y el 1, se van sumando a pares, de manera que

// cada número es igual a la suma de sus dos anteriores, de manera

// que: ... 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55

// defino las variables

Definir i,terminos,num Como Entero;

// se pide ingresar el número de terminos

Escribir 'ingrese la cantidad de términos que desea ';

Escribir 'de la serie hasta un máximo de 45';

Leer terminos;

// se impone un mácimo de 45 por las limitaciones del programa

Si terminos>45 Entonces

Escribir 'EL MÁXIMO ES 45, intente nuevamente';

Escribir 'ingrese la cantidad de términos que desea ';

Escribir 'de la serie hasta un máximo de 45';

Leer terminos;

FinSi

Dimension num[46];

// se fijan los primeros 2 valores que son fijos de la serie (0 y 1)

num[0] <- 0;

num[1] <- 1;

Escribir 'La serie de FIBONACCI de ',terminos,' términos es:';

Escribir '';

// se muestran los dos primeros términos de la serie que son fijos

Escribir '0 1 ' Sin Saltar;

Para i<-2 Hasta (terminos-1) Hacer

// se determinan los términos siguientes sumando los dos anteriores

num[i] <- num[i-1]+num[i-2];

// se muestran los términos de la serie

Escribir num[i],' ' Sin Saltar;

FinPara

FinProceso

Proceso determina\_si\_la\_lista\_es\_ascendente\_descendente\_desordenada

Definir num,i,ascendete,descendete Como Entero;

Dimension num[5];

ascendete<-0; //flag ascendeteincializada a cero

descendete<-0; // flag descendete inicializada a cero

// se ingresan los numeros

Para i<-0 Hasta 4 Hacer

Escribir "ingrese un numero; ";

Leer num[i] ;

FinPara

//proceso para determinar si es ascendente, descendete o ninguna de las dos

i<-0; //inicializa el vector en cero

Para i<-0 Hasta 3 Hacer //recorre el vector y temina en uno menos del total

si num[i] < num[i+1] Entonces // aqui compara el elemento con el que sigue

ascendete<-ascendete+1; // si se cumple que es menor que el posterior

FinSi //la flag ascendente se incrementa en uno

si num[i] > num[i+1] Entonces// recorre el vector

descendete<-descendete+1; // aqui se incremente la flab en uno si es descendete

FinSi

FinPara

// aqui se determina según los valores de las flag

//si es ascendete la flag ascendete debe tener valor 4 porque ha recorrido todo el vector

// si es descendete la flag descendete debe tener valor 4

// si no se cumple ninguna de las anteriores estará desordenada

//el valor 4 se debe a que compara el valor de i con el de i+1, con lo cual se recorre hasta el

//valor total del vector -1.

Si ascendete=4 Entonces

Escribir "La lista es ASCENDENTE";

SiNo

Si descendete=4 Entonces

Escribir "La lista es DESCENDENTE";

SiNo

Escribir "La lista está DESORDENADA";

FinSi

FinSi

FinProceso

Proceso intercambio\_de\_lugares

//Crear un programa que lea por teclado un arreglo de 6 números

//y la desplace una posición hacia abajo: el primero pasa a ser el segundo,

//el segundo pasa a ser el tercero y así sucesivamente.

//El último pasa a ser el primero.

//Defino variables y dimensiono num en 6

Definir i, num,primer Como Entero;

Dimension num[6];

// se ingresan los números en un bucle

para i<-0 Hasta 5 Hacer

Escribir Sin Saltar "ingrese el elemento Nº ", (i+1), " de 6";

Leer num[i];

FinPara

//se utiliza un comodin para guardar el primer valor

primer<-num[0];

//se cambian de posicion los valores menos el ultimo

//que se cambia al final por el comodin

Para i<-0 Hasta 4 Hacer

num[i]<-num[i+1];

FinPara

num[5]<-primer;

//Se muestran los valores cambiados de lugar

escribir"Los elementos desplazados quedaràn:";

para i<-0 Hasta 5 Hacer

Escribir Sin Saltar num[i]," ";

FinPara

Escribir "";

FinProceso

Proceso ordenar\_lista\_introduciendo\_un\_numero\_intermedio

// Leer 5 elementos numéricos que se introducirán ordenados

// de forma creciente. Éstos los guardaremos en un arreglo

// de tamaño 6. Leer un número N, e insertarlo en el lugar

// adecuado para que el arreglo continúe ordenado.

Definir num,menor,i,j,dato5 Como Entero;

Dimension num[6];

j <- 0;

// se pide que se ingresen los5 datos ordenados

Escribir 'Ingrese 5 nùmeros ordenados en forma creciente' Sin Saltar;

Para i<-0 Hasta 4 Hacer

Leer num[i];

FinPara

// se pide que se ingrese el dato a incorporar

Escribir 'Ingrese el número a insertar en el arreglo';

Leer num[5];

// se asigna a dato5 el valor ingresado

dato5 <- num[5];

// se muestran todos lo datos ingresados

Escribir 'Los números ingresados son';

Para i<-0 Hasta 5 Hacer

Escribir num[i],' ' Sin Saltar;

FinPara

// se comparan los datos del arreglo con dato5

// para encontrar el lugar que debe ocupar

// usando Y para encontrar que datos cumplen la doble condicion

Para i<-0 Hasta 5 Hacer

Si num[i]<dato5 Y num[i+1]>dato5 Entonces

// se guarda en J cual es el indice del lugar que debe ocupar el dato

j <- (i+1);

FinSi

FinPara

// se recorre el arreglo desde el final hasta el dato posterior al ingresado

// cambiando de lugar en el arreglo de cada dato

Para i<-5 Hasta (j+1) Con Paso -1 Hacer

num[i] <- num[i-1];

FinPara

// se fija el lugar donde estará el DATO ingresado

num[j] <- dato5;

Escribir '';

// se muestra la lista ordenada con el dato incorporado

Escribir 'la lista ordenada es';

Escribir '';

Para i<-0 Hasta 5 Hacer

Escribir num[i],' ' Sin Saltar;

FinPara

Escribir '';

FinProceso

Proceso eliminar\_elemento\_sin\_dejar\_huecos

// Leer por teclado un arreglo de 5 elementos numéricos

// y una posición (entre 0 y 4). Eliminar el elemento

// situado en la posición dada sin dejar huecos.

// define las variables

Definir i,num,posicion,eliminado Como Entero;

Dimension num[5];

// se pide el ingreso de 5 numeros

Escribir 'ingrese 5 números';

Para i<-0 Hasta 4 Hacer

Escribir 'ingrese el elemnto Nº ',i,' ' Sin Saltar;

Leer num[i];

FinPara

// pide que se ingrese la posicion y se repite si el número

// es <0 o >4

Repetir

Escribir 'Ingrese una posición entre 0 y 4' Sin Saltar;

Leer posicion;

Hasta Que posicion>-1 Y posicion<5

// guarda el numero eliminado para mostrarlo al final

eliminado <- num[posicion];

// cambia las posiciones de los elementos para eliminar el hueco del eliminado

Para i<-posicion Hasta 3 Hacer

num[i] <- num[i+1];

FinPara

// muestra la lista final

Escribir 'la lista queda';

Para i<-0 Hasta 3 Hacer

Escribir num[i],' ' Sin Saltar;

FinPara

Escribir '';

// muestra el numero eliminado

Escribir 'El número eliminado es ',eliminado;

Escribir "";

FinProceso

Proceso fusionar\_y\_ordenar\_dos\_listas

Definir num\_a,num\_b,i,fusion,j,comodin,menor Como Entero;

Definir orden Como Logico;

Dimension num\_a[5],num\_b[5],fusion[10];

orden <- Verdadero;

Repetir

orden <- Verdadero;

Escribir 'Ingrese la primera lista de 5 números ordenados en forma creciente';

Para i<-0 Hasta 4 Hacer

Escribir 'elemento ',(i+1),' ' Sin Saltar;

Leer num\_a[i];

FinPara

Para i<-0 Hasta 3 Hacer

Si num\_a[i+1]<num\_a[i] Entonces

orden <- Falso;

FinSi

FinPara

Si orden=Falso Entonces

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Escribir '\* La lista debe estar ordenada en forma ASCENDENTE, inténtelo nuevamente \*';

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

FinSi

Hasta Que orden=Verdadero

Repetir

orden <- Verdadero;

Escribir 'Ingrese la segunda lista de 5 números ordenados en forma creciente';

Para i<-0 Hasta 4 Hacer

Escribir 'elemento ',(i+1),' ' Sin Saltar;

Leer num\_b[i];

FinPara

Para i<-0 Hasta 3 Hacer

Si num\_b[i+1]<num\_b[i] Entonces

orden <- Falso;

FinSi

FinPara

Si orden=Falso Entonces

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Escribir '\* La lista debe estar ordenada en forma ASCENDENTE, inténtelo nuevamente \*';

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

FinSi

Hasta Que orden=Verdadero

j <- 0;

Para i<-0 Hasta 9 Con Paso 2 Hacer

fusion[i] <- num\_a[j];

fusion[i+1]<-num\_b[j];

j <- j+1;

FinPara

Escribir 'la lista fusionada sin ordenar quedará:';

Para i<-0 Hasta 9 Hacer

Escribir fusion[i],' ' Sin Saltar;

FinPara

Escribir ' ';

Para i<-0 Hasta 8 Hacer

Para j<-(i+1) Hasta 9 Hacer

Si fusion[j]<fusion[i] Entonces

comodin <- fusion[i];

fusion[i] <- fusion[j];

fusion[j] <- comodin;

FinSi

FinPara

FinPara

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Escribir '\* La lista fusionada y ordenada se muestra a continuación \*';

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Para i<-0 Hasta 9 Hacer

Escribir fusion[i],' ', Sin Saltar;

FinPara

Escribir '';

FinProceso

Proceso contar\_vocales\_frase

//: Diseñe un programa que permita ingresar una cadena de caracteres,

//y detecte cuántas vocales tiene.

definir frase,vocales Como Caracter;

Definir cantvoc,i,j Como Entero;

Escribir "Ingrese la frase";

leer frase;

frase<-Minusculas(frase);

vocales<-"aeiouáéíóú";

cantvoc<-0;

i<-0;

j<-0;

para i<-0 Hasta Longitud(frase)-1 Hacer

para j<-0 Hasta Longitud(vocales)-1 Hacer

si Subcadena(frase,i,i)=Subcadena(vocales,j,j) Entonces

Escribir "se detecto la vocal ", Subcadena(vocales,j,j);

cantvoc<-cantvoc+1;

FinSi

FinPara

FinPara

Escribir "La frase tiene ",cantvoc," vocales";

FinProceso

Proceso mostrar\_cadena\_mas\_larga

// Calcular la longitud de 2 cadenas de caracteres,

//y mostrar la cadena con la mayor longitud.

Definir cad1,cad2 Como Caracter;

Definir a,b Como Entero;

Escribir "Ingrese la primera cadena de caracteres";

Leer cad1;

Escribir "Ingrese la segunda cadena de caracteres";

Leer cad2;

//se guarda la longitud de cada cadena en a y b

a<-Longitud(cad1);

b<-Longitud(cad2);

// se compara a y b y se muestra la más extensa

si a>b Entonces

Escribir "La cadena más larga es: ", cad1;

SiNo

Escribir "La cadena más larga es: ", cad2;

FinSi

FinProceso

Proceso Eliminar\_espacios\_en\_blanco

// Diseñe un algoritmo que elimine los espacios en blanco de un texto.

// defino las variables a utilizar

Definir cad,separado Como Caracter;

Definir largo,a,b,i,j,k,l Como Entero;

Dimension separado[100];

// poide que se ingrese la frase

Escribir 'Ingrese la cadena para eliminarle los espacios: ';

Leer cad;

// inicializo las variables

i <- 0;

j <- 0;

k <- 0;

l <- 0;

largo <- Longitud(cad);

// separo los caracteres y los pongo en un arreglo para despues detectar los espacios

Para i<-0 Hasta largo-1 Hacer

separado[i] <- Subcadena(cad,i,i);

FinPara

l <- (largo-1);

// compara cada caracter con el ESPACIO y si es un espacio

// corre todos los caracteres que siguen un lugar en el arreglo

Para i<-0 Hasta l Hacer

Si separado[i]=' ' Entonces

j <- i;

Para k<-j Hasta l-1 Hacer

separado[k] <- separado[k+1];

FinPara

l <- l-1;

FinSi

FinPara

// muestra la frase sin espacios

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Escribir 'LA FRASE SIN SEPARACIÓN ES:';

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Para i<-0 Hasta l Hacer

Escribir separado[i] Sin Saltar;

FinPara

Escribir '';

FinProceso

Proceso detectar\_palidromo

// Diseñar un algoritmo que tomando como entrada

// una cadena de texto nos devuelva si es o no un palíndromo.

// Se conoce que se denomina palíndromo a una palabra o frase que,

// ignorando los blancos, se lee igual de izquierda a derecha que

// de derecha a izquierda.

Definir cad,separado,comodin Como Caracter;

Definir largo,i,j,k,l Como Entero;

Definir medio Como Real;

Definir palidromo Como Logico;

Dimension separado[100];

// pide que se ingrese la frase

Escribir 'Ingrese la cadena para determinar si es un palidromo';

Leer cad;

// inicializo las variables

i <- 0;

j <- 0;

k <- 0;

medio <- 0;

largo <- Longitud(cad);

// Escribir 'largo inicial ',largo

palidromo <- Falso;

l <- largo-1;

// separo los caracteres para luego invertirlos

Para i<-0 Hasta largo-1 Hacer

separado[i] <- Subcadena(cad,i,i);

FinPara

Para i<-0 Hasta l Hacer

Si separado[i]=' ' Entonces

j <- i;

Para k<-j Hasta l-1 Hacer

separado[k] <- separado[k+1];

FinPara

l <- l-1;

FinSi

FinPara

// aca determmino si el largo es par o impar y defino la mitad de la frase

// defino el nuevo largo sin espacios

largo <- l+1;

// Escribir 'largo sin espacios ',largo

Si largo MOD (2)=0 Entonces

medio <- largo/2;

// si es par comparo directamente

Para i<-0 Hasta (medio-1) Hacer

Si separado[i]=separado[largo-1-i] Entonces

palidromo <- Verdadero;

SiNo

palidromo <- Falso;

FinSi

FinPara

SiNo

// si es impar determino el termino medio y no se icompara el que esta

// justo en la mitad

medio <- ((largo/2)-0.5);

Para i<-0 Hasta medio-1 Hacer

Si separado[i]=separado[largo-1-i] Entonces

palidromo <- Verdadero;

SiNo

palidromo <- Falso;

FinSi

FinPara

FinSi

// responde el programa si es un palidromo o no segun sea verdadero o falso

Si palidromo=Verdadero Entonces

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Escribir ' LA FRASE ES UN PALIDROMO ';

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

SiNo

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

Escribir ' LA FRASE NO ES UN PALIDROMO ';

Escribir '\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*';

FinSi

FinProceso

//Escribir una función recursiva para elevar un numero a una potencia.

// Implementación del cálculo de una potencia mediante una función recursiva

// El paso recursivo se basa en que a^b = se calcula mutiplicando a por si misma

//tantas veces como sea el valor de b

// Tenieno en cuenta que a^0 = 1

SubProceso potenciacion<-potencia(base,exponente)

//Defino variables

Definir potenciacion Como Entero;

si exponente=0 Entonces

//El cálculo termina cuando el exp es 0

potenciacion<-1;

SiNo

//Cálculo recursivo de la potencia (Base por sí misma tantas

//veces como sea el número del exponente

potenciacion<-base\*potencia(base,exponente-1);

FinSi

FinSubProceso

Proceso potenciacion\_recursiva

//Defino variables

Definir a,b Como Entero;

//Solicito la base y el exponente

Escribir "Ingrese el número a elevar a una potencia:";

Leer a;

Escribir "Ingrese la potencia a la que desea elevar el número anterior:";

Leer b;

//Muestro resultado

Escribir "El resultado es: ", potencia(a,b);

FinProceso

// Implementar un subprograma recursivo que realice la serie Fibonacci.

// SubProceso que calcula los términos de fibonacci

// fijando los dos primeros como 1

SubProceso fibonacci <- fibo(cantidad)

Definir fibonacci Como Entero;

Si cantidad=1 O cantidad=2 Entonces

// Si cantidad es 1 o 2 el resultado es 1 porque son los

// dos primeros términos fijos de la serie

fibonacci <- 1;

SiNo

// Si es mayor a 2 calculo los términos (n)

// sumando los términos anteriores (n-1) y (n-2) en forma recursiva

// f(n)=f(n-1)+f(n-2)

fibonacci <- fibo(cantidad-1)+fibo(cantidad-2);

FinSi

FinSubProceso

Proceso fibonacci\_recursivo

// Defino variables

Definir terminos,i Como Entero;

// Solicito que ingrese el número de términos

Escribir 'Ingrese la cantidad de términos de Fibonacci que desea visualizar:';

Leer terminos;

// Muestro los términos desde el primero hasta el solicitado

Escribir 'La serie de Fibonacci hasta el término ',terminos,' es:';

Para i<-1 Hasta terminos Hacer

Escribir 'El término ',i,' es ',fibo(i);

FinPara

FinProceso