LAB03

Objetivos:

- Uso de instrucciones sencillas: expresiones y asignaciones
- Utilización de bibliotecas de funciones y procedimientos
- Creación de funciones usando expresiones y asignaciones
- Creación de procedimientos usando expresiones y asignaciones

Lab 3. Programas y subprogramas con instrucciones sencillas

3.1. Lista de ejercicios

Estos ejercicios se deben resolver con las instrucciones que se han explicado hasta el momento en clase (asignación, instrucción nula, llamada a otros procedimientos o funciones).

3.1.1. Valor de una función I

Implementa la siguiente función que toma como dato un entero y devuelve otro entero de acuerdo con la siguiente fórmula:

$$f(x) = 3(4 - 5x) - 2(5 - 6x)$$

3.1.2. Valor de una función II

Implementa la siguiente función que toma como dato un entero y devuelve otro entero de acuerdo con la siguiente fórmula:

$$f(a) = 2a(4a + 3) - 4(3a - 7)$$

3.1.3. Valor de una función III

Implementa la siguiente función que toma como dato un entero y devuelve otro entero de acuerdo con la siguiente fórmula:

$$f(c) = (5c - 7)^2$$

3.1.4. Valor de una función IV

Implementa la siguiente función que toma como datos dos reales y devuelve otro real de acuerdo con la siguiente fórmula:

$$f(p,q) = (3p + 2q)(p - 3q)$$

3.1.5. Valor de una función V

Implementa la siguiente función que toma como datos dos enteros y devuelve otro entero de acuerdo con la siguiente fórmula:

$$f(a,b) = 5(2a - b)$$

Comprueba que sale el resultado esperado (-65) para cuando *a*=-3 y *b*=7.

3.1.6. Valor de una función VI

Implementa la siguiente función que toma como datos dos enteros y devuelve otro entero de acuerdo con la siguiente fórmula:

$$f(p,q) = (3p + 2q)(p - 3q)$$

Comprueba que sale el resultado esperado (40) cuando *p*=-2 y *q*=3.

3.1.7. Valor de una función VII

Implementa la siguiente función que toma como datos dos reales y devuelve otro real de acuerdo con la siguiente fórmula:

$$f(x,y) = \frac{2x - y + 3}{x^2 + 2y}$$

Comprueba que sale el resultado esperado (1.488372) cuando *x*=3/2=1.500 e *y*=2/3=1.666667.

3.1.8. Porcentajes

Implementa un procedimiento llamado *porcentajes* que, dados tres enteros A, B y C, devuelva cuál es el porcentaje que supone cada uno de sus números entre el total que forman la suma de ellas tres.

Para hacer el cálculo del porcentaje se puede usar la siguiente fórmula:

$$PorcentajeA = \frac{A}{A + B + C} \times 100$$

3.1.9. Índice de masa corporal

El índice de masa corporal (IMC) es una razón matemática que asocia la masa y la talla de un individuo. También se le conoce como *índice de Quetelet,* porque fue ideada por Adolphe Quetelet.

Crea una función llamada *imc* que calcule el índice de masa corporal de una persona a partir del peso y su altura, según la siguiente fórmula:

$$IMC = \frac{peso\ [kg]}{altura^2\ [m^2]}$$

3.1.10. Sumar dos dígitos

Crea el procedimiento *Sumar* que dados dos caracteres que representan un dígito cada uno, y sabiendo si hay llevada o no, devuelva el dígito que corresponda a su suma e indique si el resultado tiene llevada o no.

Por ejemplo, un dígito es '6' y otro es '2' el resultado es '8' si no hay llevada (llevada es False) y '9' si sí que la hay (llevada es *True*) y llevada valer *False* (porque la suma no tiene llevada). En cambio, si un dígito es '8' y

otro es '7', el resultado debe ser '5' (o '6', si hay llevada) y llevada valer *True*.

3.1.11. Restar dos dígitos

Crea el procedimiento *Restar* que dados dos caracteres que representan un dígito cada uno y sabiendo si hay llevada o no, devuelva el dígito que corresponda a su resta e indique si esa suma tiene llevada o no.

Por ejemplo, el primer dígito es '6' y el segundo es '3' el resultado es '3' si no hay llevada (llevada es *False)* y llevada debe ser *False.* En cambio, si el primer dígito es '7' y el segundo es '9', el resultado debe ser '8' (o '7' si hay llevada) y llevada valer *True*.

3.1.12. Área de un triángulo I

Crea la función *área_triangulo_1* que, dadas la base y la altura de un triángulo calcule su área, según la fórmula que aparece a continuación:

$$A = \frac{b \cdot h}{2}$$

3.1.13. Área de un triángulo II

Crea la función *área_triangulo_2* que, dadas las longitudes de los lados de un triángulo, calcule su área según la fórmula que aparece a continuación:

$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

$$donde\ s = \frac{a+b+c}{2}$$

La biblioteca *Ada.Numerics.Elementary_Functions* contiene la función *sqrt* para calcular la raíz cuadrada de un número real.

3.1.14. Área de un triángulo III

Crea la función *área_triangulo_3* que, dadas *a* y *b* las longitudes de dos de los lados de un triángulo y *C* el ángulo que forman esos dos lados (en grados), calcule su área según la fórmula que aparece a continuación:

$$A = \frac{a \cdot b \cdot \sin(\hat{C})}{2}$$

La biblioteca *Ada.Numerics.Elementary_Functions* contiene la función *sin* para calcular el seno de un ángulo expresado en radianes. Los cálculos necesarios para transformar un ángulo en grados a radianes es la siguiente:

$$radianes = \frac{grados \cdot \pi}{180}$$

La biblioteca *Ada.Numerics.Elementary_Functions* contiene la función *sin* para calcular el seno de un ángulo expresado en radianes. Utiliza el valor de π (Pi) que se define en la biblioteca de programas *Ada.Numerics.*

3.1.15. Área de un triángulo IV

Crea la función *área_triangulo_4* que, dadas *a,* la longitud de uno de los lados, y *B* y *C*, los ángulos contiguos a dicho lado (medidos en grados), calcule su área según la fórmula que aparece a continuación:

$$A = \frac{a^2 \cdot \sin(\hat{B}) \cdot \sin(\hat{C})}{2 \cdot \sin(\hat{B} + \hat{C})}$$

3.1.16. Área de un triángulo V

Crea la función *área_triangulo_5* que, dadas *a,* la longitud de uno de los lados, *B,* un ángulo contiguo al lado a, y A, un lado no contiguo al lado a calcule su

área alguna función anterior y sabiendo que para calcular el ángulo contiguo que falta se puede conseguir utilizando la siguiente equivalencia:

$$\hat{C} = 180º - \hat{B} - \hat{A}$$

3.1.17. Área de un triángulo VI

Crea la función *área_triangulo_6* que, dadas las coordenadas de los vértices de un triángulo, calcule su área obteniendo la distancia de cada uno de los segmentos que lo forman y usando alguna función previa. La fórmula para calcular la distancia entre dos puntos P(x_1,y_1) y Q(x_2,y_2) es la siguiente:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

3.1.18. Hora

Crea un procedimiento llamado *hora* que, dada una cantidad de segundos, devuelva el número de horas minutos y segundos que representan esos segundos.

3.1.19. Segundos

Crea una función llamada *segundos* que, dada una hora del día definida en horas, minutos y segundos, nos devuelva cuántos segundos han transcurrido desde las 00:00 horas de ese día.

3.1.20. Sumar horas

Crea el procedimiento *sumar_horas* que, dados dos lapsos de tiempo medidos en horas, minutos y segundos, los sume, y devuelva la cantidad de horas minutos y segundos que representan en total. Para resolver el problema utiliza los programas de los dos apartados anteriores. Por ejemplo, dados *5h, 40m, 2s* y *1h, 50m, 7s*, el resultado esperado sería: 7h 30m 9s.

3.1.21. Entero a Dígitos

Crea el procedimiento *entero_a_digitos* que, dado un número entero positivo, desglosa su valor separándolo en *unidades, decenas, centenas* y *unidades de millar.*

3.1.22. Swap (intercambiar)

Crea el procedimiento *swap* que, dados dos valores de tipo carácter, intercambie sus valores. Es decir, que, si se le pasan los valores 'A' y 'Z', entonces debe devolver 'Z' y 'A'.

3.1.23. Valor

Crea la función *valor* que, dadas las unidades, decenas, centenas y unidades de millar, devuelve el valor del entero correspondiente.

3.1.24. El polinomio de Shaw-Basho

Crea una función llamada *Shaw_Basho* que, dado un valor de un entero positivo X, devuelva el valor asociado a ese polinomio para esa X. Los valores de este polinomio se han hecho famosos por la serie *Lost.* Los sucesivos valores para X=0 en adelante son 4, 12, 35, 89, 213, 511, 1194, 2622, 5346, 10150, 18093… El polinomio tiene la siguiente expresión:

$$f(x) = \frac{1}{120}(42x^5 - 305x^4 + 1100x^3 - 895x^2 + 1018x + 480)$$

3.1.25. Múltiplo de diez

Crea una función llamada *múltiplo_de_diez* que, dado un entero positivo *X*, devuelva *true* cuando ese entero sea múltiplo de 10 y *false* en caso contrario.

3.1.26. Área del círculo

Crea un procedimiento llamado *area_circulo* que, dada la longitud del radio, calcule el área del círculo que define, según la fórmula siguiente: $A(círculo) = \pi r^2$. Utiliza el valor de π (Pi) que se define en la biblioteca de programas *Ada.Numerics.*

3.1.27. Volumen del cilindro

Crea la función *volumen_cilindro* que, dada la longitud del radio de la base, *r*, y la altura, *h*, calcule el volumen del cilindro usando el programa *area_circulo* y sabiendo que el volumen del cilindro se calcula multiplicando el área de la base por la altura. O sea, se calcula usando la siguiente fórmula: $V(cilindro) = A(circulo) \cdot h$.

3.1.28. Volumen de la esfera

Crea la función *volumen_esfera* que, dada la longitud del radio, calcula el volumen de una esfera con dicho radio. El volumen se calcula usando la siguiente fórmula: $V(esfera) = \frac{4}{3}\pi r^3$.

3.1.29. Volumen de cilindro con semiesfera

Crea el procedimiento llamado *volumen_cilindro_semiesfera* que, dado un radio r y una altura h, calcula el volumen de un cilindro de radio r y altura h al que se le ha puesto encima media esfera de radio r. Utiliza para resolverlo las dos funciones anteriores.

3.1.30. Múltiplo de

Crea una función llamada *múltiplo_de,* que, dados dos enteros *X* y *N*, devuelva *true* si *X* es múltiplo de *N* y *false* en caso contrario.

3.1.31. Número complejo en forma trigonométrica

Crea el procedimiento *a_trigonometrica*, que dado un número complejo en formato polar (son dos valores: módulo y argumento), nos devuelve el mismo número complejo representado en forma trigonométrica (son dos reales, *a* y *b,* que representan coordenadas en un plano) y que se calculan usando las fórmulas que aparecen más adelante.

$$a = modulo \cdot cos(argumento)$$

$$b = modulo \cdot \sin(argumento)$$

La biblioteca *Ada.Numerics.Elementary_functions* incluye las funciones *seno (sin)* y *coseno (cos).*

3.1.32. Letra en la posición

Crea la función "+" que, dados un carácter C y un entero X, devuelve un carácter que está X posiciones más adelante en el alfabeto. Por ejemplo, dada la 'A' y +1, el resultado debe ser 'B'; dados Z y -2, el resultado debería ser 'X'.

3.1.33. Es bisiesto

Crea una función llamada *es_bisiesto* que, dado un entero positivo que representa un año, devuelva *true* si dicho año es bisiesto y *false* en caso contrario. Un año es bisiesto, según el calendario gregoriano es aquel que es divisible entre 4, salvo que sea año secular (el último del siglo, el que termina en 00) que no es bisiesto. Aún así, los años seculares que son múltiplos de 400, sí que son bisiestos a pesar de ser múltiplos de 100. Así, por ejemplo, son bisiestos los años 1996, 2020 (múltiplos de 4), 1600, 2000, 2400 (acaban en 00 y son múltiplos de 400) y no son bisiestos los siguientes: 1997, 2021 (no son múltiplos de 4), 1900, 2100, 2500 (múltiplos de 100).

3.1.34. Hoy

Crea un procedimiento llamado *hoy que*, devuelva tres enteros diciendo qué día es hoy (día, mes y año). El programa debe utilizar la biblioteca de programas *Ada.Calendar,* que ofrece una función llamada *clock* para obtener el momento actual (día y hora). Y un procedimiento llamado *Split* que, dado un momento en el tiempo, devuelve el día, mes y año de ese momento y los segundos transcurridos desde las 00:00 de ese día hasta llegar a ese momento. O también se pueden usar las funciones *year, month, day* y *seconds.*

```
-- read the computer's clock to get the current time of day
function Clock return Time;
-- Result = numero de segundos desde 1900 hasta la hora actual

procedure Split (Date   : in  Time;
                 Year   : out Year_Number;
                 Month  : out Month_Number;
                 Day    : out Day_Number;
                 Seconds: out Day_Duration);

function Time_Of (Year   : in Year_Number;
                  Month  : in Month_Number;
                  Day    : in Day_Number;
                  Seconds: in Day_Duration:=0.0) return Time;

-- selector operations
function Year    (Date: in Time) return Year_Number;
function Month   (Date: in Time) return Month_Number;
function Day     (Date: in Time) return Day_Number;
function Seconds (Date: in Time) return Day_Duration;
```

3.1.35. Cuántos días

Crea una función llamada *cuantos_dias* que, dada una fecha de nacimiento, y una fecha actual, indica los días que han transcurrido entre dichas fechas.

¿Sabías que la biblioteca de funciones *Ada.Calendar* tiene una operación que podría solucionar el problema con una llamada a la misma?

3.1.36. Digito como entero

Crea una función que dado un valor de tipo carácter entre '0' y '9', devuelva un entero con ese mismo valor. Así, si se le pasa el carácter '6', debe devolver el entero 6.

3.1.37. Entero como dígito

Crea una función que dado un entero Natural menor que 10, devuelva el carácter que corresponde a dicho dígito. Esta función en la inversa a la anterior. Por ejemplo, si se le pasa el numero natural 0, el resultado debe ser el carácter '0'.

3.1.38. Sumar dígitos carácter

Crea un procedimiento llamado *Suma* que, dados dos caracteres que representan cada uno a un dígito (sólo válidos los caracteres del '0' a '9'), Devuelve dos resultados: (1) el dígito correspondiente al resultado de su suma y (2) si dicha suma tiene llevada o no.

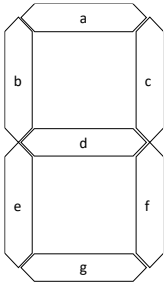
Por ejemplo, sumar '5' y '4' debe devolver '9' y False. Y sumar '5' y '9' debe devolver '4' y True (La suma da 14, el dígito de la suma es 4 y el uno se refiere a la llevada).

3.1.39. Codificar y decodificar fecha

Crea una función llamada *codificar_fecha* que, dada una fecha, codifique dicha fecha como un número entero siguiendo el orden *ddmmaaaa*. Por ejemplo, para el 21 de agosto de 1975 (21,8,1975) el resultado esperado es 21081975.

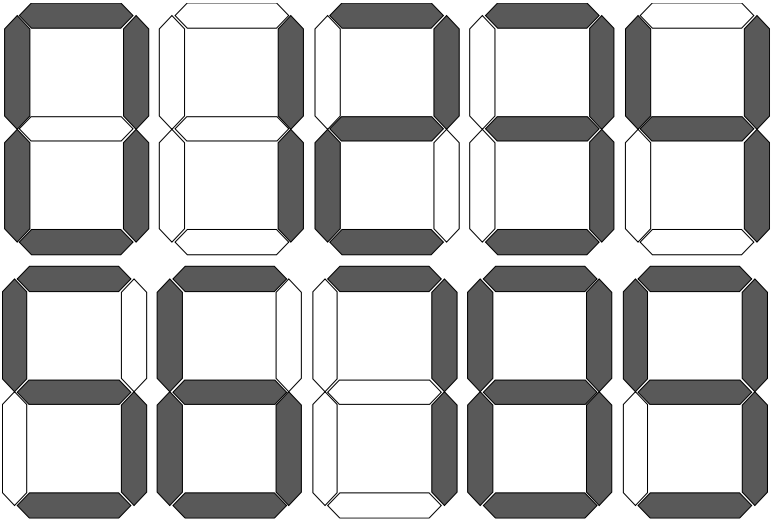
Crea el procedimiento *decodificar_fecha,* que dado un entero de ocho cifras con una fecha codificada siguiendo el método indicado, devuelva el día, el mes y el año correspodiente. Así, para el entero 21081975 (21_08_1975), el resultado debería ser los enteros positivos 21, 8 y 1975.

3.1.40. Display de siete segmentos I



Un display de siete segmentos se programa indicando cuáles de sus segmentos deben estar encendidos. Cada uno de los segmentos se denomina con una letra de la *a* a la *g*. La combinación de estos segmentos es la que produce una serie de dígitos que se podrían utilizar para mostrar valores en un aparato electrónico.

A continuación, aparecen los dígitos del cero al 9 tal y como se representarían. El 1, por ejemplo, tiene activados dos segmentos (d y f). El 7, además tiene activado el segmento a. El 4 tiene activados 4 segmentos (b, c, d y f).



Nosotros podríamos programar un display utilizando un vector de 7 valores booleanos. Si uno de los valores es True, entonces, el correspondiente segmento debe lucir en el display y si es False, entonces no debe aparecer.

Crea un procedimiento llamado *display_7_segmentos* que dado un número entre el 0 y el 9, devuelva qué segmentos deben estar encendidos para representar dicho dígito en un display de siete segmentos. Así si se la pasa un 5, el resultado debe indicar que los segmentos a activar son a, b,d, f y g.