PEC1

Formato y fecha de entrega

La PEC se debe entregar antes del día **21 de marzo de 2017 a las 23:59**. Para la entrega se deberá entregar un fichero en formato **ZIP**, que contenga:

- Este fichero con las respuestas a los distintos ejercicios que se plantean.
- Los ficheros de código que se piden, respetando el nombre indicado en el enunciado.

La entrega se hará en el apartado de entregas de EC del aula de teoría.

Presentación

En esta PEC empezaremos a trabajar tanto con el lenguaje algorítmico como con pequeños programas. Practicaremos con los tipos de datos, las expresiones y las funciones de entrada/salida.

Competencias

Transversales

- Capacidad de comunicación en lengua extranjera.

Específicas

- Capacidad de diseñar y construir aplicaciones informáticas mediante técnicas de desarrollo, integración y reutilización.
- Conocimientos básicos sobre el uso y la programación de los ordenadores, sistemas operativos, bases de datos y programas informáticos con aplicación a la ingeniería.

Objetivos

- Saber seleccionar el tipo de dato más adecuado dado un enunciado
- Saber definir y evaluar expresiones
- Definir pruebas adecuadas para los algoritmos/programas
- Utilizar la composición alternativa

- Crear pequeños programas en C

Recursos

Para realizar esta actividad tenéis a vuestra disposición los siguientes recursos:

Básicos

- Materiales en formato web de la asignatura correspondientes a las 4 primeras semanas indicados en la guía de estudio
- Laboratorio de C

Complementarios

- Internet: La forma más efectiva de encontrar información sobre cualquier duda sobre C es consultarlo a través de un buscador.

Criterios de valoración

Cada ejercicio representa un porcentaje del total de la actividad. Se valorará tanto la validez de las respuestas como su completitud.

- Los ejercicios en lenguaje algorítmico, deben respetar el formato. Recordad que tenéis en la Wiki el *Nomenclátor,* que resume la notación algorítmica que utilizamos en la asignatura.
- Los ejercicios en lenguaje C tienen que compilar para ser evaluados. En tal caso, se valorará:
    - que funcionen
    - que se respeten los criterios de estilo (revisad la Guía de estilo de Lenguaje C de la Wiki)
    - que el código esté comentado
    - que las estructuras utilizadas sean las correctas

[20%] Ejercicio 1: Tipos de datos elementales

Define en lenguaje algorítmico y lenguaje C la**s variables y constantes** adecuadas para los siguientes enunciados.

Ten en cuenta tanto el **tipo** de información como su **rango de valores** y asigna **nombres adecuado**s a cada variable según su significado.

Nota: Las **fechas** las podéis representar como **enteros** en formato **aammdd**.

- Para una aplicación de álbumes fotográficos se quieren registrar los siguientes datos relacionados con una fotografía:
    - el **tamaño** del **archivo** (en KBytes)
    - la **categoría** de la foto (comida, paisaje, gente, animales, viajes, deportes
    - el **tamaño** de la **foto** (**altura** y **anchura**, en píxeles)
    - la **fecha** en que se tomó la foto.

En lenguaje algorítmico:

```
type
  tPhotoType = { Food, Landscape, People, Animals, Travel, Sports }
end type
var
  fileSize: integer;
  photoCategory: tPhotoType;
  photoSizeWidth: integer;
  photoSizeHeight: integer;
  photoDateYear: integer;
  photoDateMonth: integer;
  photoDateDay: integer;
end var
```

En lenguaje C:

```
typedef enum { Food, Landscape, People, Animals, Travel, Sports }
tPhotoType;
unsigned int fileSize;
tPhotoType photoCategory;
unsigned int photoSizeWidth;
unsigned int photoSizeHeight;
unsigned char photoDateYear;
unsigned char photoDateMonth;
unsigned char photoDateDay;
```

- Queremos guardar la información de las notas de un estudiante de FP.
  - Hay que guardar las **notas de tres PEC** y de **una PR** (dividida en dos entregas obligatorias).
  - Las notas de las PEC y de la PR son de **tipo cualitativo** (A, B, C +, C-, D).
  - Una variable que indique si **ha entregado al menos dos PEC**
  - Una variable que indique si **ha entregado las dos PR**
  - Una variable para la **nota final** de la **prueba presencial** (valor numérico)
  - Una variable que nos indique si como **prueba presencial** ha hecho el **examen final** o la **prueba de síntesis**.

En lenguaje algorítmico:

```
type
  tScore = { A, B, CPlus, CMinus, D };
  tTest = { FinalExam, ValidationTest };
end type
var
  scorePEC1: tScore;
  scorePEC2: tScore;
  scorePEC3: tScore;
  scorePR1: tScore;
  scorePR2: tScore;
  hasDeliveredAtLeastTwoPEC: boolean;
```

```
  hasDeliveredAllPR: boolean;
  scoreFinal: integer;
  onSiteTest: tTest;
end var
```

En lenguaje C:

```
  typedef enum { false, true } bool;
  typedef enum  {A ,B, CPlus, CMinus, D } tScore;
  typedef enum { FinalExam, ValidationTest } tTest;
  tScore scorePEC1;
  tScore scorePEC2;
  tScore scorePEC3;
  tScore scorePR1;
  tScore scorePR2;

  bool hasDeliveredAtLeastTwoPEC;
  bool hasDeliveredAllPR;
  unsigned char scoreFinal;
  tTest onSiteTest;
```

[20%] Ejercicio 2: Expresiones

Dada la siguiente definición de variables:

```
type
    tColor = {YELLOW, RED, ORANGE};
end type

var
    color: tColor;
    check: boolean;
    number: integer;
    price: real;
end var
```

Indica si las siguientes expresiones son correctas o no. En caso de que no lo sean, indica el motivo o motivos:

a) `color = RED or price < 100.0 or check or number + 2`

Incorrecto: La variable **color** espera un valor de tipo **tColor.**

Y aunque el primer valor de la asignación (**RED**) es correcto y el operador lógico utilizado (**or**) no permitiría continuar con la evaluación de las siguientes expresiones, estas devuelve valores *boolean (price<100.0), boolean (check)* e *integer (number+2)*.

b) `(number * 2) div price`

Incorrecto: number es de tipo *integer* y price de tipo *real*. Por lo tanto la operación ***div*** no podrá realizarse correctamente. Se debe usar una función de cambio de tipo, por ejemplo *integerToReal()* o *realToInteger()*.

c) (number **mod** 2 = 1) **and** check = **true**

Correcto

d) number < (number+1) < (number+2)

Incorrecto. Ya que la expresión se evalúa de izquierda a derecha. En algún momento se intentará comparar un *booleano (*el resultado de *number < number+1*) con un

integer (number+2).

[30%] Ejercicio 3: Diseño algorítmico

Diseña en lenguaje algorítmico un algoritmo que:

- Lea un **entero** por teclado...
  - Y en caso de que...
    - sea un número par
    - divisible por 3
    - corresponda al código de una letra minúscula
  - Muestre por pantalla el carácter.
  - En caso contrario, muestre el siguiente mensaje:
    - "The number is not valid"

```
algorithm printCharFromCode
  // Variables declaration
  var
   numberToCheck: integer;
    transformedCodeToChar: char;
   isAValidCode: boolean;
    isAnEvenNumber: boolean;
   isDivisibleByThree: boolean;
    hasALowerCaseCode: boolean;
  end var

 // Read values
 numberToCheck:= readInteger();

 // Check if is an even number
 isAnEvenNumber:= (numberToCheck mod 2 = 0);

 // Check if it is divisible by three
 isDivisibleByThree:= (numberToCheck mod 3 = 0);
```

```
// Check if it has a code of a lower case
// Numbers from 97 to 122 correspond to lower cases characters
// Source: https://www.wikiwand.com/en/ASCII#/Printable_characters
hasALowerCaseCode:= (numberToCheck >= 97) and
                    (numberToCheck <= 122);

 // Check if it is a valid code
 isAValidCode:= (isAnEvenNumber and isDivisibleByThree and
hasALowerCaseCode);

  // Print on screen a message
  if isAValidCode then
   // Print the appropriate character
   transformedCodeToChar = codeToChar(numberToCheck);
   writeChar(transformedCodeToChar);
  else
    // Print an error message
    writeString("The number is not valid");
  end if
end algorithm
```

- Dados **tres reales leídos por teclado**
  - Determinar si...
    - son los tres ángulos de un triángulo (los ángulos suman menos o son igual a 180º)
  - y en caso afirmativo:
    - Indique si es un triángulo **equilátero** (todos los ángulos iguales) con el mensaje “Equilateral triangle”
    - Indique si es un triángulo **rectángulo** (un ángulo de 90º) con el mensaje “Rectangle triangle”
    - Indique si es un triángulo **isósceles** (ángulos: 45º, 45º, 90º) con el mensaje “Isosceles triangle”
    - Indique si es un triángulo **escaleno** (todos ángulos diferentes) con el mensaje “Scalene triangle”

```
algorithm whatTypeOfTriangleIs
  // Variables declaration
  var
     angle1: real;
     angle2: real;
     angle3: real;
     isATriangle: boolean;
    isEquilateralTriangle: boolean;
    isRectangleTriangle: boolean;
    isIsoscelesTriangle: boolean;
    isScaleneTriangle: boolean;
  end var

 // Read values
 angle1:= readReal();
 angle2:= readReal();
 angle3:= readReal();
```

```
// Check if they are angles of a triangle
isATriangle:= (angle1 + angle2 + angle3) <= 180;

// Check if it is equilateral triangle
isEquilateralTriangle:= (angle1 = angle2) and
                        (angle1 = angle3) and
                        (angle2 = angle3);

// Check if it is a rectangle triangle
isRectangleTriangle:= (angle1 = 90) or
                      (angle2 = 90) or
                      (angle3 = 90);

// Check if it is an isosceles triangle
isIsoscelesTriangle:= (angle1 = angle2) or
                      (angle1 = angle3) or
                      (angle2 = angle3);

// Check if it is a scalene triangle
isScaleneTriangle:= (angle1 ≠ angle2) and
                    (angle1 ≠ angle3) and
                    (angle2 ≠ angle3);

// Print message for equilateral triangle
if isATriangle and isEquilateralTriangle then
   writeString("Equilateral triangle");
end if


// Print message for rectangle triangle
if isATriangle and isRectangleTriangle then
   writeString("Rectangle triangle");
end if
```

```
  // Print message for isosceles triangle
  if isATriangle and isIsoscelesTriangle then
       writeString("Isosceles triangle");
  end if

  // Print message for scalene triangle
  if isATriangle and isScaleneTriangle then
       writeString("Scalene triangle");
  end if

end algorithm
```

[30%] Ejercicio 4: Programación en C

Haz un programa en C que resuelva el problema propuesto en cada apartado. Copia el código en la respuesta y guarda cada programa en un archivo con el nombre que se indica en el apartado:

a) [checkMean.c] Programa que lea 5 enteros y un real y muestre por pantalla una "y" si el número real coincide con la media de los 5 números enteros o una "n" en caso contrario.

```
#include <stdio.h>

// Constants declaration
unsigned char totalItems = 5;

// Types definition
typedef enum { false, true } bool;

int main(int argc, char **argv)
{
    // Variables declaration
    unsigned int number1;
    unsigned int number2;
    unsigned int number3;
    unsigned int number4;
    unsigned int number5;
    unsigned int sumOfItems;
    float averageOfNumbers;
    float numberToCheck;
    bool areEquals;

    // Read and save five integer numbers
    printf("a) Please, write five integer numbers \n");
    scanf("%d", &number1);
    scanf("%d", &number2);
    scanf("%d", &number3);
    scanf("%d", &number4);
```

```
    scanf("%d", &number5);

    // Read one float number
    printf("b) Please, write one float number \n");
    scanf("%f", &numberToCheck);

    // Sum all integer numbers
    sumOfItems = number1 + number2 + number3 + number4 + number5;

    // Create an average of all integer numbers
    averageOfNumbers = (float) sumOfItems / (float) totalItems;

    // Check if float number is equal to average number
    areEquals = (numberToCheck == averageOfNumbers);

    // print messages
    if (areEquals) {
        printf("y \n");
    } else {
        printf("n \n");
    }

    return 0;
}
```

b) [capsSwitch.c] Programa que lea un carácter y muestre por pantalla el carácter con capitalización invertida (mayúscula si era minúscula, o minúscula si era mayúscula), para el resto de caracteres debe mostrar el mismo carácter.

```
#include <stdio.h>
#include <ctype.h>

// Types definition
typedef enum { false, true } bool;

int main(int argc, char **argv)
{
    // Variables declaration
    bool isAValidCharacterToConvert;
    unsigned char originalCharacter;
    unsigned char convertedCharacter;

    // Read and save original character
    printf("Please, insert a character \n");
    originalCharacter = getchar();

    // Check if original character is an alphabetic value
    isAValidCharacterToConvert = isalpha(originalCharacter);

    // Print on screen a message
    if (isAValidCharacterToConvert) {
        // First, convert original character to lower case or uppera case
        convertedCharacter = islower(originalCharacter) ?
                             toupper(originalCharacter) :
                             tolower(originalCharacter);

        // Then, print converted character
        printf("%c \n", convertedCharacter);
```

```
    } else {
        // If original character is not an alphabetic value, print original
character
        printf("%c \n", originalCharacter);
    }

    return 0;
}
```

c) [numPal.c] Un programa que lea un entero que tenga 5 dígitos e indique si el número entrado es palíndromo (por ejemplo, 12321 es palíndromo). En caso de que sea palíndromo muestre el mensaje "The number is Palindrome", de lo contrario el mensaje "The number is not Palindrome". Verificar que el número entrado tiene cinco dígitos.

```
#include <stdio.h>

// Types definition
typedef enum { false, true } bool;

int main(int argc, char **argv)
{
    // Constants declaration
    unsigned char digitsRequired = 5;

    // Variables declaration
    unsigned int originalNumber;
    unsigned int reverseNumber = 0;
    unsigned int temporaryNumber;
    unsigned char digitsOfOriginalNumber = 0;
    bool isPalindrome;

    // Read and save original number
       printf("Please, insert a five digit number \n");
    scanf("%d", &originalNumber);

    // Save a copy of original number
    temporaryNumber = originalNumber;

    /*
    * Check if temporary number is a five digits number
    * pre    : temporaryNumber = 12345  | digitsOfOriginalNumber = 0
    * 1º loop: temporaryNumber = 1234   | digitsOfOriginalNumber = 1
    * 2º loop: temporaryNumber = 123    | digitsOfOriginalNumber = 2
```

```
* 3º loop: temporaryNumber = 12     | digitsOfOriginalNumber = 3
* 4º loop: temporaryNumber = 1      | digitsOfOriginalNumber = 4
* 5º loop: temporaryNumber = 0      | digitsOfOriginalNumber = 5
* post   : temporaryNumber = 0      | digitsOfOriginalNumber = 5
*
*/
while (temporaryNumber != 0) {
    temporaryNumber = (temporaryNumber / 10);
    digitsOfOriginalNumber++;
}

// It's not a five digit number...
if (digitsOfOriginalNumber != digitsRequired) {
    // Print on screen an error message and exit
    printf("It's not a five digit number. Please try again. \n");
    return 0;
}

// Save one more time a copy of original number
temporaryNumber = originalNumber;

/*
 * Create a reverse copy of temporary number
 * pre    : temporaryNumber = 12345 | reversedNumber = 0
 * 1º loop: temporaryNumber = 1234  | reversedNumber = 5
 * 2º loop: temporaryNumber = 123   | reversedNumber = 54
 * 3º loop: temporaryNumber = 12    | reversedNumber = 543
 * 4º loop: temporaryNumber = 1     | reversedNumber = 5432
 * 5º loop: temporaryNumber = 0     | reversedNumber = 54321
 * post   : temporaryNumber = 0     | reversedNumber = 54321
 * */
while (temporaryNumber != 0) {
    reverseNumber = (reverseNumber * 10) + (temporaryNumber % 10);
    temporaryNumber = temporaryNumber / 10;
}
```

```
    // Check if original number is equal to reverse number
    isPalindrome = (originalNumber == reverseNumber);

    // Print on screen a message
    if (isPalindrome) {
        // Success message
        printf("The number is Palindrome \n");
    } else {
        // Error message
        printf("The number is not Palindrome \n");
    }

    return 0;
}
```