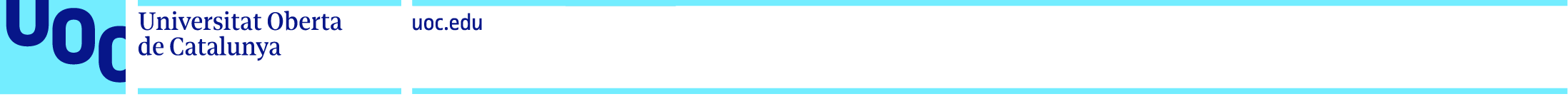
****

**Fundamentos de Programación**

PEC1 - 20181

Fecha límite de entrega: 01/10/2018

Estudiante

**Apellidos: de Lima Barbeiro Campus**

**Nombre: Lidiane Aparecida**

**Objetivos**

* Saber identificar las variables de tipos básicos necesarias para resolver un problema
* Saber leer y escribir variables de tipos básicos

**Formato y fecha de entrega**

La PEC se debe entregar antes del día 01 de octubre de 2018 a les 23:59.

Se debe entregar un fichero en formato ZIP, que contenga:

● Este mismo documento con la respuesta del ejercicio 1 y el último apartado del ejercicio 2

● Un proyecto Codelite que contenga el fichero .c solicitados en el primer apartado del ejercicio 2

La entrega se debe hacer en el apartado de entregas de la AC del aula de teoría.

**Enunciado**

Este semestre la compañía UOCRailway nos ha pedido crear una aplicación para gestionar una compañía de transporte ferroviario. En concreto, los trenes, las mercancías y los pedidos.

Para dar respuesta a esta petición, mediante las PEC, iremos creando una pequeña parte de la aplicación. Esta gestionará los trenes. El resto lo completaremos en las prácticas.

Para empezar nos piden lo siguiente:

Ejercicio 1: Declaración de variables [50 %]

Diseñar en lenguaje algorítmico un algoritmo que haga los siguiente:

Apartado a [40%].

Declare las variables y tipos enumerados necesarios para gestionar los datos de un tren. Para cada tren, de momento, es necesita guardar la siguiente información:

* Un identificador de tipo entero, que es el número que identifica el tren.
* Un entero que indique el año de fabricación del tren.
* Un enumerado que indique el tipo de propulsión. Éste puede ser ELECTRICAL, CARBON, SOLAR, GASOLINE, DIESEL, MAGNETIC.
* Un entero que indique el número de vagones del tren.
* Un real que indique el volumen en metros cúbicos de un vagón, teniendo en cuenta que todos los vagones son iguales.
* Un carácter que indique el modelo del tren.
* Un booleano que indique si el tren tiene o no locomotora.

Apartado b [30%]. Lea por el canal estándar de entrada los valores de las variables de tipo entero, real y carácter.

Para la lectura se debe indicar en el canal estándar de salida que información se espera que el usuario introduzca. Por ejemplo, para leer una variable *age* correspondiente a la edad de una persona, se debería escribir:

writeString(“Enter the current age: ”);

age:=readInteger();

Apartado c [30%]. Muestre en el canal estándar de salida el valor de las variables leídas, indicando a qué corresponde cada información:

Ejercicio 2: programación en C [50%]

Apartado a [70%] Codificación

Codificar en C el algoritmo del ejercicio 1. En la programación los reales se deben escribir con dos decimales.

Apartado b [30%] Pruebas / Ejecución del algorítmo

Los algorítmos codificados deben siempre probarse. Es decir, se deben ejecutar los programas dando diferentes valores a las variables de entrada y comprobando que la salida corresponde a los valores esperados. A este proceso se le llama realizar *Juegos de pruebas*.

En este apartado se solicita que adjuntéis copias de pantalla de dos ejecuciones del algorítmo. En ellas se deben ver claramente los valores de entrada y salida.

**Criterios de corrección:**

En el ejercicio 1:

* Que se siga la notación algorítmica utilizada en la asignatura. Ved el documento *Nomenclator* en la xWiki.
* Que se sigan las instrucciones dadas y el algoritmo responda al problema planteado.

En el ejercicio 2:

* Que el programa se adecue a las instrucciones dadas.
* Que el programa compile y que funcione según lo solicitado.
* Que se declaren los tipos adecuados según el tipo de datos que representa.
* Que se respeten los criterios de estilo de programación C. Ved la *Guía de estilo de programación en C* que tenéis a la Wiki.

Respuestas

1-) algorithm

type ***propulsion***= {ELECTRICAL, CARBON, SOLAR, GASOLINE, DIESEL, MAGNETIC}

end type

**var**

*identificador*: **integer**;

*anyoFabric*: **integer**;

*cantidVagones*: **integer;**

*modelo*: **char**;

*hayLocomotora*: **boolean**;

propulsion: **propulsion**;

*volumenVagon:* **real;**

**end var**

writeString("Introduzca el número identificador del tren:");

identificador = readInteger();

writeString("Introduzca el número el año de fabricación:");

anyoFabric = readInteger();

writeString("Introduzca el la cantidad de vagones:");

vagones = readInteger();

writeString("Introduzca el volumen que tiene el vagon:");

volumenVagon = readReal();

writeString("Introduzca el modelo del tren:");

modelo = readChar();

writeString("Datos del Tren");

writeString("Datos del Tren: identificador: ");

writeInteger(identificador);

writeString("Año de fabricación: ");

writeInteger(anyoFabric);

writeString("Cantidad de vagones: ");

writeInteger(cantidVagones);

writeString("Volumen total de los vagones: ");

writeReal(cantidVagones\*VOLUMEN\_VAGON);

writeString(("Modelo del Tren: ");

writeChar(modelo);

**end algorithm**

2-)

