Universitat Oberta de Catalunya uoc.edu

Fundamentos de Programación
PEC1 - 20182

Fecha límite de entrega: 04/03/2019

Estudiante
Apellidos: Giménez Gorrís.
Nombre: Álvaro.

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 04 de marzo de 2019 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 UOCAirways nos ha pedido crear una aplicación para gestionar una compañía de transporte aéreo. En concreto, los aviones, los pasajeros y los vuelos.

Para dar respuesta a esta petición, mediante las PEC, iremos creando una pequeña parte de la aplicación. Ésta gestionará los aviones. 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 lo siguiente:

Apartado a [40%].

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

- Un identificador de tipo entero, que es el número que identifica el avión.
- Un carácter que identifique el modelo del avión.
- Un entero que indique el año de fabricación del avión.
- Un enumerado que indique el tipo de uso del avión, que puede serCOMMERCIAL, PRIVATE, GOVERNMENTAL, MILITAR, EXPERIMENTAL, OTHERS.
- Un real que indique el peso del avión, en toneladas (sin pasaje ni combustible ni carga).
- Un real que indique la velocidad máxima del avión (en Km/h)
- Un entero que indique la altura máxima que el avión puede alcanzar
- Un entero que indique el número de motores
- Un entero que indique el número de asientos.
- Un booleano que indique si el avión está activo o no.

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:

En primer lugar, he declarado los tipos enumerados y las variables que he utilizado para realizar el algoritmo.
A continuación, he introducido las instrucciones necesarias para que el programa lea la información que el usuario tenga que aportar y he guardado dicha información en el valor de la variable correspondiente.
Finalmente, he indicado las instrucciones para que el programa muestre la información que el usuario ha introducido en cada variable y a qué característica del avión corresponde cada variable.

```
type
        tUseType={COMMERCIAL, PRIVATE, GOVERNMENTAL, MILITAR,
        EXPERIMENTAL, OTHERS};
end type
algorithm planeInformation
var
        number: integer;                    {plane identifier.}
        model: char;                        {plane model.}
        year: integer;                      {fabrication year.}
        use: tUseType;                      { type of use.}
        weight: real;                       {plane weight.}
        maximumSpeed: real;                 {maximum plane speed.}
        maximumHeight: integer;             {máximum plane height.}
        enginesNumber: integer;             {number of engines.}
        seatsNumber: integer;               {number of seats.}
        activePlane: boolean;               {plane on flight or not.}

end var
        writeString("Enter the identifier number: ");
        number:=readInteger();
        writeString("Enter the plane model: ");
        model:=readChar();
        writeString("Enter the fabrication year: ");
        year:=readInteger();
        writeString("Enter the plane weight: ");
        weight:=readReal();
        writeString("Enter the maximum plane speed: ");
        maximumSpeed:=readReal();
        writeString("Enter the maximum plane height: ");
        maximumHeight:=readInteger();
        writeString("Enter the number of engines: ");
        enginesNumber:=readInteger();
        writeString("Enter the number of seats: ");
        seatsNumber:=readInteger();

        writeString("Plane identification: ");
        writeInteger(number);
        writeString("Plane model: ");
        writeChar(model);
```

```
writeString(“Fabrication year: ”);
writeInteger(year);
writeString(“Plane weight: ”);
writeInteger(weight);
writeString(“Maximum speed: ”);
writeReal(maximumSpeed);
writeString(“Maximum height: ”);
writeReal(maximumHeight);
writeString(“Number of engines: ”);
writeInteger(enginesNumber);
writeString(“Number of seats: ”);
writeInteger(seatsNumber);
```

end algorithm

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

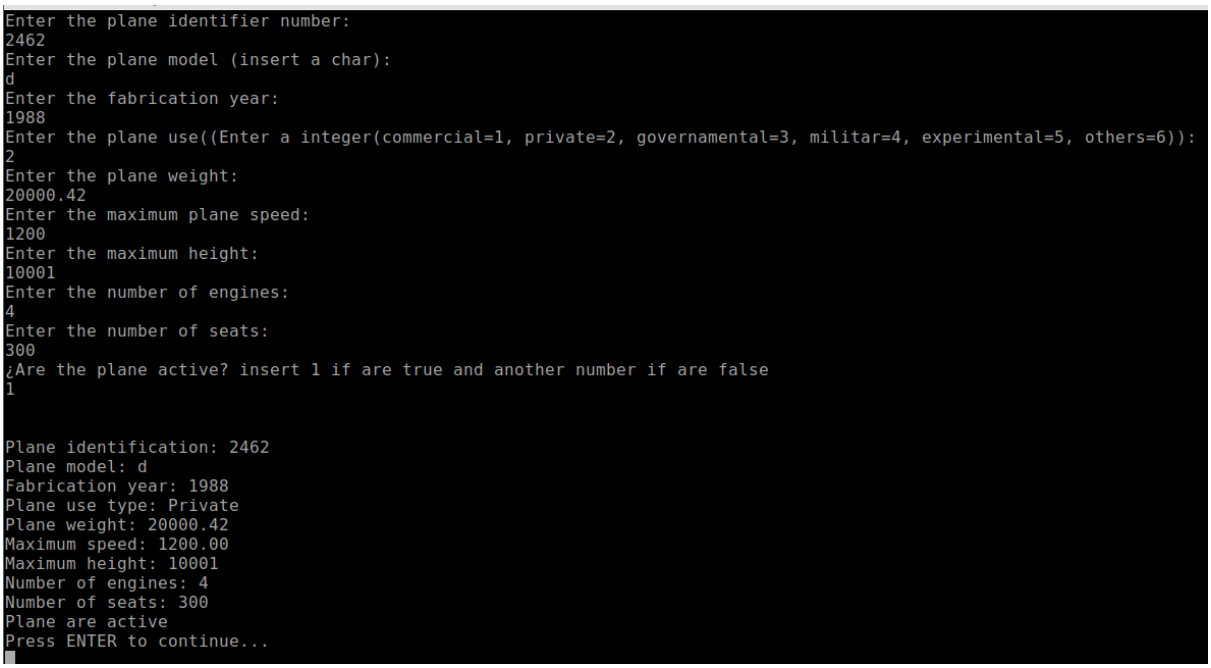
Los algorítmos codificados deben siempre probarse. Es decir, se deben ejecutar los programas dando diferentes valores a las variables de entrada y comprobando que la salida corresponde a los valores esperados. A este proceso se le llama realizar *Juegos de pruebas*.

Apartado b1 [15%]

En este apartado se solita que se diseñe un juego de pruebas.Es decir que completéis la tabla siguiente indicando para unos valores de entrada concretos, que salida se espera de la ejecución del programa. Después comprobad que efectivamente el programa hace lo esperado.

Datos entrada	
Nombre variable	**Valor entrada**
number	2462
model	d
year	1988
weight	20000,42
maximumSpeed	1200
MaximumHeight	10001
enginesNumber	4
seatsNumber	300

Datos de salida
2462
d
1988
20000,42
1200
10001
4
300



Apartado b2 [15%].

Ejecutad el programa entrando datos incorrectos, por ejemplo un real cuando se espera un entero o un carácter cuando se espera un entero. Observad el resultado de la ejecución y comentadlo.

En el caso de introducir letras cuando se esperan enteros, el programa directamente se cierra.

En el caso de introducir reales cuando se esperan enteros, el programa se salta la siguiente instrucción y no deja asignar un valor a la siguiente variable.

En el caso de introducir más de una letra cuando se espera una letra, el programa directamente se cierra.

En el caso de introducir enteros cuando se espera una letra, el programa se salta la siguiente instrucción y no deja asignar un valor a la siguiente variable.

En el caso de introducir reales cuando se espera una letra, el programa salta 3 instrucciones, con lo cual, no permite introducir los valores de las siguiente dos variables.

En el caso de introducir una letra cuando se espera un real, el programa se cierra directamente.

En el caso de introducir un entero cuando se espera un real, el programa se salta una instrucción, con lo que no permite introducir el valor a la siguiente variable.

.

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.