Fundamentos de Programación
PEC1 - 20211

Fecha límite de entrega: **27/09/2021 a las 23:59**

Apellidos: _____
Nombre: _____

Entrega

La PEC deberá entregarse antes del **día 27/09/21 a las 23:59**.

La entrega debe realizarse en el apartado de **entregas de EC** del Aula de teoría. Se corregirá **únicamente** la **última versión** entregada dentro del plazo establecido.

Se debe entregar un único archivo en **formato ZIP**, que contenga:

- Un documento, en **formato PDF**, con el diseño algorítmico. No es necesario incluir todo el enunciado, solo las respuestas.
- El ***workspace* de Codelite**, con el proyecto en C del Ejercicio 2, tal y como se explica en el apartado correspondiente de la *xWiki*.

No respetar el formato de entrega puede conllevar que la actividad no pueda ser corregida , y en cualquier caso **penalizará** su evaluación.

A continuación se citan algunos ejemplos de formatos de entrega incorrectos:

- Documentos en formato distinto a PDF (.docx, .odt, .rtf, .txt, .pages, etc.).
- Ficheros y carpetas sueltos del proyecto en C (.c, .h, .workspace, etc.).

Actualizaciones del enunciado

Cualquier **aclaración**, actualización o corrección de posibles errores del enunciado se publicará en el tablón **del aula de teoría**. Es importante tener en cuenta estas aclaraciones para resolver la actividad, ya que **en caso de discrepancia, tendrán preferencia sobre el enunciado original**.

Objetivos de aprendizaje

Los **objetivos de aprendizaje** de esta PEC son los que se indican a continuación. Estos objetivos de aprendizaje constituyen a su vez, los **indicadores** en los que se basará la corrección y evaluación de la actividad.

Los objetivos de aprendizaje marcados con ★ aparecen por primera vez en esta PEC.

Tratamiento de datos — 50%

- ★ Ser capaz de elegir los tipos básicos de datos de forma correcta. 40%
- ★ Ser capaz de definir y usar los tipos enumerativos de forma correcta. 10%
- ★ Ser capaz de declarar variables de forma correcta. 20%
- ★ Ser capaz de leer datos por el canal estándar de forma correcta. 15%
- ★ Ser capaz de mostrar datos por el canal estándar de forma correcta. 15%

Diseño algorítmico — 20%

- ★ Ser capaz de diseñar un algoritmo funcional y sencillo que dé solución al problema planteado a través un flujo de ejecución de sentencias óptimo. 80%
- ★ Ser capaz de utilizar la notación algorítmica de forma adecuada y aplicar las buenas prácticas en cuanto a la nomenclatura y el uso de comentarios. 20%

Codificación en C — 20%

- ★ Ser capaz de construir un programa en C plenamente funcional, partiendo de un algoritmo diseñado previamente 20%
- ★ Ser capaz de cumplir con los requisitos formales de entrada y salida de datos en C. 20%
- ★ Ser capaz de seguir las normas de estilo establecidas para el lenguaje C. 10%
- ★ Ser capaz de superar los juegos de pruebas y casos extremos con éxito. 50%

Herramientas y entorno — 10%

- ★ Ser capaz de construir un programa completo con la estructura esperada, libre de errores y *warnings.* 100%

Enunciado

La compañía *UOCoworking* nos ha encargado el desarrollo de una aplicación para gestionar una red de centro de trabajo cooperativo. En concreto, deberemos gestionar los centros de trabajo colaborativo y los trabajadores.

Para dar respuesta a esta petición, a lo largo de las distintas PEC iremos desarrollando una parte de la aplicación, que se encargará de la gestión de los centros de trabajo colaborativo, mediante la definición del modelo de datos y la implementación de distintas funcionalidades y algoritmos.

En las prácticas PR1 y PR2, completaremos la aplicación con la gestión de los trabajadores.

Aplicación a desarrollar en esta PEC: **declarar, leer y mostrar los datos básicos de un centro de trabajo colaborativo**.

El desarrollo de la aplicación en esta PEC tiene **tres partes**:

1. Diseño del algoritmo
2. Codificación en C
3. Prueba del programa en C

---

1. Diseño del algoritmo

Diseñar un algoritmo que incluya las funcionalidades que se detallan a continuación.

1.1. Entrada de datos. Declarar las variables para gestionar los datos de un centro de trabajo colaborativo. En esta primera actividad, es necesario que se declaren las siguientes variables para posteriormente poder cumplimentarlas:

- Un identificador numérico del centro.
- El tipo de centro, dependiendo de sus características principales. Puede ser uno de los siguientes: *STARTUPS, FREELANCERS, RURAL, SPECIALIZED, GENERALIST.*
- La categoría del centro, que indica su calidad. Será un número entero.
- El número de espacios de trabajo para alquilar.
- El precio de alquiler mensual de una sala de trabajo, que puede tener decimales.
- La distancia en km del centro de trabajo colaborativo al centro de la ciudad, que puede tener decimales.
- Si el centro tiene o no cocina.
- Si el centro tiene o no un auditorio.

- El porcentaje de ocupación del centro, que puede tener decimales.

En cada caso deberá escogerse el tipo de datos más adecuado. Si fuera necesario el uso de tipos enumerados, deberán ser definidos previamente.

1.2. Entrada de datos. Leer por el canal estándar de entrada, los valores de las variables de tipo **entero** y **real** correspondientes a un centro de trabajo colaborativo. El resto de variables, por el momento, no es necesario leerlas.

La lectura de datos implica una interacción con el usuario y, por lo tanto, éste deberá ser informado de lo que se pide, el tipo de datos básico de la variable, las unidades, el rango válido o los valores posibles (en caso de que los hubiera).

Por ejemplo, para leer la edad de una persona y guardarla en una variable de nombre age*, una posible implementación sería la siguiente:*

```
writeString("CURRENT AGE [YEARS]? (AN INTEGER) >>");
age := readInteger();
```

1.3. Salida de datos. Mostrar por el canal estándar de salida, los valores de las variables enteras y reales leídas anteriormente, indicando claramente a qué corresponde cada dato, así como sus unidades (si las hubiera). El resto de variables, por el momento, no es necesario mostrarlas.

Siguiendo con el ejemplo anterior, una posible solución para la interacción con el usuario sería la siguiente:

```
writeString("AGE [YEARS]:");
writeInteger(age);
```

2. Codificación en C

Codificar en C el algoritmo diseñado anteriormente.

El programa en C debe cumplir con las siguientes particularidades:

- Los números reales deben mostrarse con una precisión de dos decimales.

En el enunciado se pide declarar un conjunto de variables, pero solo se deben leer por teclado algunas de ellas. En C, esto originará uno o más warning *debido a la existencia de variables declaradas sin utilizar. Estos* warning *son normales en esta PEC y no supondrán ninguna penalización.*

3. Prueba del programa en C

Ejecutar y superar los juegos de prueba automáticos disponibles en la herramienta ACME.

*El proceso de validación y corrección de la herramienta ACME se basa en una comprobación **literal** de la salida obtenida por el programa sometido a prueba, con los resultados esperados de los juegos de prueba introducidos previamente. Por ello, los textos de la interfaz de usuario deben ser exactamente idénticos a los esperados.*

Hay que tener en cuenta que el proceso de copiar y pegar textos entre distintos editores, entornos y herramientas puede generar caracteres ocultos que hagan que la comparación de los textos literales sea incorrecta a pesar de que los textos literales aparentemente sean idénticos.

*A modo de ejemplo, a continuación se muestra una ejecución del programa con datos aleatorios, así como la entrada y salida de la herramienta ACME, donde pueden verse los textos literales esperados. En caso de discrepancia, los textos literales que aparecen en la herramienta ACME **tendrán preferencia** respecto a los textos literales que aparecen en este enunciado.*

```
INPUT DATA
ID? (AN INTEGER) >>
3
CATEGORY? (AN INTEGER) >>
4
SPACES (AN INTEGER) >>
5
PRICE [EUR]? (A REAL) >>
20
DISTANCE FROM CITY CENTER [KM]? (A REAL) >>
45.55
OCCUPATION [PERCENT]? (A REAL) >>
34.60
RESULTS
ID: 3
CATEGORY: 4
SPACES: 5
PRICE [EUR]: 20.00
DISTANCE FROM CITY CENTER [KM]: 45.55
OCCUPATION [PERCENT]: 34.60

Press any key to continue...
```

Input data	Expected output
3 4 5 20 45.55 34.60	INPUT DATA ID? (AN INTEGER) >> CATEGORY? (AN INTEGER) >> SPACES? (AN INTEGER) >> PRICE [EUR]? (A REAL) >> DISTANCE FROM CITY CENTER [KM]? (A REAL) >> OCCUPATION [PERCENT]? (A REAL) >> RESULTS ID: 3 CATEGORY: 4 SPACES: 5 PRICE [EUR]: 20.00 DISTANCE FROM CITY CENTER [KM]: 45.55 OCCUPATION [PERCENT]: 34.60