Universitat Oberta de Catalunya uoc.edu

Fundamentos de Programación

PEC8 - 20191

Fecha límite de entrega: 25/11/2019

Estudiante

Apellidos: DELGADO FLORES

Nombre: PABLO JOSÉ

Objetivos

- Comprender el tipo de datos tabla y saber definirlo correctamente.
- Implementar recorridos y búsquedas dentro de las tablas.
- Profundizar en la modularización del código utilizando acciones y funciones.
- Profundizar en el uso de parámetros de entrada, parámetros de salida y parámetros de entrada / salida.

Formato y fecha de entrega

La PEC se debe entregar antes del día **25 de noviembre de 2019 a las 23:59**.

Para la entrega se deberá entregar un archivo en formato ZIP, que contenga:

- Este documento con la respuesta del ejercicio 1.
- El workspace de Codelite que contenga **el proyecto y todas las carpetas creadas** en el ejercicio 2a.

Hay que hacer la entrega en el apartado de entregas de EC del aula de teoría.

Enunciado

Siguiendo con la ayuda que proporcionamos a UOCBookings, la compañía nos ha pedido nuestra colaboración para ampliar el programa que les ayude a gestionar sus hoteles. En esta PEC, trabajaremos con el tipo de datos tabla, conjuntamente con la entrada y salida interactiva. Disponemos del siguiente algoritmo, en lenguaje algorítmico, a medio diseñar:

```
const
        MAX_HOTELS: integer = 100;

end const
type
   tTypeHotel = {BUDGET, INN, RESORT, CONDO, LUXURY, COUNTRY}
tHotel = record
        id: integer;
        brand:string;
        name:string;
        city: string;
        category:integer;
        typeHotel: tTypeHotel;
        numRooms:integer;
        priceDouble:real;
        distanceFromCityCenter: real;
        hasPool: boolean;
        hasGym: boolean;
        closeToSubway: boolean;
        percentOccupation: real;
    end record

   tHotelTable = record
        hotels: vector[MAX_HOTELS] of tHotel;
        nHotels: integer;
    end record

end type

algorithm UOCBooking


{... algorithm to complete ...}
end algorithm
```

Nota: En este ejercicio de lenguaje algorítmico debéis utilizar las acciones siguientes.

action hotelsTableLoadDataFromFile(**out** tabHotels: tHotelTable, **in** fileName:**string**); que lee datos del fichero *fileName* y los carga en la tabla *tabHotels* con información de los mismos.

action hotelWrite(**in** h:tHotel); que muestra por el canal estándar los datos del hotel *h*.

function hotelComputePoints (hotel: tHotel, price: **real**, distance:**real**): **integer**; que calcula puntos asociados a un hotel.

action hotelCopy (**out** dst:tHotel, **in** src:tHotel); que copia los datos del hotel *src en dst*

No hace falta diseñarlas.

Ejercicio 1: Diseño en lenguaje algorítmico (40%)

Apartado a: [20%] Diseña la acción *hotelsTableInitialize.* que recibe como parámetro una tabla de tipo *tHotelTable* e inicializa el número de elementos a cero. Para hacer lo que pide la acción, elige cómo debe ser el parámetro (de entrada, de salida o de entrada / salida)

SOLUCIÓN

```
action hotelsTableInitialize(inout selectedHotelsTable:tHotelTable)
    selectedHotelsTable.nHotels:=0;
end action
```

Apartado b: [40%] Diseña la acción *hotelsTableSelect* que tenga los siguientes parámetros

De entrada:

- *hotelsTable* de tipo *tHotelTable*,
- *city* de tipo cadena de caracteres con el nombre de la ciudad donde debe estar el hotel (sin espacios en blancos)
- *price,* de tipo real que representa el precio óptimo para una habitación doble.
- *distance,* de tipo real que representa la distancia óptima del hotel al centro de la ciudad
- *points,* de tipo entero que representa un número de puntos.

De salida:

- *selectedHotelsTable*, de tipo *tHotelTable* con la selección de los hoteles de la tabla *hotelsTable* que están en la ciudad indicada y que según la función *hotelComputePoints* de la PEC7, que podéis utilizar sin volver a diseñar, tengan una puntuación superior o igual a la indicada.

SOLUCIÓN

```
action hotelsTableSelect(in hotelsTable:tHotelTable, in city:string, in
price:real, in distance:real, in points:integer, out
selectedHotelsTable:tHotelTable)
    var
        i:integer;
        j:integer;
        puntuation:integer;
    end var
    j:=1;
    puntuation:=0;
    hotelsTableInitialize(selectedHotelsTable);
    for i:=1 to hotelsTable.nHotels do
        if (hotelsTable.hotels[i].city=city) then
            puntuation:=hotelComputePoints(hotelsTable.hotels[i], price,
distance);
            if (puntuation ≥ points) then
                hotelCopy(selectedHotelsTable.hotels[j],
hotelsTable.hotels[i]);
                selectedHotelsTable.nHotels:=
selectedHotelsTable.nHotels+1;
                j:=j+1;
            end if
        end if
    end for
end action
```

Apartado c: [40%] Diseña la función *hotelsComputeAverageOccupation,* que recibe los siguientes parámetros

- *hotelTable* de tipo *tHotelTable*
- *city* de tipo cadena de caracteres que representa el nombre de una ciudad (sin espacios en blancos)

y devuelve

- la media global de ocupación de los hoteles de la ciudad indicada. La media debe calcularse como el número total de habitaciones ocupadas en los hoteles de la ciudad indicada dividido entre el número total de habitaciones disponibles en dichos hoteles, calculado como porcentaje.

Ejemplo de cómo se calcularía a partir de los siguientes datos:
Hotel 1, 20 hab. en total, 20% ocupación.
Hotel 2, 30 hab. en total, 50% ocupación.
Hotel 3, 50 hab. en total, 10% ocupación.

El resultado sería:
En el hotel 1 tenemos 4 habitaciones ocupadas.
En el hotel 2 tenemos 15 habitaciones ocupadas.
En el hotel 3 tenemos 5 habitaciones ocupadas.

hotelsComputeAverageOccupation = 100(4+15+5)/(20+30+50) = 24%*

Fijaos que no es lo mismo que la media de cada uno de los porcentajes de ocupación de los hoteles, que sería: (20+50+10)/3 = 26.6%

SOLUCIÓN

```
function hotelsComputeAverageOccupation (in hotelTable:tHotelTable, in
city:string):real
    var
        occupation:real;
        j:integer;
        totalRooms:real;
        occupiedRooms:real;
        aux:real;
    end var
    totalRooms:=0;
    occupiedRooms:=0;
    aux:=0;
    for j:=1 to hotelsTable.nHotels do
        if (hotelsTable.hotels[i].city=city) then
            aux:=((integerToReal)hotelTable.hotels[j].numRooms *
hotelTable.hotels[j].percentOccupation) div 100;
            totalRooms:=totalRooms +
(integerToReal)hotelTable.hotels[j].numRooms;
            occupiedRooms:=occupiedRooms+aux;
        end if
    end for
    occupation:=100 * occupiedRooms div totalRooms;
    return occupation;
end function
```

Ejercicio 2: Programación en C (60%)

Apartado a: En este ejercicio hay que codificar en C el ejercicio 1 siguiendo con la estructura de carpetas y ficheros. Concretamente hay que hacer lo siguiente:

1. Descomprime el archivo *zip* que incluye los archivos del proyecto. El proyecto está estructurado en carpetas: carpeta *include* donde está el archivo ***hotel.h*** y carpeta *src* donde están los ficheros ***hotel.c*** y ***main.c.*** También encontraréis los ficheros *hotels1.txt, hotels2.txt, hotels3.txt y hotels4.txt* que os proporcionamos para poder cargar datos en la tabla de hoteles.
2. Dentro del fichero **hotel.h** añadid la declaración del tipo *tHotelTable.*

3. Dentro del fichero **hotel.h** declarad las cabeceras de las acciones y funciones *hotelsTableInitialize, hotelsTableSelect* y *hotelsComputeAverageOccupation.*
4. Dentro del fichero **hotel.c** codificad la acción *hotelsTableInitialize* (ejercicio 1.a)
5. Dentro del fichero **hotel.c**, codificad la acción *hotelsTableSelect* (ejercicio 1.b)
6. Dentro del fichero **hotel.c**, codificad la función *hotelsComputeAverageOccupation* (ejercicio 1.c)
7. Dentro del fichero **main.c**,
    a. completad la declaración de variables para codificar main.c
    b. completad la codificación del main.c para que haga lo siguiente:
        - Lea el nombre del fichero desde el que se desean cargar datos de los hoteles. Debéis usar los ficheros *hotelsi.txt* que os hemos proporcionado con el enunciado.
        - Cargue la tabla de hoteles utilizando la acción *hotelsTableLoadDataFromFile,* que encontraréis en el código proporcionado.
        - Solicite al usuario la ciudad donde quiere el hotel, el precio y la distancia óptimos y el número de puntos que quiere usar para seleccionar hoteles.
        - Haga la selección de los hoteles usando la acción del apartado 1b y presente por el canal estándar de salida los datos de los hoteles resultantes de la selección según los criterios dados. Primero debe decir cuántos hoteles ha encontrado en la selección (0 en caso de que no haya ninguno) y después mostrar los datos de los hoteles encontrados.
        - Independientemente de los hoteles encontrados, muestre por el canal estándar de salida un mensaje indicando cual es el porcentaje de ocupación global de los hoteles de la ciudad indicada, usando la función del apartado 1c.

Criterios de corrección:

En el ejercicio 1:

- Que se siga la notación algorítmica utilizada en la asignatura. Véase documento *Nomenclator* la XWiki de contenido.
- Que se sigan las instrucciones dadas y el algoritmo responda al problema planteado.
- Que se diseñen y se llamen correctamente las acciones y funciones demandadas.

En el ejercicio 2:

- Que el programa se adecue a las indicaciones dadas.
- Que el programa compile y funcione de acuerdo con lo que se pide.
- Que se respeten los criterios de estilo de programación C. Véase la G*uía de estilo de programación en C* que tiene en la Wiki de contenido.
- Que se implemente correctamente la modularización del proyecto, dividiendo el código en carpetas y poniendo lo que corresponde a cada carpeta.