ISI UTN ROSARIO

TRABAJO PRÁCTICO 3
2023

Algoritmos y Estructuras de Datos

Contenido

Introducción

En este TP3 se introduce el concepto de registros y archivos.

Deben codificar en lenguaje Python este TP, excepto la gestión de novedades que solo la deben diagramar en chapin y no programar (es lo que está en amarillo).

Desarrollo

La primera vez que se ejecuta el programa, además de crear todos los archivos del sistema, deben insertar un primer registro obligatorio en el archivo USUARIOS correspondiente al usuario administrador, con los siguientes datos:

Código	Usuario	Clave	Tipo
1	admin@shopping.com	12345	administrador

El código del usuario es numérico entero consecutivo comenzando en 1.

Luego, ya sea la primera ejecución o sucesivas ejecuciones, deberán realizar lo siguiente:

El programa mostrará un menú con 3 opciones:

1. Ingresar con usuario registrado
2. Registrarse como cliente
3. Salir

- Si el operador elige la opción 3, abandonan el programa.
- Si el operador elige la opción 2, el programa deberá permitir crear un usuario de tipo 'Cliente' que no exista actualmente. Estos usuarios se deben dar de alta en el archivo USUARIOS.DAT.
- Para ello deberán ingresar un mail (hasta 100 caracteres de longitud y que no exista otro usuario igual), clave (exactamente de 8 caracteres), y asignar al campo tipoUsuario el valor '*cliente*'. (ver el diseño del archivo USUARIOS.DAT).

  Luego de crear exitosamente el cliente, se vuelve al menú anterior de 3 opciones para que el cliente recién creado pueda ingresar con su usuario y contraseña respectiva.
- Si el operador elige la opción 1, se le solicitará su usuario y contraseña.

  Deben verificar que existan el usuario y su clave guardado en el sistema (archivo USUARIOS.DAT), y de acuerdo al tipo de usuario que está ingresando, presentar el menú correspondiente. Recordar que existe un menú para cada tipo de usuario (Administrador, Dueño de Local y Cliente).

  Reusar el control de claves desarrollado en el TP1 y TP2, permitiendo hasta 3 intentos como máximo.

Para el caso del usuario de tipo Administrador

- Si el usuario que ingresa (por medio de su nombre de usuario y su clave) coincide con un usuario y la respectiva clave guardada, y el mismo es del tipo **Administrador**, entonces se deberá mostrar el menú correspondiente para este tipo de usuario:

MENU ADMINISTRADOR

1. Gestión de locales
2. Crear cuentas de dueños de locales
3. Aprobar / Denegar solicitud de descuento
4. Gestión de Novedades
5. Reporte de utilización de descuentos
0. Salir

si el operador, elige la opción: **1. Gestión de Locales**, se abrirá un submenú de opciones, con:

SUBMENU ADMINISTRADOR

1. Gestión de locales
   a) Crear locales
   b) Modificar local
   c) Eliminar local
   d) Mapa de locales
   e) Volver

Lo mismo sucede al elegir la opción 4:

4. Gestión de novedades *(solo chapin)*
   a) Crear novedades
   b) Modificar novedad
   c) Eliminar novedad
   d) Volver

En ambos casos, con la opción **e) Volver**, se abandona la pantalla de este submenú y se vuelve al menú principal de administrador.

Con la opción **0. Salir** del menú principal, se abandona el **sistema**.

Para el caso de Dueños de locales

- Si el usuario que ingresa (por medio de su nombre de usuario y su clave) coincide con un usuario y la respectiva clave guardada, y el mismo es del tipo **Dueño de Local**, deberán mostrar el menú correspondiente a un usuario de ese tipo:

MENU DUEÑO DE LOCAL

1. Crear descuento
2. Reporte de uso de descuentos
3. Ver novedades *(solo chapin)*
0. Salir

Menú para el Dueño del Local

Para el caso de Clientes

- Si el usuario que ingresa (por medio de su nombre de usuario y su clave) coincide con un usuario y la respectiva clave guardada, y el mismo es del tipo **Cliente**, deberán mostrar el menú correspondiente a un usuario de ese tipo:

MENU CLIENTE
1. Buscar descuentos en local
2. Solicitar descuento
3. Ver novedades *(solo chapin)*
0. Salir

Menú para el Cliente

Que hacer en cada opción

Usuario del tipo Administrador

Desarrollar en Python los 5 puntos del menú del administrador (excepto el punto 4-Gestión de Novedades que se hará solo en chapin):

SUBMENU ADMINISTRADOR

Punto 1. Gestión de Locales

a) Crear locales
b) Modificar local
c) Eliminar local
d) Mapa de locales
e) Volver

Cada vez que se ingresa a Crear, Modificar o Eliminar Locales, mostrarle al operador los locales cargados hasta el momento.

Mantener siempre el archivo de LOCALES **ordenado por nombre** del local, y realizar las búsquedas y la exhibición de datos respetando dicho orden.

- **Crear Locales**: no se sabe cuántos locales se podrán crear cada vez que se ingresa a esta opción (proponer fin de datos acorde).

  De cada local se deberán ingresar: su nombre, su ubicación (por ejemplo '*primer piso, ala este, sector B*'), el rubro al que pertenece el local (puede ser '*indumentaria*', '*perfumería*' o '*comida*', validar que sea alguno de estos 3 valores posibles), y el código del usuario que corresponde al dueño del local (validar que dicho código corresponda a un usuario existente en el archivo USUARIOS.DAT y que el tipo de ese usuario sea '*Dueño de local*').

  No puede haber dos locales con el mismo nombre (validar usando búsqueda dicotómica).

  El código de local es numérico entero consecutivo, comenzando en 1, y lo debe asignar automáticamente el sistema.

  Al finalizar la carga, mostrar la cantidad de locales por rubro, ordenados de manera descendente por cantidad, para poder visualizar rápidamente el rubro con menor y mayor cantidad de locales

(utilizar en memoria RAM arreglo de registros ordenados que contenga esta información siempre actualizada).

Tener en cuenta que estos valores pueden ir cambiando cada vez que se ingresa a la opción **1. Gestión de locales**, ya que se van creando nuevos locales, o modificando (en particular el rubro) o eliminando locales.

Cada vez que se ingresa a **1. Gestión de locales' -> a. Crear locales**, los nuevos locales creados se deben ir agregando a los ya existentes (no perder la información) en archivo LOCALES.DAT.

- **Modificar Local**: ingresar un código de local (validar que exista en archivo LOCALES.DAT). Permitir modificar todos los campos o atributos del local, excepto su código. Tener en cuenta las mismas validaciones que en el Crear Local, referidas al nombre del local, al rubro y al código de usuario.
- **Eliminar Local**: ingresar un código de local (validar que exista en archivo LOCALES.DAT). Preguntar confirmación antes de eliminar el local. Eliminarán un local de forma lógica, sin perder la información, guardando en el campo **estado**, los valores 'A' (activo) o 'B' (baja).

  Por lo cual:

  - Al crear un Local, al campo **estado** se le debe asignar el valor 'A' (activo).
  - Para modificar un Local, el mismo debe estar activo (si está dado de baja preguntar si se lo desea activar, si es así, cambiar 'B' por 'A').
  - Para dar de baja un Local, el mismo debe estar activo.

  Al mostrar los locales (ordenados por nombre), indicar claramente cuáles están activos y cuáles dados de baja.

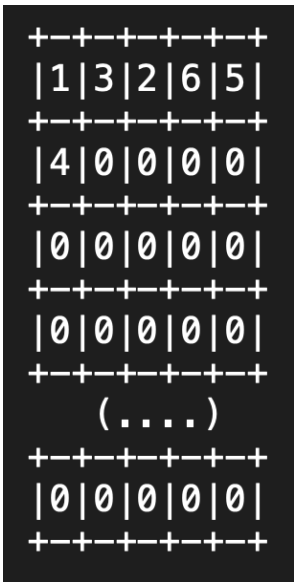
  No considerar a los locales dados de baja en el reporte de cantidad de locales por rubro.

- **Mapa de Locales:** Se debe mostrar el código de cada local, de manera consecutiva, teniendo en cuenta que los locales están ordenados de manera alfabética por su nombre. Ejemplo: si solo existieran 3 locales, donde local 1–Mimo / local 2–Garbarino / local 3–Frávega / al ordenarlos implica que los códigos que se colocarían en el mapa serían: 2 – 3 – 1.

  El mapa del shopping debe tener la forma de la imagen que está en la página 6:

  donde cada código de local debe estar encerrado en un rectángulo como en el diagrama anterior (separado por barras de valor absoluto | a los costados, y por signos +-+ arriba y abajo). Para los locales que aún no han sido ocupados, dejarlos con un número 0.

  Mostrar como máximo los 50 primeros locales del archivo LOCALES (en el caso que haya menos de 50 locales completar con 0, y si hay más de 50 locales registrados, mostrar los primeros 50 de acuerdo al orden utilizado, e indicar debajo del mapa una leyenda "próximamente se habilitará un mapa con los demás locales …"). Identificar de alguna manera a elección de uds. al local que está dado de baja.



- **Punto 2. Crear cuentas de dueños de locales**
  Otra función del Administrador es crear cuentas de dueños de locales que se darán de alta en el archivo USUARIOS.DAT. Para ello deberá ingresar un mail (hasta 100 caracteres de longitud y que no exista otro usuario igual), clave (exactamente de 8 caracteres), y asignar al campo **tipoUsuario** el valor 'D*ueño de local'* (ver el diseño del archivo USUARIOS.DAT).

  Tener en cuenta que seguramente al principio haya que crear la cuenta del dueño de un local, antes de crear efectivamente el local (es decir, el administrador debe pasar por este punto 2 antes de ir al punto 1 del menú).

- **Punto 3. Aprobar / Denegar solicitud de descuento**
  Cuando un dueño de local genere descuentos o promociones de su local (ver más adelante en el apartado correspondientes a las funciones que puede hacer un Dueño de Local), las mismas van a estar registradas en el archivo PROMOCIONES.DAT, y el Administrador las va a ver en esta opción del menú, y los podrá aprobar o denegar (esto dependerá de si las promociones cumplen o no con la política comercial global del shopping).

  Al ingresar a esta opción, el sistema deberá mostrarle al Administrador un listado de todas las promociones en estado='*pendiente' (*siempre que el archivo no esté vacío). Mostrar los datos registrados de cada promoción, y además del código de local, incluir el nombre del mismo.

  Debajo del listado pedirle al Administrador un código de promoción (verificar que existe y que esté pendiente) y si la acepta o rechaza. Actualizar el archivo PROMOCIONES.DAT con el nuevo estado ('*aceptada*' o '*rechazada*') en la promoción correspondiente.

- **Punto 4. Gestión de Novedades**
  La gestión de novedades será realizada únicamente en diagrama chapin, por lo que deben ir al apartado 'Diagramación en Chapin' para ver el detalle de lo que hay que diagramar en este punto.

  En el programa en Python, mostrarán la leyenda 'Diagramado en Chapin' al elegir cada opción del menú que estará solo diagramada.

- **Punto 5. Reporte de utilización de descuentos**
  El administrador podrá monitorear la utilización de los descuentos o promociones en los locales del shopping mediante un reporte gerencial brindado por el sistema.
  Este reporte solicitará al administrador que ingrese un rango de fechas (*Fecha-desde y Fecha-hasta*).
  Se mostrarán los datos de las promociones con estado=*'aprobada'* del archivo PROMOCIONES.DAT y cuya fechaDesdePromo >= *Fecha-desde* y fechaHastaPromo <= *Fecha-hasta*. Agregar además en cada promoción listada, la cantidad de usos que tuvo esa promoción, obteniendo esta información del archivo USO_PROMOCIONES.DAT**.** Obtener la información de la cantidad de usos de cada promoción contando la cantidad de registros del archivo USO_PROMOCIONES.DAT donde coincida el campo codPromo con el mismo campo del archivo PROMOCIONES.DAT, y siempre que la fecha de uso de la promo esté entre la *Fecha-desde* y *Fecha-hasta ingresadas*.

Usuario del tipo Dueño de locales

Desarrollar en Python los puntos 1 y 2 del menú (el punto 3 estará diagramado en chapin):

MENU DUEÑO DE LOCAL
1. Crear descuento
2. Reporte de uso de descuentos
3. Ver novedades
0. Salir

- **Punto 1. Crear descuento**

En esta opción el dueño del local podrá crear (dar de alta descuentos o promociones) para cualquiera de sus locales. Luego el Administrador, por medio de su menú correspondiente, será el encargado de "aprobarlas" o "rechazarlas".

Cada descuento, promoción u oferta tendrá ciertas variables como: el texto descriptivo de la oferta (por ej.: '20% pago contado', '80% de descuento en la segunda unidad', '2x1 para mismo producto', etc.), el rango de fechas vigente de la oferta, y el día de la semana que estará vigente dicha oferta. Respecto del día de la semana, se indica, de lunes a domingo, con un **1** si es válida la oferta ese día y con un **0** si no lo es (siempre dentro del rango de fechas de la vigencia de la oferta).

Al ingresar a la opción **1. Crear descuento**, el sistema debe **listar** los descuentos o promociones vigentes pero solo de el o los locales correspondientes al Dueño de Local logueado (es decir, el codLocal del archivo PROMOCIONES se corresponda con el codLocal del archivo LOCALES.DAT, y estos locales tengan como codUsuario al dueño de local logueado y su estado de local sea 'A').

La promoción está vigente si la fecha de hoy (del sistema) está dentro de la fechaDesdePromo y fechaHastaPromo.

Indicar claramente de cada promoción su estado ('Pendiente', 'Aceptada', 'Rechazada').

Luego de listar las promociones cargadas de los locales correspondientes al dueño de local logueado, el sistema debe solicitar el código de local al que corresponde la promoción que se va a crear.

Ingresar además el texto de la promoción, la vigencia (fecha-desde y fecha-hasta), validando que la fecha-desde sea <= a la fecha-hasta, y que la fecha-desde sea >= a la fecha-actual-del-sistema.

Ingresar los días de la semana en que estará disponible (de lunes a domingo), con un **1** si es válida la oferta ese día y con un **0** si no lo es.

Asignar estado de la promoción en 'pendiente'.

Dar de alta la promoción en el archivo PROMOCIONES.DAT.

- **Punto 2. Reporte de uso de descuentos**

El dueño del local podrá monitorear la utilización de los descuentos o promociones de sus locales del shopping mediante un reporte gerencial brindado por el sistema.

Este reporte solicitará al dueño de locales un rango de fechas (*Fecha-desde y Fecha-hasta*).

Se mostrarán los datos de las promociones con estado='aprobada' del archivo PROMOCIONES.DAT y cuya fechaDesdePromo >= *Fecha-desde* y fechaHastaPromo <= *Fecha-hasta*.

El informe deberá estar agrupado por local (mostrar código y nombre del mismo).

Incluir los datos de cada promoción desde el archivo PROMOCIONES.DAT, agregando la cantidad de usos que tuvo esa promoción. Obtener esta información del archivo USO_PROMOCIONES.DAT, contando la cantidad de registros del archivo USO_PROMOCIONES.DAT donde coincida el codPromo del archivo PROMOCIONES.DAT con el codPromo del archivo USO_PROMOCIONES.DAT, y siempre que la fecha de uso de la promo esté entre la *Fecha-desde* y *Fecha-hasta* ingresadas.

Por ejemplo:

Título del Informe
Fecha desde: 01/12/2022 Fecha hasta: 31 /03/2023

Local 3: Frávega

Código Promo	Texto	Fecha desde	Fecha hasta	Cantidad de usos

Local 2: Garbarino

Código Promo	Texto	Fecha desde	Fecha hasta	Cantidad de usos

- **Punto 3. Ver novedades**

Mostrar la leyenda 'Diagramado en Chapin' al elegir cada opción del menú, ya que estará solo diagramada.

Usuario del tipo Cliente

Desarrollar en Python los puntos 1 y 2 del menú (el punto 3 estará diagramado en chapin):

MENU CLIENTE
1. Buscar descuentos en local
2. Solicitar descuento
3. Ver novedades
0. Salir

- **Punto 1. Buscar descuentos en local**

La búsqueda de un descuento por parte de un cliente consiste en que el cliente ingrese un codLocal y una fecha determinada (validar que esa fecha sea >= a la de hoy (fecha del sistema)).

El sistema mostrará los descuentos o promociones del archivo PROMOCIONES.DAT que se correspondan al codLocal ingresado por el cliente, que estén aprobadas (estado='aprobada') y vigentes (la promoción está vigente si la fecha ingresada por el cliente está dentro de la fechaDesdePromo y fechaHastaPromo). Además, la promoción debe tener un **1** en el día correspondiente al día de la fecha de búsqueda ingresada por el cliente (es decir, se aplica a ese día).

Mostrar:

Codigo Promo	Texto	Fecha desde	Fecha hasta

- **Punto 2. Solicitar descuento**

Si el cliente concurre al shopping un determinado día y quiere hacer uso de una promoción, deberá ingresar el código de la misma. Validar que dicha promoción esté aprobada (estado='aprobada') y vigente (la fecha de hoy -del sistema- está dentro de la fechaDesdePromo y fechaHastaPromo). Además, la promoción debe tener un **1** en el día correspondiente al día de hoy.

Si está todo ok, dar de alta un nuevo registro en el archivo USO_PROMOCIONES.DAT con codCliente correspondiente al cliente logueado en el sistema, el codPromo ingresada por el cliente y fechaUsoPromo igual a hoy (fecha del sistema).

- **Punto 3. Ver novedades**

  Mostrar la leyenda 'Diagramado en Chapin' al elegir cada opción del menú, ya que estará solo diagramada.

Diagramación en Chapin

1) Diseñar el chapin para estas opciones del menú (asumiendo que está logueado un usuario del tipo Administrador):

1. Gestión de novedades
   a) Crear novedades
   b) Modificar novedad
   c) Eliminar novedad
   d) Volver

Las novedades o notificaciones son creadas por el Administrador del sistema a los efectos que los dueños de locales o los clientes estén informados sobre determinadas disposiciones que afectan a la vida comercial del shopping. Una novedad puede ir dirigida a los dueños de locales o a los clientes. Estos dueños de locales o clientes, cuando ingresen al sistema podrán verlas desde sus respectivos menús, por medio de la opción 3. Ver Novedades.

Se solicita en este apartado, diseñar mediante diagrama chapin, un algoritmo que permita en primera instancia mostrar el menú iterativo de Gestión de Novedades al administrador (ya asumimos que el usuario logueado es del tipo Administrador).

- **a) Crear Novedades**: no se sabe cuántas novedades se podrán crear cada vez que se ingresa a esta opción (proponer fin de datos acorde).

  De cada novedad se deberá ingresar: su código (lo ingresará el operador –validar que no exista previamente en el archivo NOVEDADES-), texto de la Novedad, fecha Desde Novedad, fecha Hasta Novedad, tipo de usuario (validar que pueda ser solo 'dueño de local' o 'cliente'). Al crear la novedad en el archivo NOVEDADES asignar el estado='A' (activo). La fecha desde y fecha hasta de una novedad indican el periodo de validez o vigencia de las mismas.

  La fecha Desde Novedad debe ser mayor o igual a la fecha actual del sistema, y además la fecha Desde Novedad debe ser menor o a lo sumo igual que la fecha Hasta Novedad.

  La fecha actual del sistema obtenerla por medio de la función Hoy() ya disponible (no desarrollarla).

  Mantener las novedades ordenadas por código (lo que implicará que las búsquedas por este campo serán mediante dicotómica).

  Cada vez que se ingresa a **4. Gestión de novedadess' -> a. Crear novedades**, las nuevas novedades creadas se deben ir agregando a las ya existentes (no perder la información).

- **Modificar Novedad**: ingresar un código de novedad (validar que exista). Permitir modificar todos los campos o atributos de la novedad, excepto su código. Tener en cuenta las mismas validaciones que en el Crear Novedad, referidas a la fecha desde novedad y fecha hasta novedad, como así también el tipo de usuario al que irá dirigida la novedad.

  Para modificar una novedad, la misma debe estar activa (si está dada de baja preguntar si se la desea activar, si es así, cambiar 'B' por 'A').

- **Eliminar Novedad:** ingresar un código de novedad (validar que exista). Pedir confirmación antes de eliminar la novedad. Eliminaremos una novedad de forma lógica, sin perder la

información, esto es guardando en un atributo denominado **estado**, que puede tomar los valores 'A' (activo) o 'B' (baja). En este caso, al eliminar la novedad, tomará el valor 'B'.

Para dar de baja una novedad, la misma debe estar activa.

En todos los casos, al ingresar a cada opción del menú (crear, modificar o eliminar), preguntar si desea ver las novedades existentes. Al mostrar las novedades (ordenadas por código), indicar claramente cuáles están activas y cuáles dadas de baja.

- **Con la opción d)** Volver al menú principal del Administrador (solo invocarlo a este último –no desarrollarlo-).

2) Diseñar en chapin un único procedimiento parametrizado para la opción 'Ver Novedades', que mediante parámetros, esté preparado para mostrar las novedades dirigidas a un Dueño de Local o a un Cliente, es decir, este procedimiento podrá ser invocado desde el punto 3. Ver Novedades tanto del menú de Dueños de Locales como del menú de Clientes.

Deben desarrollar en chapin sólo el procedimiento **Ver Novedades (…)**.

Este procedimiento debe mostrar todas las novedades del archivo NOVEDADES que cumpla las siguientes condiciones:

- El estado de la novedad debe ser 'A' (activo).
- La fecha actual del sistema debe estar entre fechaDesdeNovedad y fechaHastaNovedad. Para obtener la fecha actual del sistema usar la función Hoy() (sin desarrollarla).
- El tipo de usuario de la novedad debe corresponderse con el que está logueado en el sistema (para ello se recibirá esta información por medio de un parámetro que lo deberán definir claramente en este procedimiento **Ver Novedades**).

Mostrar las novedades que cumplan las condiciones anteriores y ordenadas por código de novedad.

Archivos

USUARIOS.DAT

codUsuario int

nombreUsuario (mail del administrador, del dueño del local o del cliente) **string(100)**

claveUsuario **string(8)**

tipoUsuario (‘administrador’, ‘dueño de local’, ‘cliente’) **string(20)**

LOCALES.DAT

codLocal **int**

nombreLocal **string(50)**

ubicacionLocal **string(50)**

rubroLocal *(‘indumentaria’, ‘perfumería’, ‘comida’).* **string(50)**

codUsuario (*codUsuario de tipo ‘dueño de local’*) **int**

estado: **char**

PROMOCIONES.DAT

codPromo **int**

textoPromo **string(200)**

fechaDesdePromo **date**

fechaHastaPromo **date**

diasSemana **array [0..6] int**

estado (‘pendiente’, ‘*aprobada’, ‘rechazada’)* **string(10)**

codLocal **int**

USO_PROMOCIONES.DAT

codCliente (*codUsuario de tipo ‘cliente’*) **int**

codPromo **int**

fechaUsoPromo **date**

NOVEDADES.DAT

codNovedad **int**

textoNovedad **string(200)**

fechaDesdeNovedad **date**

fechaHastaNovedad **date**

tipoUsuario (*‘dueño de local’, ‘cliente’*) **string(20)**

estado: **char**

Aclaraciones importantes

Tener en cuenta antes de entregar, que el no cumplir con cualquiera de los puntos enunciados debajo puede significar desaprobar o sacarse la nota mínima de aprobación:

- El Chapin debe estar hecho a mano, escaneado y subido en PDF. Deben indicar claramente las variables utilizadas y su tipo de datos, principalmente lo referido a arreglos, registros y archivos.
- NO usar recursividad.
- En Python:
    - NO usar librerías externas no permitidas.
    - NO usar sort, find, join o ningún método similar.
    - El programa debe compilar correctamente.
- TODOS los integrantes del equipo deben entregar EL MISMO trabajo.
- TODOS los archivos deben estar en la carpeta c:\\tp3\\
- En caso de entregar un link a Google Drive, la última fecha de modificación debe ser anterior a la fecha límite de entrega.
- NO se aceptan entregas por e-mail o por wathsapp.