**Evaluación Parcial N°3**

| **Sigla** | **Nombre Asignatura** | **Tiempo Asignado** | **% Ponderación** |
| --- | --- | --- | --- |
| FPY1101 | Fundamentos de Programación | 3h | **30%** |

**1. Situación evaluativa**

| x | **Ejecución práctica** |
| --- | --- |

**2. Agente evaluativo**

| x | **Heteroevaluación** |  |  | **Coevaluación** |  |  | **Autoevaluación** |
| --- | --- | --- | --- | --- | --- | --- | --- |

**3. Instrucciones generales para el/la estudiante**

| Esta es una evaluación que corresponde a una prueba de ejecución práctica y tiene un **30%**deponderaciónsobre la nota final de la asignatura.    El **tiempo** para desarrollar esta evaluación es de **120 minutos** y se realiza de manera **individual** en el **laboratorio**    **La evaluación consiste en:**   * Desarrollo de una aplicación en Python y en entorno de desarrollo Visual Studio Code * Uso de colecciones * Uso de archivos de texto * Uso de librerías estándar de Python * Uso de GitHub como repositorio del proyecto |
| --- |

**4. Evaluación**

| PyGame es una pequeña tienda de videojuegos que se quiere especializar en juegos retro. Hasta el momento, tiene juegos del 2004 hasta el 2008, pero su problema es que la lista de juegos los tienen en un archivo .csv llamado “juegos.csv”. Una muestra pequeña de ese archivo se muestra a continuación:  juegos.csv   | Super Mario 64 DS,1,24.95,Nintendo DS,2004  Lumines: Puzzle Fusion,1,14.95,Sony PSP,2004  WarioWare Touched!,2,22.95,Nintendo DS,2004  Hot Shots Golf: Open Tee,1,12.95,Sony PSP,2004 | | --- |   Cada línea del archivo contiene la información de un juego. La información está separada con comas (,) y el orden es: *nombre del juego, cantidad de jugadores, precio, consola, año*  PyGame le pide a usted que desarrolle una pequeña aplicación que le permita hacer lo siguiente, basado en un menú:   1. **Filtro jugador:** esta opción debe permitir buscar juegos que sean “singleplayer” o “multiplayer”, según lo que elija el usuario, es decir, juegos que el número de jugadores sea de 1, o juegos que el número de jugadores sea mayor a 1. Por ejemplo, *Super Mario 64 DS* es un juego “singleplayer”, ya que solo se juega de a 1. En cambio, *WarioWare Touched!* es un juego “multiplayer”, ya que el número de jugadores es mayor a 1 (2). El resultado de la búsqueda debe ser mostrado (solo los 10 primeros) por pantalla usando una estructura de datos a su elección (lista, diccionario, etc.) 2. **Filtro precio:** esta opción debe permitir ingresar un precio mínimo y un precio máximo. Se debe buscar todos los juegos que estén entre ese rango de precio y mostrarlos por pantalla (solo los 10 primeros) en una estructura de datos a su elección. Si el filtro no entrega juegos, entonces debe mostrar un mensaje que diga: “no hay juegos en ese rango de precios”. 3. **Filtro consola y año:** esta opción debe permitir ingresar el nombre de una consola entre los nombres posibles ('Nintendo DS', 'Nintendo Wii', 'PlayStation 3', 'Sony PSP', 'X360') y un año, y mostrar por pantalla los nombres de los juegos (solo los 10 primeros), ordenados alfabéticamente, que cumplen con los filtros. Si no hay juegos que mostrar, debe mostrar un mensaje que diga: “No hay juegos con esos filtros”. 4. **Escribir archivo:** esta opción debe permitir escribir un archivo llamado: “filtro\_juegos.txt” que escriba la información generada de alguno de los filtros. El usuario debe elegir cual de todos los filtros escribir en el archivo, mediante un pequeño menú. Si el filtro seleccionado no tiene información, entonces no debe escribir el archivo y mostrar un mensaje diciendo: “No hay información que escribir”. Si el archivo es creado, debe mostrar el mensaje: “archivo creado!!” 5. **Salir:** El programa termina, mostrando el mensaje: “Saliendo del sistema…”   Las distintas opciones del menú principal deben estar programadas usando funciones creadas por usted mismo.  **Debe subir su código a github creando un repositorio llamado: “Prueba 3” con un commit: “Versión final”. Debe mostrarle al profesor que lo hizo antes de irse de la sala.**  **Además, deben enviar su código por AVA como de costumbre. El nombre del archivo debe ser: nombre\_apellido\_prueba3.py**  Vea los ejemplos de ejecución a continuación que permitirán aclarar cómo se debe desarrollar la ejecución del programa ***(los caracteres en negrita son datos ingresados por teclado)***.  Ejemplo 1:  \*\*\*MENU PRINCIPAL\*\*\*  1) Filtro jugador.  2) Filtro precio.  3) Filtro consola y año  4) Escribir archivo.  5) Salir  Ingrese opcion: **2**  Ingrese precio minimo: **40**  Ingrese precio maximo: **55**  EL resultado de la busqueda es (los 10 primeros):  ['Guitar Hero III: Legends of Rock', 'Afrika']  \*\*\*MENU PRINCIPAL\*\*\*  1) Filtro jugador.  2) Filtro precio.  3) Filtro consola y año  4) Escribir archivo.  5) Salir  Ingrese opcion: **1**  Singleplayer o Multiplayer (s/m)?: **s**  EL resultado de la busqueda es (los 10 primeros):  ['ï»¿Super Mario 64 DS', 'Lumines: Puzzle Fusion', 'Hot Shots Golf: Open Tee', 'Spider-Man 2', 'The Urbz: Sims in the City', 'Ridge Racer', 'Metal Gear Ac!d', 'Madden NFL 2005', 'Pokmon Dash', 'Feel the Magic XY/XX']  \*\*\*MENU PRINCIPAL\*\*\*  1) Filtro jugador.  2) Filtro precio.  3) Filtro consola y año  4) Escribir archivo.  5) Salir  Ingrese opcion: **4**  \*\*\* Menu de escritura \*\*\*  1) Escribir filtro jugador.  2) Escribir filtro precio.  3) Escribir filtro consola y año.  Ingrese opcion: **3**  No hay informacion que escribir.  \*\*\*MENU PRINCIPAL\*\*\*  1) Filtro jugador.  2) Filtro precio.  3) Filtro consola y año  4) Escribir archivo.  5) Salir  Ingrese opcion: **4**  \*\*\* Menu de escritura \*\*\*  1) Escribir filtro jugador.  2) Escribir filtro precio.  3) Escribir filtro consola y año.  Ingrese opcion: **1**  Archivo creado!!  \*\*\*MENU PRINCIPAL\*\*\*  1) Filtro jugador.  2) Filtro precio.  3) Filtro consola y año  4) Escribir archivo.  5) Salir  Ingrese opcion: **5**  Saliendo del sistema... |
| --- | --- |