Algoritmos y Estructuras de Datos
Examen Final 04/03/2021 – Regulares 2015+ [Python y PE]

Una empresa de producción de noticias necesita un programa que le permita operar con los eventos que ha cubierto. De cada **Evento**, se tiene un *código de identificación* (una cadena que puede tener dígitos y caracteres), un *título* (una cadena corta), una *descripción o resumen del evento* (una cadena con un texto terminado en punto y con palabras separadas por un blanco (por ejemplo: "El terremoto tuvo su epicentro en San Juan y se sintió en todo el norte argentino."), el *costo de producción* por su desarrollo (un valor de tipo float), un número entre 0 y 19 que indica el *tipo de evento* (por ejemplo: 0: Deportivo, 2: Social, 3: Político, etc.), y otro número, pero entre 0 y 9 para indicar el *segmento diario* en el que se emitirá la noticicia o desarrollo para ese evento (por ejemplo: 0: matinal primera hora, 1: matinal media mañana, 2: mediodía, etc.).

En base a lo anterior, desarrollar un programa completo que disponga al menos de dos módulos:

- En uno de ellos, definir la clase **Evento** que represente al registro a usar en el programa, y las funciones básicas para operar con registros de ese tipo. Este módulo deberá contener también la definición de cualquier otro tipo de registro que se indique.
- En otro módulo, incluir el programa principal y las funciones generales que sean necesarias. Para la carga de datos, aplique las validaciones que considere necesarias. El programa debe basarse en un menú de opciones para desarrollar las siguientes tareas:

1. Generar un **arreglo de registros** que contenga los datos de todos los eventos. Puede generarlo cargando los datos en forma manual o generando los datos en forma aleatoria. El arreglo debe permanecer ordenado por código de identificación en todo momento durante la carga. Debe considerar que esta opción puede ser invocada varias veces a lo largo del programa, y que en cada ejecución pueden agregarse tantos registros como desee el operador, sin eliminar los datos que ya estaban cargados. Será considerada la eficiencia de la estrategia de carga y los algoritmos que aplique.
2. Mostrar el arreglo generado en el punto anterior, a razón de un registro por línea en la consola de salida.
3. A partir del arreglo, generar un **archivo binario de registros** que contenga los datos de todos los eventos cuyo monto de producción sea mayor a un valor **p** que se carga por teclado. Cada vez que esta opción se seleccione, el archivo debe crearse otra vez, eliminando los anteriores registros que hubiese contenido.
4. Mostrar todos los datos del archivo que generó en el punto 3, a razón de un registro por línea.
5. Recorra el **archivo** que generó en el punto 3, y genere a partir de él un **arreglo unidimensional de valores de tipo float**, que contenga en cada casilla solo el monto de producción de los eventos del archivo cuyo tipo de evento sea mayor o igual a 5. Muestre el arreglo generado de esta forma, y al final del listado agregue una línea adicional indicando el promedio de los montos mostrados.
6. Determine si existe en el arreglo creado en el punto 1, un evento cuyo código de identificación coincida con el valor **cod** que se carga por teclado. Si existe, muestre sus datos completos, detenga la búsqueda y retorne la cadena contenida en el campo descripción del evento. Si no existe, informe con un mensaje y retorne la cadena "No existe.".
7. A partir del arreglo creado en el punto 1, determine cuántos eventos existen para cada una de las posibles combinaciones entre tipos de evento y segmentos diarios (un total de 20 * 10 = 200 contadores). Genere **todos los contadores posibles**, pero muestre solo los resultados que **correspondan a los tipos de evento mayores a 7**.
8. Tome la cadena retornada en el punto 6 (si existía el registro buscado), y determine cuántas palabras de esa cadena comenzaban con una letra mayúscula y tenían al menos una t y una s (en cualquier orden y no necesariamente seguidas). Como se dijo, la cadena debe terminar con un punto y las palabras deben separarse entre ellas con un (y solo un) espacio en blanco. La cadena debe ser procesada caracter a caracter, a razón de uno por cada vuelta del ciclo que itere sobre ella.

Criterios generales de evaluación:

a.) Desarrollo del programa completo, incluyendo los dos módulos pedidos como mínimo, el menú correctamente planteado, funciones bien diseñadas y parametrizadas (cuando sea apropiado), validaciones cuando sean aplicables y estilo de código fuente: **[máximo: 2 puntos (8.3% del puntaje)]**

b.) Desarrollo correcto de los ítems 1 y 2: **[máximo: 4 puntos (16.6% del puntaje)]**

c.) Desarrollo correcto de los ítems 3 y 4: **[máximo: 4 puntos (16.7% del puntaje)]**

d.) Desarrollo correcto del ítem 5: **[máximo: 4 puntos (16.7 % del puntaje)]**

e.) Desarrollo correcto del ítem 6: **[máximo: 4 puntos (16.7% del puntaje)]**

f.) Desarrollo correcto del ítem 7: **[máximo: 2 puntos (8,3 % del puntaje)]**

g.) Desarrollo correcto del ítem 8: **[máximo: 4 puntos (16.7% del puntaje)]**

Para aprobar el parcial, el alumno debe llegar a un *total acumulado de al menos 60% del puntaje (es decir, alrededor de 14.4 puntos acumulados)*, pero obligatoriamente debe estar desarrollado el programa, funcionando y operativo.

NOTA	PORCENTAJE	CALIFICACIÓN
1		Insuficiente
2		Insuficiente
3		Insuficiente
4		Insuficiente
5		Insuficiente
6	60% a 68%	Aprobado
7	69% a 77%	Bueno
8	78% a 86%	Muy Bueno
9	87% a 95%	Distinguido
10	96% a 100%	Sobresaliente