Universidad San Carlos de Guatemala
Facultad de ingeniería.
Ingeniería en ciencias y sistemas

SISTEMA DE ANÁLISIS DE RENDIMIENTO ACADÉMICO Y MÉTRICAS ESTADÍSTICAS

PONDERACIÓN: 15 pts

Tiempo estimado: 10 hrs

Índice

1. MARCO FORMATIVO

1.1. Valor

Originalidad del Trabajo	Cada estudiante o equipo debe desarrollar su propio código y/o documentación, aplicando los conocimientos adquiridos en el curso.

1.2. Competencia(s)

Tipo de Competencia	
Competencia General	El estudiante debe demostrar una comprensión cabal del lenguaje de programación mediante su aplicación en la resolución de problemas inherentes a las fases de un compilador.
Competencia Específica	El estudiante debe ser capaz de implementar soluciones que involucren la lectura, análisis y manipulación de archivos de texto mediante el uso de estructuras de control, además de aplicar cálculos estadísticos para el análisis de datos académicos.

Objetivo SMART

SMART	Definición	Objetivo redactado
Específico (¿Qué?)	El objetivo es concreto y tangible.	Los estudiantes serán capaces de desarrollar un programa que lea, procese y analice archivos de texto con datos académicos, aplicando técnicas de separación de datos, cálculos estadísticos avanzados (media, mediana, desviación estándar, percentiles) y generación de reportes analíticos coherentes.
Medible (¿Cuánto?)	El objetivo tiene una medida objetiva de éxito.	Lograr que al menos el 85% de las funcionalidades requeridas (lectura de múltiples archivos, separación de datos, cálculos estadísticos, análisis comparativo y generación de reportes) sean implementadas correctamente y validadas en ejecución.
Alcanzable (¿Cómo?)	El objetivo debe ser posible con los recursos disponibles.	Utilizando únicamente C++ y las bibliotecas estándar del lenguaje (STL), junto con las herramientas de desarrollo disponibles en el laboratorio (IDE, consola, compiladores g++/clang++, documentación oficial).
Realista (¿Para qué?)	El objetivo contribuye a metas más amplias.	Fortalecer las habilidades prácticas en manipulación de archivos, procesamiento de datos, aplicación de estadística descriptiva e inferencial, y análisis de información académica, preparando a los estudiantes para proyectos de análisis de datos, sistemas de información y desarrollo de aplicaciones de gestión académica.
A Tiempo (¿Cuándo?)	El objetivo tiene fecha límite o mejor aún un cronograma de hitos de progreso	Cumplir este aprendizaje dentro del período establecido en el cronograma académico o en un plazo máximo de 10 horas de desarrollo efectivo.

2. Enunciado de la Práctica

En el curso de Lenguajes Formales y de Programación, es fundamental que el estudiante comprenda el proceso de análisis y manipulación de información estructurada, simulando el comportamiento de las fases iniciales de un compilador, tales como la lectura, separación, validación y análisis estadístico de datos.

Para ello, se propone el desarrollo de una aplicación en lenguaje C++, ejecutada completamente en consola, que permita registrar, procesar y analizar información relacionada con el rendimiento académico de estudiantes en diferentes cursos universitarios.

La aplicación deberá leer múltiples archivos de texto con formato definido, procesar su contenido mediante el uso de estructuras de control y manejo de cadenas, realizar cálculos estadísticos avanzados y generar reportes analíticos que permitan visualizar el desempeño académico de forma clara y precisa.

Archivos de Entrada

estudiantes.lfp

Este archivo contiene la información general de los estudiantes registrados.

Formato:

carnet,nombre,apellido,carrera,semestre

Donde:

• carnet: Identificador único del estudiante (número entero)

• nombre: Nombre del estudiante (cadena de texto)

• apellido: Apellido del estudiante (cadena de texto)

• carrera: Carrera que cursa (Sistemas, Civil, Industrial, etc.)

• semestre: Semestre actual que cursa (número entero 1-10)

Ejemplo:

202012345,Juan,Pérez,Sistemas,5

202013456,María,López,Industrial,3

202014567,Carlos,González,Civil,7

cursos.lfp

Este archivo contiene la información de los cursos disponibles.

Formato:

codigo,nombre,creditos,semestre,carrera

Donde:

• codigo: Código único del curso (número entero)

• nombre: Nombre del curso (cadena de texto)

• creditos: Cantidad de créditos del curso (número entero 1-8)

• semestre: Semestre en que se imparte el curso (número entero 1-10)

• carrera: Carrera a la que pertenece el curso

Ejemplo:

771,Lenguajes Formales y de Programación,5,3,Sistemas

795,Inteligencia Artificial 1,5,7,Sistemas

348,Resistencia de Materiales,6,5,Civil

notas.lfp

Este archivo contiene las calificaciones de los estudiantes en los diferentes cursos.

Formato:

carnet,codigo_curso,nota,ciclo,anio

Donde:

• carnet: Identificador del estudiante

• codigo_curso: Código del curso

• nota: Calificación obtenida (número decimal 0-100)

• ciclo: Ciclo académico (1S o 2S)

• anio: Año en que se cursó (número entero)

Ejemplo:

202012345,771,85.5,1S,2024

202012345,795,92.0,2S,2024

202013456,771,78.0,1S,2024

Funcionalidades del Sistema

El programa deberá permitir:

• Cargar los archivos de estudiantes, cursos y notas desde el sistema

• Procesar la información utilizando separación por comas

• Almacenar los datos en estructuras adecuadas (vectores, estructuras o clases)

• Realizar cálculos estadísticos avanzados sobre las calificaciones

• Relacionar la información entre los tres archivos

• Generar reportes analíticos en formato HTML

• Mostrar un menú interactivo en consola

Reportes a Generar

El sistema deberá generar los siguientes reportes:

Reporte 1: Estadísticas Generales por Curso

Muestra para cada curso: nombre del curso, cantidad de estudiantes que lo han cursado, nota promedio, nota máxima, nota mínima, desviación estándar y mediana de las calificaciones.

Reporte 2: Rendimiento por Estudiante

Muestra para cada estudiante: nombre completo, carnet, carrera, semestre actual, promedio general de todas sus notas, cantidad de cursos aprobados (nota >= 61), cantidad de cursos reprobados (nota < 61) y créditos acumulados.

Reporte 3: Top 10 Mejores Estudiantes

Lista ordenada de los 10 estudiantes con mejor promedio general, mostrando: posición, carnet, nombre completo, carrera, semestre y promedio.

Reporte 4: Cursos con Mayor Índice de Reprobación

Muestra los cursos ordenados por porcentaje de reprobación (de mayor a menor), indicando: código, nombre del curso, total de estudiantes, cantidad de aprobados, cantidad de reprobados y porcentaje de reprobación.

Reporte 5: Análisis por Carrera

Para cada carrera muestra: nombre de la carrera, cantidad total de estudiantes, promedio general de la carrera, cantidad de cursos disponibles y distribución de estudiantes por semestre.

Menú del Sistema

El programa deberá contar con un menú similar al siguiente:

==========

SISTEMA DE ANÁLISIS ACADÉMICO

==========

1. Cargar archivo de estudiantes
2. Cargar archivo de cursos
3. Cargar archivo de notas
4. Generar Reporte: Estadísticas por Curso
5. Generar Reporte: Rendimiento por Estudiante
6. Generar Reporte: Top 10 Mejores Estudiantes
7. Generar Reporte: Cursos con Mayor Reprobación
8. Generar Reporte: Análisis por Carrera
9. Salir

Seleccione una opción:

2.1 Descripción del problema a resolver

Una universidad desea analizar el rendimiento académico de sus estudiantes con el objetivo de identificar patrones de desempeño, detectar cursos con alta reprobación, reconocer a los estudiantes destacados y tomar decisiones informadas para mejorar la calidad educativa.

Para ello, la universidad proporciona tres archivos de entrada:

• Un archivo con el listado de estudiantes registrados

• Un archivo con los cursos disponibles en el pensum

• Un archivo con las calificaciones obtenidas por los estudiantes

El sistema deberá procesar estos archivos, relacionar la información entre ellos, realizar cálculos estadísticos avanzados y generar reportes analíticos que permitan visualizar de manera clara y precisa el estado académico de la institución.

2.2 Alcance de la práctica

Obligatorio

• Uso del lenguaje C++

• Lectura de archivos .lfp

• Separación de datos mediante comas

• Uso de estructuras de control (if, while, for, switch)

• Implementación de cálculos estadísticos

• Generación de reportes HTML

• Uso de vectores o arreglos para almacenar datos

• Relación de información entre múltiples archivos

• Menú funcional en consola

Opcional

• Manejo de errores al leer archivos (archivos inexistentes, corruptos)

• Implementación de algoritmos de ordenamiento eficientes

• Uso de estructuras o clases personalizadas

• Gráficos estadísticos en los reportes HTML (mediante librerías JavaScript)

2.3 Requerimientos técnicos

La práctica debe desarrollarse haciendo uso de:

- Lectura de archivos mediante clases de funciones nativas del lenguaje.
- Separación y análisis de datos utilizando el método split() de la clase String.
- Uso de estructuras de control (condicionales, ciclos) y colecciones (listas o arreglos) para almacenar y procesar los registros.
- Generación de archivos HTML para reportes de resultados.
- Implementación de un menú basado en consola para la interacción con el usuario.

Los estudiantes deberán documentar su código y organizar el proyecto en un repositorio GitHub, con manual técnico y de usuario.

3. Entregables

Tipo	Descripción
Repositorio en GitHub	Repositorio con el nombre **LFP_<SECCIÓN>_1S2026**, que contenga la carpeta del proyecto **Practica1** con todo el código fuente, así como un archivo **README.md** con las instrucciones de ejecución y ejemplos de uso.
Código Fuente	Implementación completa, organizada y documentada según buenas prácticas de programación. El código debe incluir la lectura de archivos delimitados (split','), el registro de movimientos, generación de reportes en HTML y manejo de excepciones.
Manual Técnico	Documento en **PDF o MD** que describa la estructura del programa, las clases utilizadas, los métodos principales, los requerimientos técnicos y observaciones relevantes del desarrollo. Debe incluir diagramas o pseudocódigo que expliquen la lógica de lectura y procesamiento de archivos.
Manual de Usuario	Documento en **PDF o MD** con instrucciones claras para ejecutar el programa desde consola, cargar archivos, generar reportes y navegar por el menú. Debe incluir ejemplos prácticos y capturas de pantalla del sistema en funcionamiento.

Diagrama de Flujo	Representación gráfica del proceso general del sistema, incluyendo la lectura de archivos, separación de datos con split, procesamiento de movimientos y generación de reportes.
Informe de Desarrollo	Documento en **PDF o MD** que detalle la implementación de la práctica, los retos técnicos encontrados y las soluciones aplicadas. Debe incluir capturas del menú en ejecución, resultados de los reportes generados y conclusiones finales.
Reportes Generados (HTML)	Conjunto de archivos **.html** que muestran los resultados solicitados: historial de movimientos, balance general, top de productos con mayor stock y menor precio, y listado completo de inventario.
Enlace de Entrega (UEDI)	Enlace oficial de entrega en la plataforma **UEDI**, apuntando al repositorio de GitHub con todos los materiales consolidados (código, documentación y reportes).

4. Material de apoyo

Pressman, R. (2014). Ingeniería del Software: Un enfoque práctico. McGraw-Hill. (Capítulos sobre estructuras de control, modularidad y buenas prácticas de desarrollo).

Sebesta, R. W. (2012). Conceptos de Lenguajes de Programación (9ª edición). Pearson Educación.
(Lectura recomendada para comprender la evolución, clasificación y fundamentos de los lenguajes de programación).

Joyanes Aguilar, L. Fundamentos de Programación. McGraw-Hill. (Capítulos sobre manejo de cadenas, lectura de archivos y estructuras de control).

Walpole, R. E., Myers, R. H., Myers, S. L., & Ye, K. (2012). Probabilidad y Estadística para Ingeniería y Ciencias (9ª edición). Pearson.

Stroustrup, B. (2013). The C++ Programming Language (4th Edition). Addison-Wesley.

5. Recursos y herramientas a utilizar

Software:

- Entorno de desarrollo (IDE) o editor de texto para escribir y ejecutar programas (Eclipse, IntelliJ, Visual Studio Code o equivalente).

- Consola del sistema operativo para compilar, ejecutar y probar el programa.

Plataformas:

- **GitHub**: para el repositorio y control de versiones.
- **UEDI**: para la entrega oficial del proyecto y documentación.

Recursos de apoyo:

- Manuales de programación estructurada y fundamentos de diseño algorítmico.
- Artículos y guías sobre análisis y manipulación de datos a partir de archivos de texto.
- Recursos sobre diagramas de flujo y modelado de procesos secuenciales.

Lecturas recomendadas:

- Fundamentos de **estructuras de control** (condicionales y repetitivas).
- Introducción a los **tipos de datos y operaciones básicas**.
- Conceptos básicos sobre **paradigmas y generaciones de lenguajes de programación**.
- Material introductorio sobre **lectura secuencial de archivos y procesamiento de cadenas**.

6. Cronograma

Tipo	Fecha Inicio	Fecha Fin
Asignación de Práctica	10-02-2026	01-03-2026

Rúbrica de Calificación

7.1 Requisitos para optar a la calificación

No.	Criterio de evaluación	Punteo Máximo	Satisfactorio (100%–61%)	Necesita mejorar (60%–0%)
1.1	Uso de estructuras de control (ciclos, condicionales)	1	Utiliza correctamente ciclos y estructuras condicionales para controlar el flujo del programa y resolver los procesos planteados.	El uso de estructuras de control es limitado o incorrecto, generando errores o comportamientos inesperados.
1.2	Manejo de cadenas y separación de datos	1	Implementa correctamente la separación por comas y realiza validaciones adecuadas sobre los datos extraídos.	La separación de datos no se usa correctamente o presenta errores en la validación.
1.3	Organización y legibilidad del código	1	El código está bien estructurado, con indentación, comentarios y nombres de variables apropiados.	El código carece de estructura o claridad, dificultando su lectura y mantenimiento.
1.4	Entrega & Documentación (repo, manuales, diagramas)	1	Entrega completa del repositorio con manual técnico, manual de usuario y diagrama de flujo en el formato solicitado.	La entrega carece de alguno de los documentos requeridos o no sigue el formato establecido.
1.5	Interactividad del menú (funcionalidad básica)	1	El menú permite al usuario navegar entre las opciones principales y ejecutar correctamente las acciones del sistema.	El menú presenta errores o no permite la ejecución fluida de las funcionalidades.
1.6	Pruebas y ejecución correcta	1	El programa compila y ejecuta correctamente, mostrando resultados coherentes con los archivos de entrada.	El programa no compila o genera resultados incorrectos al ejecutar las pruebas.
	Sub-total Habilidades	**6 ptos**		

7.2 Resumen de Puntuaciones

No.	Criterio de evaluación	Punteo Máximo	Satisfactorio (100%–61%)	Necesita mejorar (60%–0%)
2.1	Lectura y procesamiento de archivos (estudiantes.lfp, cursos.lfp, notas.lfp)	2	El sistema lee y procesa correctamente los tres archivos, separando los datos y almacenándolos en estructuras adecuadas.	El sistema no procesa correctamente los archivos o presenta errores en la lectura y separación de datos.
2.2	Implementación de cálculos estadísticos (promedio, mediana, desv. estándar, percentiles)	4	Implementa correctamente todas las funciones estadísticas requeridas: promedio, mediana, desviación estándar y percentiles. Los cálculos son precisos y se aplican adecuadamente.	Las funciones estadísticas presentan errores de cálculo, están incompletas o no se implementaron correctamente.
2.3	Generación de reportes en HTML (6 reportes completos y funcionales)	3	Los 6 reportes HTML muestran la información requerida de forma ordenada, legible y con formato profesional. Incluyen todos los datos solicitados.	Los reportes presentan errores, información incompleta, formato incorrecto o faltan reportes.
	Sub-total Conocimiento	**9 pts**		

7.3 Comentarios Generales

_____

_____

_____

_____

Detalle de la Calificación

No.	Criterio de evaluación	Punteo Máximo	Satisfactorio (100%–61%)	Necesita mejorar (60%–0%)	Punteo Obtenido
1.1	Uso de estructuras de control (ciclos, condicionales)	1	Utiliza correctamente ciclos y estructuras condicionales para controlar el flujo del programa y resolver los procesos planteados.	El uso de estructuras de control es limitado o incorrecto, generando errores o comportamientos inesperados.	
1.2	Manejo de cadenas y métodos (split y validaciones)	1	Implementa correctamente los métodos y realiza validaciones adecuadas sobre los datos extraídos.	El método split() no se usa correctamente o presenta errores en la separación y validación de los datos.	

1.3	Organización y legibilidad del código	1	El código está bien estructurado, con indentación, comentarios y nombres de variables apropiados.	El código carece de estructura o claridad, dificultando su lectura y mantenimiento.	
1.4	Entrega & Documentación (repo, manuales, diagramas)	1	Entrega completa del repositorio con manual técnico, manual de usuario y diagrama de flujo en el formato solicitado.	La entrega carece de alguno de los documentos requeridos o no sigue el formato establecido.	
1.5	Interactividad del menú (funcionalidad básica)	1	El menú permite al usuario navegar entre las opciones principales y ejecutar correctamente las acciones del sistema.	El menú presenta errores o no permite la ejecución fluida de las funcionalidades.	

1.6	Pruebas y ejecución correcta	1	El programa compila y ejecuta correctamente, mostrando resultados coherentes con los archivos de entrada.	El programa no compila o genera resultados incorrectos al ejecutar las pruebas.	
Sub-total Habilidades		**6 ptos**			

No.	Criterio de evaluación	Punteo Máximo	Satisfactorio (100%–61%)	Necesita mejorar (60%–0%)	Punteo Obtenido
2.1	Lectura y procesamiento de archivos (Inventario.lfp, Movimiento.lfp)	3	El sistema lee y procesa correctamente ambos archivos, separando los datos.	El sistema no procesa correctamente los archivos o presenta errores en la lectura y separación de datos.	

2.2	Actualización de inventario (compras/ventas)	3	El sistema actualiza correctamente el stock, ingresos y egresos según los movimientos de compra y venta registrados.	Las operaciones de actualización presentan errores lógicos o no modifican correctamente el inventario.	
2.3	Generación de reportes en HTML	3	Los reportes en HTML muestran la información requerida (movimientos, balance, top productos) de forma ordenada y legible.	Los reportes presentan errores, información incompleta o formato incorrecto.	
Sub-total Conocimiento		**9 pts**			
Total General	**15pts**				