**Trabajo Final Algoritmia y Programación 2024-1**
**entrega #1**

**Problema de planeación del programa de Ingeniería Industrial**

**Integrantes:**

Royer Alexander Gallego
Tomas Edil Urango Ruiz
Vanessa Arbelaez Sepulveda

Visual Studio es un entorno de desarrollo integrado (IDE) popular para crear aplicaciones en diversos lenguajes de programación (Python). Desarrollado por Microsoft, ofrece una amplia gama de herramientas para desarrollar, depurar y probar software. Características principales: Editor de código, depurador, herramientas de prueba, integración con Git y amplia gama de extensiones para personalizar la experiencia.
Ventajas: Entorno completo y potente: Ideal para proyectos de software complejos.
Soporte para múltiples lenguajes: Permite trabajar con diversos lenguajes de programación.
Versiones gratuitas y de pago: Adaptable a diferentes necesidades y presupuestos.
En resumen, Visual Studio es una herramienta poderosa y versátil para desarrolladores de software.

**Requisitos Funcionales:**

Inicio de Sesión de Usuario: El sistema debe permitir a los usuarios iniciar sesión con su nombre de usuario y contraseña.

Creación de Nuevo Usuario: El sistema debe permitir a los administradores crear nuevas cuentas de usuario.

Gestión de Perfiles de Usuario: El sistema debe permitir a los usuarios actualizar su información de perfil.

Búsqueda y Filtrado de Datos: El sistema debe permitir a los usuarios buscar y filtrar datos según ciertos criterios.

Generación de Reportes: El sistema debe permitir a los usuarios generar informes basados en datos almacenados.

Notificaciones y Alertas: El sistema debe enviar notificaciones a los usuarios sobre eventos importantes.

**Requisitos No Funcionales**

Rendimiento: El sistema debe responder rápidamente a las solicitudes del usuario.

Seguridad:El sistema debe proteger los datos sensibles de los usuarios y mantener la integridad de la información.

Usabilidad: El sistema debe ser fácil de usar y comprender para los usuarios finales.

Fiabilidad: El sistema debe estar disponible y funcionando correctamente la mayor parte del tiempo.

Compatibilidad: El sistema debe ser compatible con diferentes dispositivos y navegadores web.

Mantenibilidad:El sistema debe ser fácil de mantener y actualizar sin interrumpir las operaciones.

Escalabilidad: El sistema debe poder manejar un aumento en el número de usuarios y datos sin perder rendimiento.

Integridad de Datos:

Descripción: El sistema debe garantizar que los datos almacenados sean precisos y consistentes.

**Descripción, mejoras y características de la versión Visual Studio Code 2018**

Rendimiento Mejorado: Se realizaron mejoras significativas en el rendimiento general del IDE, incluyendo tiempos de carga más rápidos para proyectos grandes y reducción en el uso de memoria.

Soporte para Desarrollo de Aplicaciones Web: Integración mejorada con ASP.NET y ASP.NET Core para el desarrollo de aplicaciones web.
Nuevas plantillas y herramientas para crear aplicaciones web modernas y escalables.

Mejoras en Xamarin: Visual Studio 2018 continuó mejorando el soporte para desarrollo de aplicaciones móviles con Xamarin.
Nuevas plantillas y herramientas para facilitar la creación de aplicaciones móviles multiplataforma.

Mejoras en la depuración: Se agregaron nuevas características de depuración, como la capacidad de adjuntar el depurador a procesos en ejecución de manera más fácil y rápida.
Mejoras en la visualización de datos y en el seguimiento del flujo de ejecución durante la depuración.

Soporte para Python y Datos: Visual Studio 2018 mejoró su soporte para Python, incluyendo nuevas herramientas y plantillas para desarrollo.

Integración mejorada con herramientas de análisis de datos como Pandas y NumPy.

Mejoras en Git y Control de Versiones: Integración más profunda con Git y otras herramientas de control de versiones.
Mejoras en la visualización de cambios, resolución de conflictos y gestión de ramas.

Nuevo Instalador: Se introdujo un nuevo instalador para Visual Studio 2018, que permite una instalación más personalizada y modular del IDE y sus componentes.

Mejoras en el Editor de Código: Actualizaciones en el editor de código con nuevas funcionalidades de autocompletado, refactorización y resaltado de sintaxis.

Soporte para .NET Core 2.1: Visual Studio 2018 agregó soporte completo para .NET Core 2.1, permitiendo a los desarrolladores crear aplicaciones más rápidas y eficientes.

Mejoras en la Experiencia de Colaboración: Se mejoró la capacidad de colaboración en equipo con nuevas herramientas para revisión de código, comentarios y seguimiento de tareas.

Herramientas para Azure: Se agregaron nuevas herramientas para facilitar el desarrollo y despliegue de aplicaciones en la plataforma de nube de Microsoft, Azure.

Soporte para C# 7.3 y Visual Basic 15.8: Actualizaciones en el soporte para las últimas versiones de los lenguajes de programación C# y Visual Basic.

Estas son solo algunas de las características y mejoras que se introdujeron en Visual Studio 2018. La versión 2018 continuó consolidando la posición de Visual Studio como una de las IDEs más completas y poderosas para el desarrollo de aplicaciones en una variedad de plataformas y tecnologías.

Con el presente trabajo se pretende dar a conocer un listado de 1000 estudiantes con sus respectivos datos y semestre en el que se encuentran, a continuación se realiza el análisis.

**Pasos**

1. Importar archivos de nombres y apellidos a Python, utilizando import csv, llamamos los archivos cargados previamente en la consola.
2. Para combinar los nombres y apellidos y asegurarse de que no se van a duplicar los datos utilizamos la función (def generar_nombres_unicos) también definimos la cantidad de nombres completos 1000.
3. Definir los estudiantes por semestre y cupo por aula, luego utilizamos la función (generar_estudiantes_por_semestre)
4. Se obtendrá una lista de 1000 estudiantes distribuidos proporcionalmente por semestre, teniendo en cuenta los porcentajes de estudiantes por semestre y el cupo por aula para cada semestre, así se asegura que la lista final tiene la cantidad correcta de estudiantes por semestre y no excede el límite de cupo por aulas.
5. Importar la tabla de asignatura con nivel, núcleo curricular y créditos

6. Para cada asignatura se asigna una función que garantiza un código único y esto se hace utilizando la función (generar_codigo_asignatura)
7. Para calcular las horas de trabajo docente (HTD) y horas de trabajo independiente (HTI) se hace con la siguiente función (calcular_horas).
8. Para cada semestre y asignatura, se distribuyen los estudiantes en grupos de acuerdo con el cupo por aula establecido en la tabla, se calcula el número total de estudiantes por asignatura para crear grupos según el cupo máximo por aula.

   Calcular el Número de Grupos: Para cada asignatura en cada semestre, se debe calcular el número de grupos necesarios basados en el cupo máximo por aula y la cantidad total de estudiantes para ese semestre. Se utiliza la siguiente fórmula:

   Número de Grupos = Cantidad de Estudiantes / Cupo Máximo por Aula

   Se utiliza la función (ceil) para redondear hacia arriba, asegurando tener al menos un grupo por asignatura.

   Crear Datos para Cada Grupo: Para cada grupo de la asignatura, se crean los datos necesarios:

   - Código de la Asignatura
   - Horas de Trabajo Docente
   - Horas de Trabajo Independiente
   - Número Total de Estudiantes
   - Código del Curso
   - Total de Cursos Asignados
   - Fecha de Creación

   Agregamos estos datos a la lista “cursos_por_asignatura”.

   Actualizamos la cantidad total de estudiantes para el siguiente grupo, restando la cantidad asignada al grupo actual.

   Una vez que se completa este bucle, la lista (cursos_por_asignatura) contendrá todos los datos necesarios para cada grupo de cada asignatura en cada semestre. Estos datos incluyen el código de la asignatura, horas de trabajo, número de estudiantes, etc., siguiendo las reglas establecidas. Se debe asegurar de que no haya estudiantes repetidos en los grupos. También, se asigna la cantidad correcta de estudiantes por grupo, no excediendo el cupo máximo por aula.

9. Se crean los archivos de lista de clase para cada asignatura. Se crea el DataFrame para la planeación, las listas de cursos por asignatura y se genera la planeación para cada semestre. Cada archivo contendrá la información de la asignatura, el código del curso, el número de estudiantes por grupo, y el código del grupo.
10. Se organizan los archivos en una estructura de carpetas siguiendo la ruta dada en las reglas. Los archivos de lista de clase se guardarán en formatos de Excel y CSV, con los nombres correspondientes a la información de la asignatura y el grupo usando las funciones (grupo_df.to_excel) y (grupo_df.to_csv)

**Plan Proyecto**

| Tiempo (Horas) | Actividades | Fecha |
|---|---|---|
| 1 | Reunión de todos los integrantes del equipo, revisión de la guía de entrega para el trabajo final y asignación de tareas. | 03/04/2024 |
| 3 | Creación el código para el listado de los 1000 estudiantes | 05/04/2024 |
| 3 | Distribuir los estudiantes por semestre y cupo por aula al igual que el grupo de asignaturas. Calcular las horas de trabajo docente y horas de trabajo independiente de cada curso | 12/04/2024 |
| 2 | Elaboración del documento para la entrega 1 (Documento de visión y paso a paso para la generación del código) | 14/04/2024 |
| 1 | Revisión y entrega #1 del trabajo final en el tiempo definido por el docente | 16/04/2024 |
| **0.5** | **Sustentación de la entrega 1** | **17/04/2024** |
| 6 | Desarrollo del algoritmo y creación de la cuenta en GitHub | 30/05/2024 |
| 4 | Pruebas del código y ajustes necesarios | 31/05/2024 |
| 1 | Elaboración del plan de proyecto con sus respectivas actividades y tiempo empleado | 31/05/2024 |
| 2 | Creación de los datasets y las carpetas con sus respectivos archivos de xlsx y csv | 02/06/2024 |
| 2 | Subir documentación final a GitHub (datasets y algoritmos de programación) | 03/06/2024 |

| Tiempo (Horas) | Actividades | Fecha |
|---|---|---|
| 1 | Reunión de todos los integrantes del equipo, revisión de la guía de entrega para el trabajo final y asignación de tareas. | 03/04/2024 |
| 3 | Creación el código para el listado de los 1000 estudiantes | 05/04/2024 |
| 3 | Distribuir los estudiantes por semestre y cupo por aula al igual que el grupo de asignaturas. Calcular las horas de trabajo docente y horas de trabajo independiente de cada curso | 12/04/2024 |
| 2 | Elaboración del documento para la entrega 1 (Documento de visión y paso a paso para la generación del código) | 14/04/2024 |
| 1 | Revisión y entrega #1 del trabajo final en el tiempo definido por el docente | 16/04/2024 |
| 1 | Entrega del proyecto final en el tiempo definido por el docente | 04/06/2024 |
| **0.5** | **Sustentación de la entrega 2 del trabajo final** | **05/06/2024**<br>**07/06/2024** |
| | | |
| | | |

**El equipo está conformado por tres integrantes en total empleamos 37 horas de trabajo, la mayoría del tiempo trabajamos en el entorno de google colab con un documento compartido donde todos aportamos ideas.**

**Segunda entrega:**

https://github.com/TomasUR95/Trabajo-Final-Algoritmia-y-Programaci-n