Universidad Autónoma de Yucatán

Facultad de Matemáticas

Licenciatura en Ingeniería de Software

Programación Estructurada

Avances de Proyecto

Profesor: Dr. Emilió Gabriel Rejón Herrera

Equipo: Los Vectores

Fecha de entrega: 19 de junio de 2021

Mérida, Yucatán, México

Definición inicial de requerimientos del sistema.

Antecedentes de la propuesta.

Dado que el profesorado, en todo momento, ha realizado la gestión de calificaciones parciales o finales, de manera manual, ocupando el uso fundamental de un cuaderno, ha surgido la necesidad de desarrollar programas para automatizar dicha tarea, que en algunas ocasiones puede tornarse tediosa (Flor, 2018, párr. 3).

Las ventajas que ofrecen este tipo de programas son variadas, a manera de ejemplo, se tienen las siguientes: *Mayor seguridad*, ya que los datos son almacenados en una nube, la cual dispone de una copia de seguridad; *Automatización en el cálculo de las notas*, pues una vez que el usuario ingrese los datos, el programa obtiene el promedio, sin la necesidad de realizarlo de manera manual; *Mayor objetividad a la hora de calificar*, ya que, si el profesor no es el único que gestionará las calificaciones, estas últimas pasarán en manos de varias personas, antes de ser emitidas de manera oficial, por lo que la objetividad se ve en un considerable aumento; entre otras muchas ventajas (Aula 1, s. f.).

Hablando de los software profesionales cuyo objetivo es el de cubrir esta necesidad, hay tantos, para todo tipo de personalidad, adaptables al gusto de quién les dará el uso para el que fueron creados; un ejemplo es el denominado *Alma*, la cual es una aplicación que se puede configurar para adaptarla de acuerdo con las exigencias de la institución usuario (Alma, s. f., párr. 1), como se mencionó con anterioridad. Esta aplicación posee las características que a continuación serán enlistadas:

- Goza de una interfaz intuitiva.
- Las rúbricas que evalúan las actividades de los alumnos son personalizables.
- Las boletas finales que se emiten al final del curso son, de igual manera, personalizables.
- Los estudiantes son capaces de contactar con sus maestros en un mismo lugar.
- Los padres de los alumnos se mantienen informados con reportes individuales, generados de manera autóctona.
- Por lo anterior, el tiempo es aprovechado al máximo por parte de los usuarios.

Así como la aplicación *Alma*, existen muchas otras, pues es importante, para las instituciones de educación públicas / privadas, el proporcionar un servicio de excelencia, de calidad, para concluir de la mejor manera con la enseñanza de los futuros profesionistas.

Descripción del producto software.

El programa a desarrollar, fue denominado *Cómprame, ¡Te ahorraré el trabajo!*, y es un programa que almacena determinados datos en archivos con extensión .txt. Le permitirá al usuario guardar información relacionada con promedios finales, por asignatura, o por el año ya concluido, así como los nombres de los acreedores de dicho promedio, tanto de alumnos como de las instituciones procedentes. La información anterior fungirá como una ayuda espléndida para determinar los promedios individuales (como se dijo anteriormente), grupales, con la dicha de obtener a aquellos alumnos, mismos que fueron los mejores por grado, y por generación, para la otorgación pertinente del diploma al merecedor.

De igual manera, se contará con la presentación de un ranking, tal que se dispondrá de manera global de entre todas las secundarias (que sean introducidas por el usuario) del estado, con la finalidad de reconocer a aquellos que se coronen como los mejores del estado de Yucatán. Ya que la información es archivada, el usuario podrá acceder a la información proporcionada cada vez que se ingrese, así como efectuar modificaciones a la misma, de manera subsecuente.

Objetivos general y específicos del sistema.

Objetivo general:

Llevar un control de calificaciones por: escuela, grado, grupo, alumno, y asignatura.

Objetivos específicos:

1.- Analizar el desempeño general de un alumno.

1.1.- Determinar e imprimir el nivel de desempeño general de un alumno.

2.- Analizar el desempeño general de un grupo en particular.

2.1.- Determinar e imprimir el nivel de desempeño de un grupo en particular.

2.2.- Determinar el promedio más alto de un grupo en particular (uso posterior).

3.- Analizar el desempeño general por cada grado escolar.

3.1.- Ordenar de mayor a menor los promedios más altos de cada grupo por cada grado escolar.

3.2.- Imprimir a los primeros 3 promedios más altos por cada grupo.

4.- Analizar el desempeño general de todas la escuelas.

4.1.- Determinar e imprimir el nivel de desempeño general de todas la escuelas.

4.2.- Determinar e imprimir un *leaderboard* que disponga los 10 mejores promedios del estado de Yucatán.

5.- Imprimir información en consola de manera intuitiva.

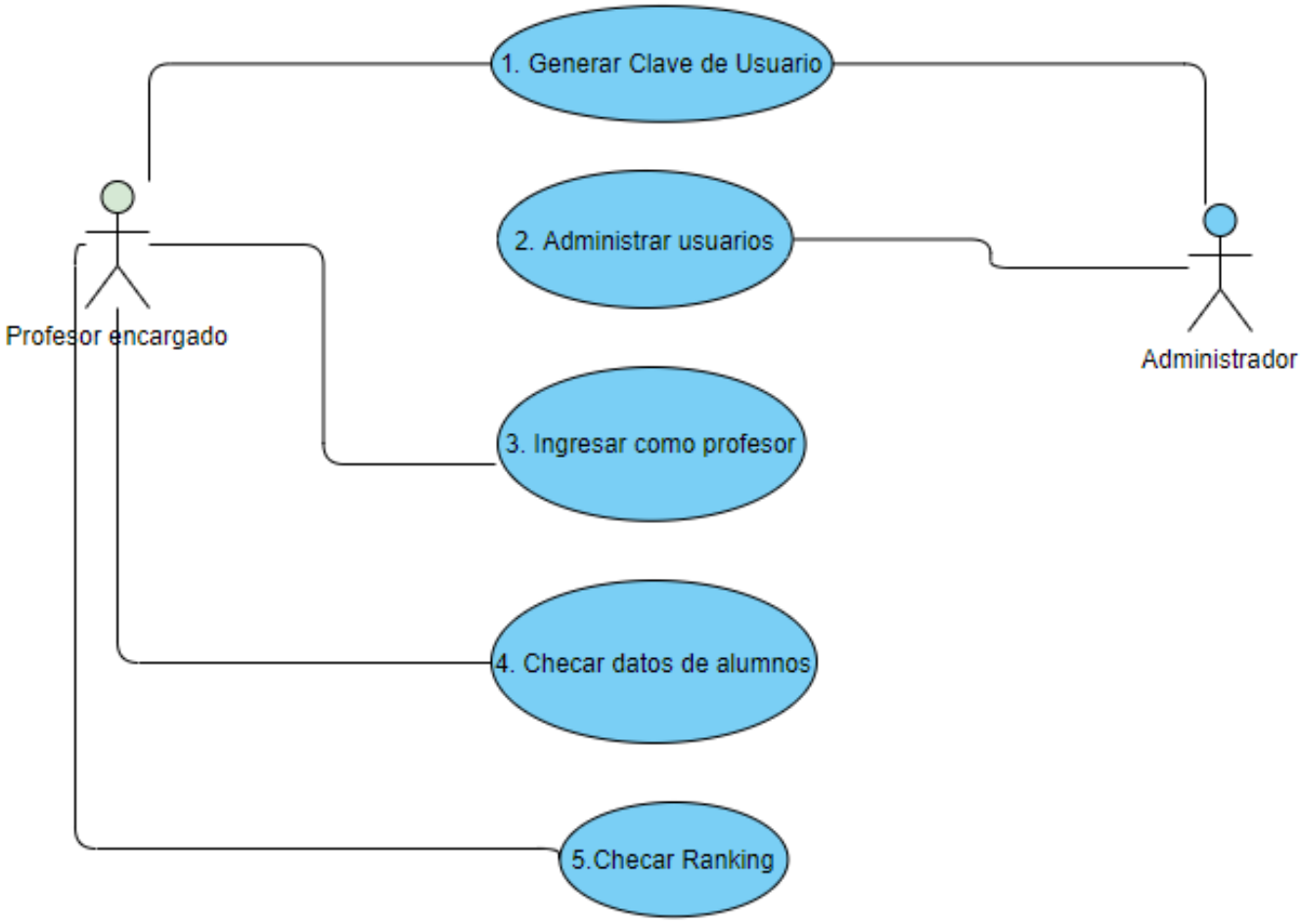
5.1.- Mostrar en pantalla información y opciones de una manera simple y clara.

6.- Acceder a opciones mediante una ID

6.1.- Determinar si el usuario que ingresa tiene una ID y contraseña válida.

6.2.- Permitir el deslogueo del usuario.

Generar un diagrama de casos de uso, para representar la funcionalidad general del sistema.



Hacer una distinción clara entre los diferentes actores del sistema. (descripción de tipos de usuarios).

Cliente:

Las entidades educativas estatales que se ingresen en comprar el programa, con una visión a futuro de expandirlo a entidades educativas nacionales.

Tipos de usuarios:

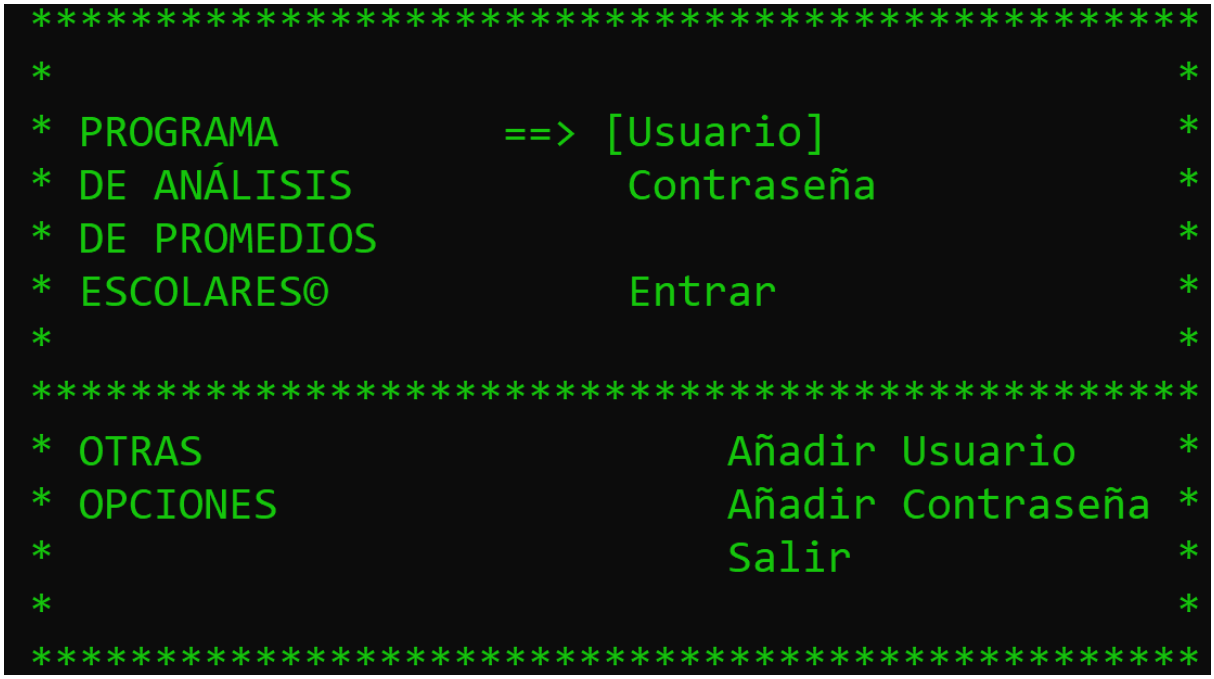
Profesor a Cargo:

Este profesor será seleccionado por la escuela con el fin de administrar el uso de la aplicación, entregando sus respectivas claves a los maestros, con las cuales podrán ingresar y modificar las calificaciones de las materias que la clave permita.

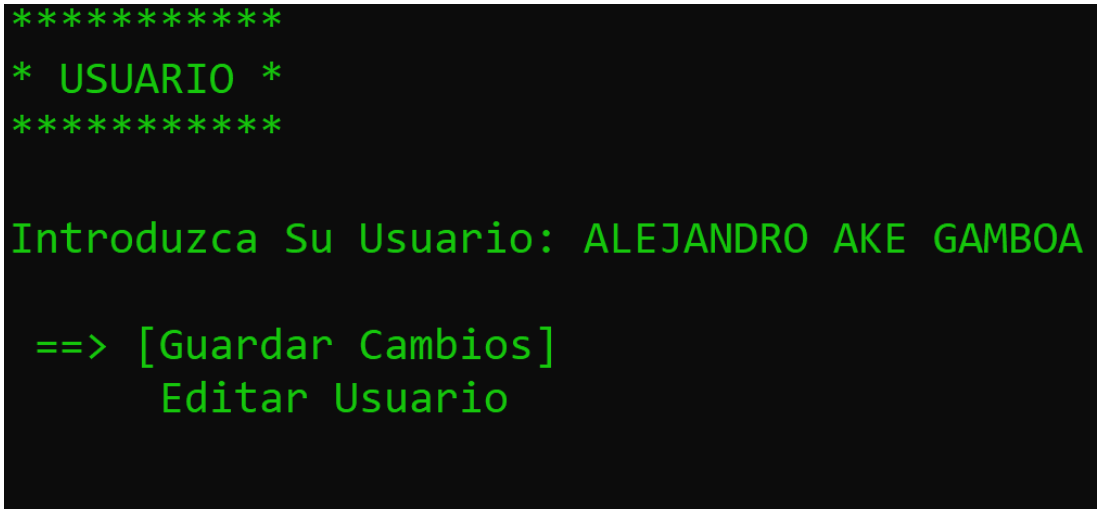
Profesores:

Los profesores serán los encargados de agregar y modificar las calificaciones de los alumnos, para poder modificar los datos se les pedirá su clave exclusiva y su usuario los cuales se le entregaran a cada profesor, lo cual le dará acceso a modificar las calificaciones de las materias que ese profesor imparte, al igual que revisar una lista de las calificaciones ya ingresadas junto con un ranking con las 3 mejores del grupo.

Elaborar las principales interfaces de usuario.

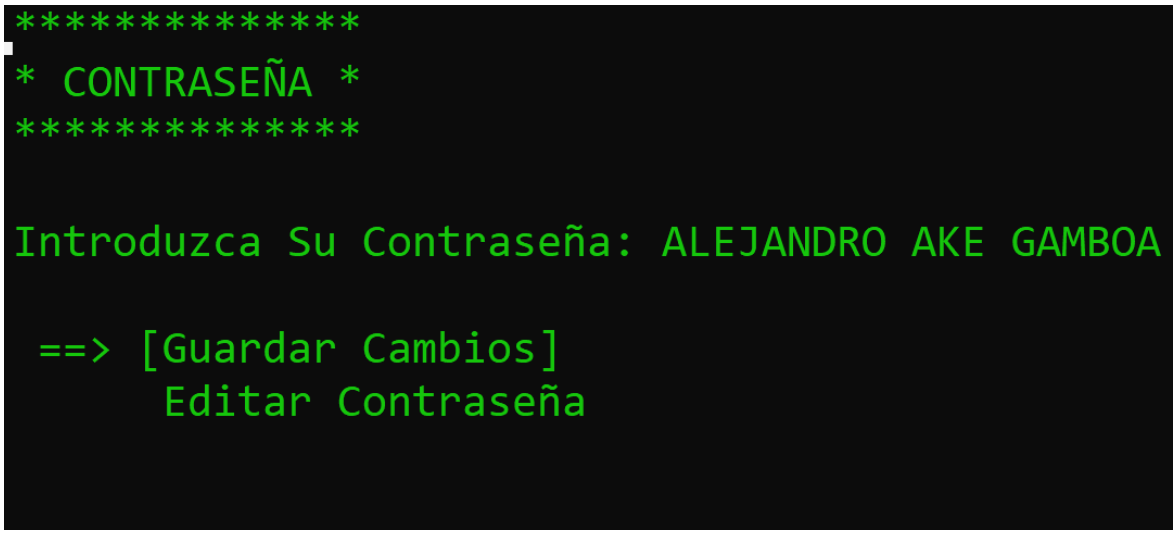


Interfaz de usuario 1. La presente interfaz será la que el usuario observará al inicializar el programa; en ella, se despliega, a la izquierda, el nombre del programa, su nombre formalizado. A la derecha, se observan 6 posibles opciones; Usuario: Le permite al usuario ingresar su usuario actual; Contraseña: Le permite al usuario ingresar su contraseña actual; Entrar: Una vez ingresados el usuario y contraseña correctos, se procede a imprimir el siguiente menú de opciones, en caso de que el usuario o la contraseña sean incorrectos, no se procederá al siguiente menú de opciones; Añadir Usuario: En esta opción le permite al usuario implementar un nuevo usuario, actualizando al anterior con el actual, recién ingresado; Añadir Contraseña: Esta opción le solicita al usuario la nueva contraseña que desea utilizar, actualizando así a la anterior; Salir: Se procede a finalizar el programa.

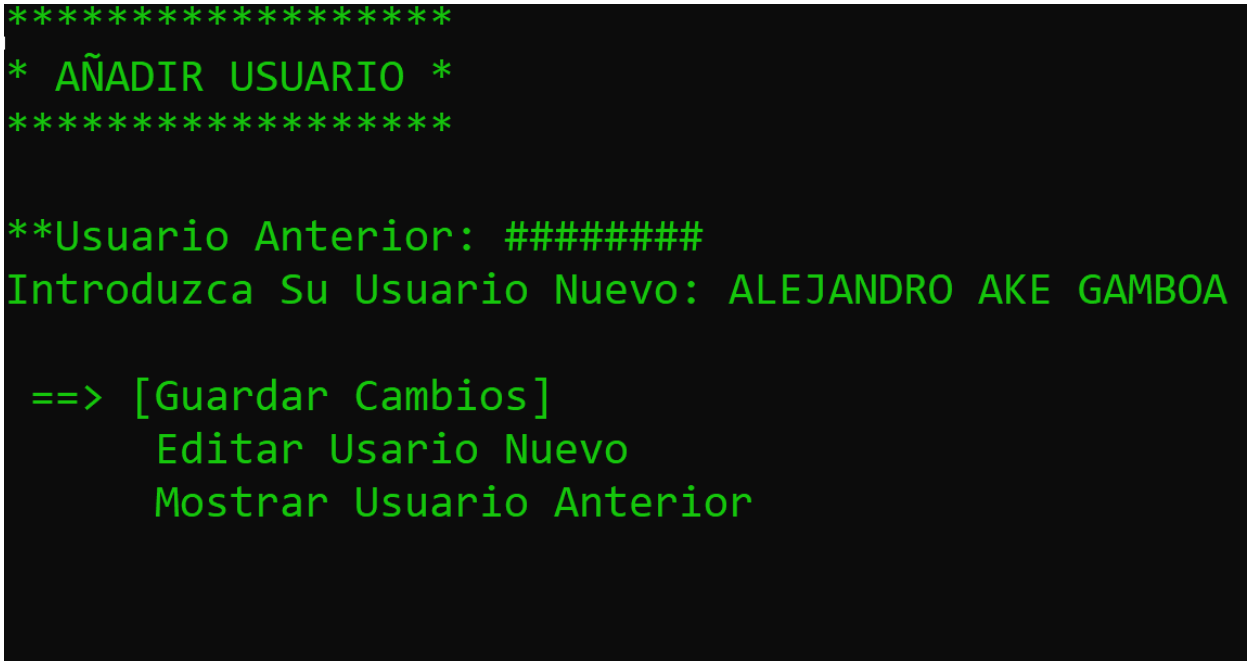


Interfaz de usuario 1.1. Una vez seleccionado en el menú principal de opciones, la opción Usuario, se procede a imprimir la siguiente interfaz, en la que se le solicita al usuario que ingrese su usuario, valga la redundancia; luego, se le presentan

dos opciones; Guardar Cambios: Se guarda el usuario tecleado; Editar Usuario: Se vuelve a solicitar el usuario, para que el usuario lo vuelva a teclear.

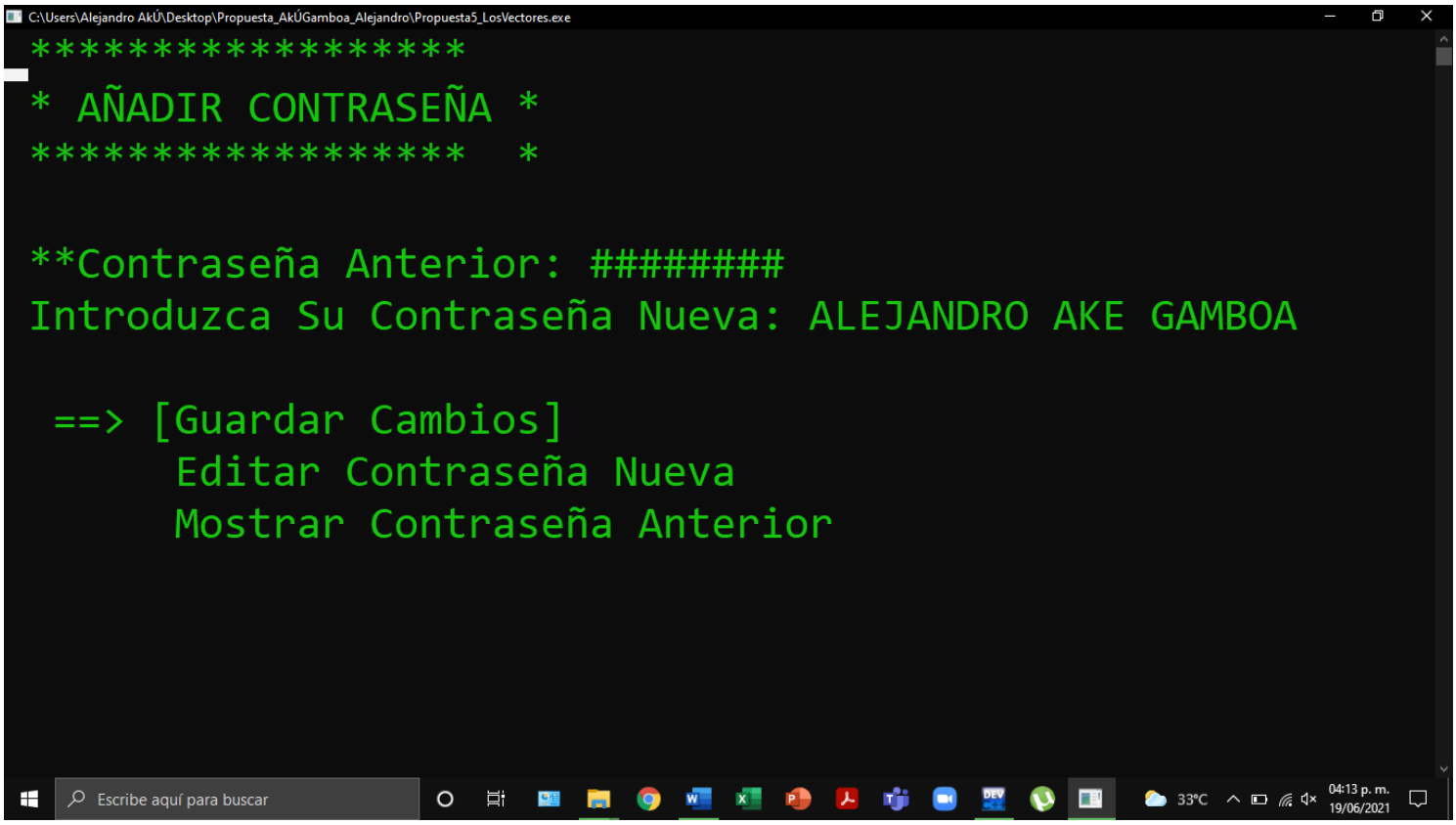


Interfaz de usuario 1.2. Si la opción seleccionada en el menú de opciones primario es la opción Contraseña, se procede a solicitar la contraseña, en el apartado Introduzca Su Contraseña: El usuario podrá teclear su contraseña, a la vez que la va visualizando en pantalla; al darle enter, se despliegan dos opciones, mismas que el usuario podrá elegir de acuerdo con sus necesidades. Guardar Cambios: Como en el caso anterior, se guarda la contraseña tecleada; Editar Contraseña: Se limpia la pantalla y vuelve a desplegarse la opción de Introduzca Su Contraseña, para que el usuario vuelva a teclear su contraseña, en caso de que se haya equivocado en la primera captura.

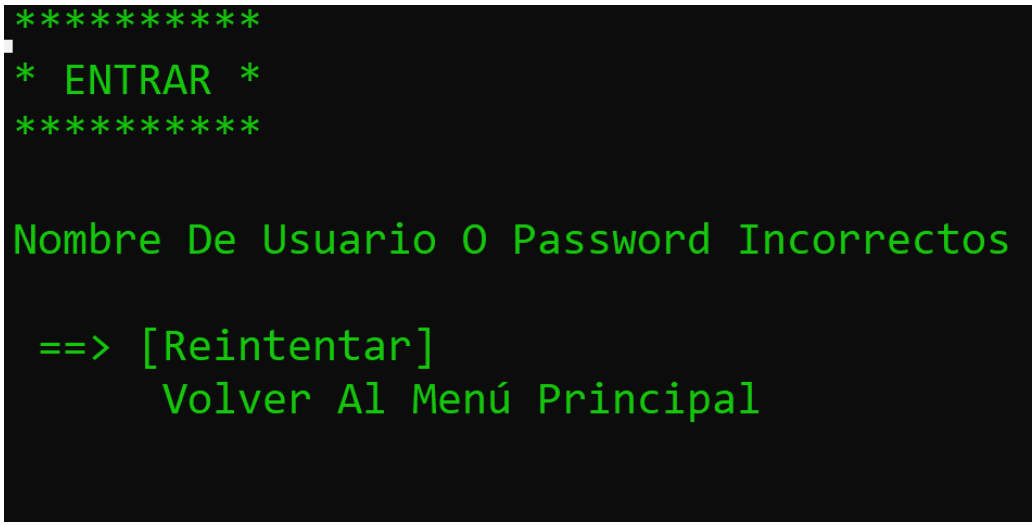


Interfaz de usuario 1.3. Si la opción seleccionada en el menú de opciones primario es Añadir Usuario, entonces se limpia la pantalla para presentar la siguiente interfaz, misma en la que se le solicita al usuario que Introduzca Su Usuario Nuevo: Una vez lo introduzca y presione la tecla ENTER, se procederá a desplegar las siguientes 3 opciones; Guardar Cambios:

Guarda el nuevo usuario tecleado, y retrocede al menú primario; Editar Usuario Nuevo: Se limpia la pantalla y se vuelve a presentar el mismo apartado, con la finalidad de que el usuario vuelva a ingresar su nuevo usuario; Mostrar Usuario Anterior: Se procede a imprimir el usuario anterior próximo de la captura de datos, en el apartado **Usuario Anterior: ###.

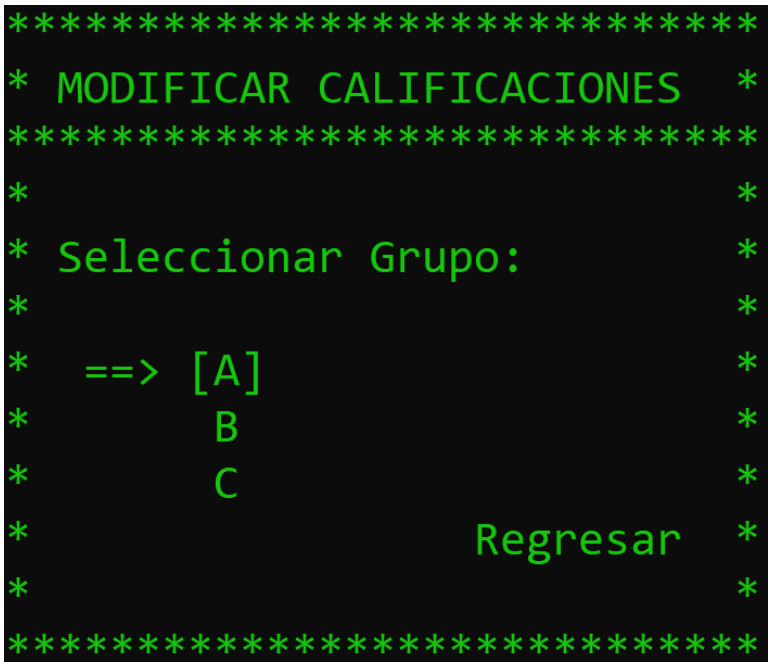


Interfaz de usuario 1.4. Si la opción seleccionada en el menú de opciones primario es Añadir Contraseña, entonces se limpia la pantalla para presentar la siguiente interfaz, misma en la que se le solicita al usuario que Introduzca Su Contraseña Nueva: Una vez lo introduzca y presione la tecla ENTER, se procederá a desplegar las siguientes 3 opciones; Guardar Cambios: Guarda la contraseña nueva tecleada, y retrocede al menú primario; Editar Contraseña Nueva: Se limpia la pantalla y se vuelve a presentar el mismo apartado, con la finalidad de que el usuario vuelva a ingresar su contraseña nueva; Mostrar Contraseña Anterior: Se procede a imprimir la contraseña anterior próximo de la captura de datos, en el apartado **Contraseña Anterior: ###.

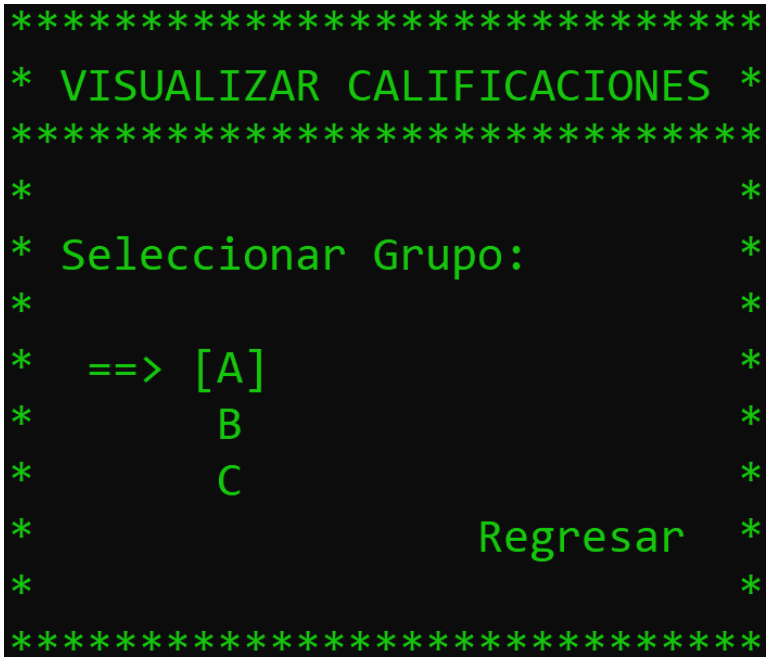


Interfaz de usuario 1.5. Si la opción seleccionada en el menú de opciones primario es Entrar, entonces se limpia la pantalla para presentar la siguiente interfaz, misma en la que se imprime que el Nombre De Usuario O Passowrd son Incorrectos, de ellos, se presentan dos posibles opciones; Reintentar: Si se selecciona esta opción, se vuelven a leer y evaluar el usuario y la contraseña, aunque realmente se puso dicha opción para darle una mejor presentación al programa; Volver Al Menú Principal: Se limpia la pantalla y se regresa al menú primario de opciones.

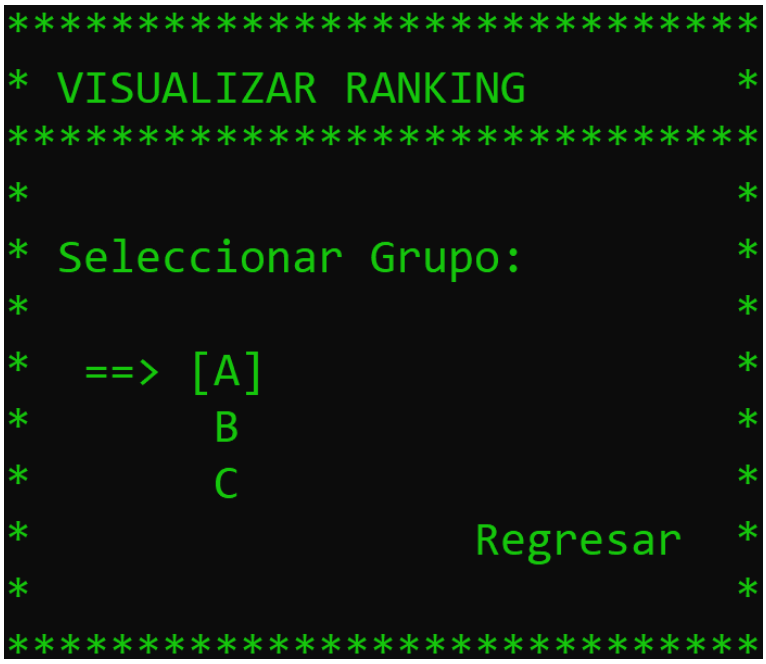
Interfaz de usuario 2. Una vez seleccionada la opción Entrar en el menú de opciones primario, y se valida que el usuario y contraseña, ambos son correctos, se procede a limpiar la pantalla e imprimir el siguiente menú de opciones secundario, en el que se despliegan 4 posibles opciones, la primera, denominada Modificar Calificaciones, la segunda, denominada Visualizar Calificaciones, la tercera, Visualizar Ranking, la cuarta y última, Cerrar Sesión: Esta, la última opción, como su nombre lo indica, finaliza el menú de opciones secundario, para redirigir al menú de opciones primario.



Interfaz de usuario 2.1. Una vez seleccionada la opción Modificar Calificaciones del menú secundario, se limpia la pantalla, y se procede a imprimir la siguiente pantalla, para que el usuario pueda seleccionar el grupo al cuál le desea modificar las calificaciones. Se tienen tres opciones: A, B, C, Regresar. En dado caso de seleccionar la opción Regresar, se limpia la pantalla y se redirige al menú secundario.



Interfaz de usuario 2.2. Una vez seleccionada la opción Visualizar Calificaciones del menú secundario, se limpia la pantalla, y se procede a imprimir la siguiente pantalla, para que el usuario pueda seleccionar el grupo al cuál le desea visualizar las calificaciones. Se tienen tres opciones: A, B, C, Regresar. En dado caso de seleccionar la opción Regresar, se limpia la pantalla y se redirige al menú secundario.



Interfaz de usuario 2.3. Una vez seleccionada la opción Visualizar Ranking del menú secundario, se limpia la pantalla, y se procede a imprimir la siguiente pantalla, para que el usuario pueda seleccionar el grupo al cuál desea visualizar el ranking de los 3 mejores alumnos, de dicho grupo seleccionado. Se tienen tres opciones: A, B, C, Regresar. En dado caso de seleccionar la opción Regresar, se limpia la pantalla y se redirige al menú secundario.

Describir, en forma clara, los principales requerimientos funcionales y no funcionales del sistema.

Funcionales

Promédiate es un programa de análisis de promedios escolares, a nivel secundarias. Promédiate ofrece una interfaz que resulta ser intuitiva para aquel que se encuentre como administrador; en este caso, en particular, el administrador por defecto es la Secretaría de Educación Pública, por sus siglas, SEP. El usuario podrá acceder, de manera fácil y sencilla, a las funcionalidades permitidas por Promédiate, mediante el despliegue en consola de las posibles opciones, las cuales se centran de acuerdo con las necesidades replanteadas con anterioridad, previas a la definición de los requisitos del sistema. Estas necesidades se mencionan, a continuación: análisis del promedio general de todas las escuelas, en Mérida, Yucatán; análisis del promedio individual por escuela; análisis del promedio individual por grupo, de cada escuela que se haya ingresado mediante el teclado; análisis del promedio individual por alumno, de cada grupo, por escuela. Tal como se mencionó, los datos serán ingresados mediante una computadora, por teclado, en consola, y serán desplegados de dos maneras, que a continuación se mencionarán.

Mediante la computadora, caben dos posibilidades en la presentación de los datos en Promédiate; puede ser que los datos se muestren en pantalla, en consola, o, por otra parte, puede que los datos que el usuario considere listos para su procesamiento posterior se guarden en archivos de extensión .txt. El usuario tiene la oportunidad de decidir, y no importa si el usuario desea sobre editar un archivo .txt, Promédiate cuenta con la opción de reingresar datos nuevos en archivos de esa extensión, aunque se encuentren ya almacenados. En el proceso de almacenamiento en el disco duro, se cuenta con la posibilidad de crear múltiples carpetas, para el almacenamiento de los archivos, pues estos necesitan un lugar en específico en dónde se alojarán; por lo que se consideró que el usuario decida, de acuerdo con lo que necesite, dónde almacenar los archivos. ¡Gracias, Promédiate!

En la interfaz, se presentan 6 posibles opciones, mismas que el usuario tendrá la oportunidad de elegir sin ningún compromiso; mediante un apuntador (⟹), y mediante las teclas **w**, **s**, **ENTER**, el usuario podrá observar para dónde va dicho apuntador, además de que la opción apuntada se encontrará encerrada entre corchetes **[ ]**, con la finalidad de que el usuario aprecie de manera pertinente la opción en la que se encuentra en el preciso instante. La entrada, como fue mencionada, será mediante las teclas ya mencionadas, y, valga la redundancia, será mediante el teclado físico de la computadora, la cual registrará los botones presionados. Una vez que se seleccione la opción requerida, por parte del usuario, se limpiará la pantalla para imprimir la interfaz de la opción seleccionada, visible para el usuario mediante la consola.

PromédiaT dispone de la capacidad de leer los movimientos del usuario mediante el teclado, opción que facilita la captura de datos por parte del usuario, con la finalidad de maximizar el tiempo de éste, en vez de estar leyendo el número de opción, acción que en ocasiones se puede convertir en una tarea laborosa para aquel que se encuentre utilizando el programa, y mucho más cuando en el menú de opciones, dispuestas en el programa en cuestión, se presentan varias posibilidades, todas con sus respectivos números de opción. Por ello, PromédiaT es la mejor opción a lo referente a programas que capturan calificaciones, que hacen ciertos labores escolares, y que sirven para maximizar las cuentas académicas de los profesores.

No funcionales

Promédiate ha sido diseñado y será implementado para que solamente un usuario lo administré, en un momento en específico. Ya que el código fuente de Promédiate no goza de una amplia complejidad, cabe la posibilidad de que, mediante la documentación indexa en el mismo código, el usuario, o sea, la SEP, pueda realizarle cambios a futuro, sean para actualizar las demandas concurrentes. Se contará también con un manual bien documentado, que constará de 5 páginas, aproximadamente, en el que se encontrará la descripción completa de Promédiate. Al ser un programa no tan complejo, desplegará los datos que el usuario desee, en consola, o podrá almacenarlos en formato .txt, por ello, interactuará, de manera cotidiana, con el intérprete de comandos denominado Símbolo del sistema.

Ya que se contará con el uso de la directiva <math.h> la fiabilidad de los datos (sean estos los promedios finales o parciales individuales, grupales, generales) de Promédiate es alta; el código se implementará de acuerdo con las fórmulas de las medidas de tendencia central, que se consideren pertinentes. De ello se pretende mostrar en pantalla, dos decimales en los promedios, y como se mencionó, el propio usuario puede supra editar lo que considere necesario. El usuario, que sea administrador, contará con una contraseña, que podrá editar de acuerdo con sus necesidades.

La interfaz proporcionada por el sistema es fácil de entender, es intuitiva, pues se seleccionaron títulos a las opciones que sean conocidos, que, muy probablemente, el usuario ya los habrá visto y utilizado previo a la utilización de PromédiaT, por ello, se estima que el nivel de experiencia, por parte, de los usuarios, sea mínima. Teniendo en cuenta que todo tipo de profesor utilizará la aplicación en cuestión, se estimaron las interfaces de manera sencilla, rápidas de aprender, e incluso puede que no exista la necesidad de aprender por parte de usuarios que se encuentren familiarizados en la utilización de aplicaciones semejantes a Promédiate, tales como: SICEI (Sistema de Información y Control Escolar Institucional), Enlínea2 (Moodle), Microsoft Windows, entre otras muchas. La interfaz conecta a las funciones, incluidas en el código fuente, para agilizar las rutinas propuestas y así, presentarle al usuario la opción que escoja, de manera instantánea, para ahorrar y maximizar el tiempo de éste.

Como se mencionó en los requerimientos funcionales, el programa tendrá una interacción constantes con el teclado físico de la computadora, pues las teclas **w**, **s**, **ENTER**, serán de utilidad en el posicionamiento del apuntador, pues resulta mucho más factible cambiar de opciones mediante la utilización de dos teclas, a estar leyendo números de opciones, lo último podría resultar una tarea tediosa para el usuario (si este no cuenta con la experiencia suficiente en programas computacionales), pues este podría equivocarse cuando se lea la opción que desea. Ya que, en la selección de opciones, de acuerdo con lo solicitado por el usuario, no pueden haber fallas, porque el usuario decide qué realizar mediante un apuntador, se estima que las excepciones que el sistema manejará, son unas cuantas, muy fáciles de corregir, con base en los posibles errores que se le puedan presentar al usuario, tales errores como calcular un promedio negativo, o dividir el promedio entre cero, son errores detectables, contables, fáciles de corregir; en caso de que concurra otro error, de otro tipo, cabe la posibilidad de reiniciar el programa, para no recurrir a la opción de cerrarlo totalmente, y así, se borren los datos ya ingresados por el usuario. La opción de cerrar y finalizar totalmente al programa, será a lo último que se recurra, pues las posibilidades de que aquello logre ocurrir, se encuentran en el intervalo $0 \leq p \leq 10$, donde p es la posibilidad de que ocurra dicho evento, y los límites del intervalo, $0, 10$, se encuentran representados en base 100, o sea, se encuentran porcentuados con base al 100%.

Requerimientos Funcionales

RF1	Menú Principal
Descripción	El sistema debe solicitarle al usuario, en el menú principal, qué acción desea realizar
Observaciones	El sistema presentará 6 posibles opciones, en las que el usuario decidirá cuál tomar, sea para teclear su usuario y contraseña, añadir nuevo usuario y nueva contraseña, o para entrar al programa, para realizar ciertas acciones una vez se encuentre dentro.

RF2	Ingreso
Descripción	El sistema debe pedir al usuario que ingrese una ID y una contraseña válida para acceder a más acciones.
Observaciones	Esta acción la podrán hacer aquellos usuarios a los que se les haya asignado una ID y contraseña previamente, o incluso, el sistema puede solicitarle al usuario que cree un usuario y una contraseña nuevos, para que el usuario, si es nuevo y requiere el sistema en ese momento, no tenga la necesidad de recurrir a la institución de estudios. Entradas: Clave numérica, contraseña (ambos pueden ser números o letras, o ambos a su vez).

RF3	Editar datos de alumnos
Descripción	El sistema debe permitir al usuario modificar los datos ingresados previamente de los alumnos. Con datos de los alumnos se hace referencia a las calificaciones de estos.
Observaciones	Una vez ingresado el usuario correctamente, podrá edita información de alumnos con respecto a sus calificaciones de acuerdo con cada asignatura, que toma el alumno en ese instante, Entradas: Variables reales (calificaciones), cadenas de caracteres(nombres). Salidas: Variables reales(calificaciones), cadenas de caracteres(nombres).

RF4	Análisis de promedios individuales por grupo
Descripción	El sistema debe generar promedios de acuerdo a los datos ingresados de los alumnos.
Observaciones	Para que el programa pueda realizar este tipo de operación requiere que previamente se hayan ingresado datos para realizar los diferentes procesos. Entradas: Variables reales(calificaciones), cadenas de caracteres(nombres). Salidas: Variables reales(calificaciones), cadenas de caracteres(nombres).

RF5	Ranking
Descripción	El sistema debe generar rankings sobre los mejores 3 promedios, de acuerdo a las calificaciones de los alumnos y de acuerdo a los grupos.
Observaciones	Para que el programa pueda realizar este tipo de operación requiere que previamente se hayan ingresado datos para realizar los diferentes procesos. Se podrán visualizar el nombre del alumno y su calificación promedio y su posición de ranking Salidas: Variables reales(promedios), cadenas de caracteres(nombres), variables enteras (posición en el ranking).

RF6	Cerrar sesión
Descripción	El sistema permite desloguear al usuario si así lo desea
Observaciones	Se debe haber ingresado correctamente mediante un usurio y contraseña válidos anteriormente, pues es la única forma de que esta opción este visible.

RF7	Apuntador
Descripción	El sistema permite al usuario elegir opciones mediante un apuntador (⟹)
Observaciones	El usuario podrá observar para dónde va dicho apuntador, además de que la opción apuntada se encontrará encerrada entre corchetes **[ ].**

Requerimientos no funcionales

RNF1	Rapidez
Descripción	El sistema ofrece una interfaz intuitiva para que el usuario administrador acceda de manera fácil y sencilla a las funcionalidades permitidas por PromediaT.
Observaciones	Los datos mostrados en pantalla serán en consola Se desplegarán los datos que el usuario desee o podrá almacenarlos en formato .txt.

RNF2	Seguridad
Descripción	El sistema debe permitir el acceso en el 100% de los casos a aquellos usuarios que cuenten con una clave y contraseña válidos, de lo contrario, el sistema no procederá al inicio de la sesión, por lo que no podrán continuar aquellos usuarios malintencionados.
Observaciones	Se debe permitir el correcto acceso a aquellos usuarios que tengan una ID válida, en caso contrario no se podrá acceder a más acciones dentro del programa.

RNF3	Interacción con teclado
Descripción	El programa permite leer los movimientos del usuario mediante el teclado, opción que facilita la captura de datos por parte del usuario, con la finalidad de maximizar el tiempo de éste.
Observaciones	Dicha movilidad se dará con las teclas **w**(apuntador hacia arriba), **s** (apuntador hacia abajo), **ENTER**(eligir opción).

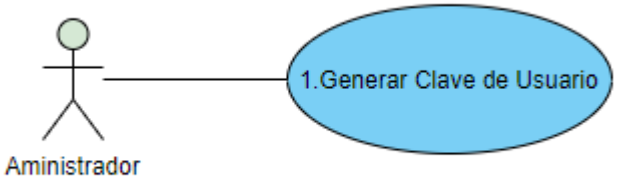
RNF4	Fiabilidad
Descripción	El sistema debe realizar de manera correcta las operaciones para obtener promedios (finales o parciales individuales, grupales, generales).
Observaciones	Se pretende mostrar en pantalla, dos decimales en los promedios, no obstante, el propio usuario puede supra editar lo que considere necesario.

Mapeo de Requerimientos

Requerimientos	Objetivos específicos					
	OE1	**OE2**	**OE3**	**OE4**	**OE5**	**OE6**
RF1: El sistema debe solicitarle al usuario, en el menú principal, qué acción desea realizar					X	
RF2: El sistema debe pedir al usuario que ingrese una ID y una contraseña válida para acceder a más acciones.						X
RF3: El sistema debe permitir al usuario modificar los datos ingresados previamente de los alumnos.			X			
RF4: El sistema debe generar promedios de acuerdo a los datos ingresados de los alumnos.	X					
RF5: El sistema debe generar rankings sobre los mejores 3 promedios, de acuerdo a las calificaciones de los alumnos y de acuerdo a los grupos.		X				
RF6: El sistema permite desloguear al usuario si así lo desea.						X
RF7: El sistema permite al usuario elegir opciones mediante un apuntador ($\Longrightarrow$).					X	
RNF1: El sistema ofrece una interfaz intuitiva para que el usuario administrador acceda de manera fácil y sencilla a las funcionalidades permitidas por PromediaT.					X	
RNF2: El sistema debe permitir el acceso en el 100% de los casos a aquellos usuarios que cuenten con una clave y contraseña válidos, de lo contrario, el sistema no procederá al inicio de la sesión, por lo que no podrán continuar aquellos usuarios malintencionados.						X
RNF3: El programa permite leer los movimientos del usuario mediante el teclado, opción que facilita la captura de datos por parte del usuario, con la finalidad de maximizar el tiempo de éste.					X	
RNF4: El sistema debe realizar de manera correcta las operaciones para obtener promedios (finales o parciales individuales, grupales, generales).				X		

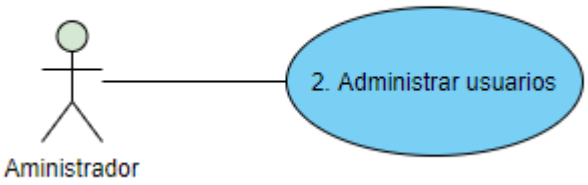
Elaborar los casos de uso para cada funcionalidad, donde se describa el proceso y las rutas de acción alternativas.

Caso de uso 1. Generar clave de usuario.



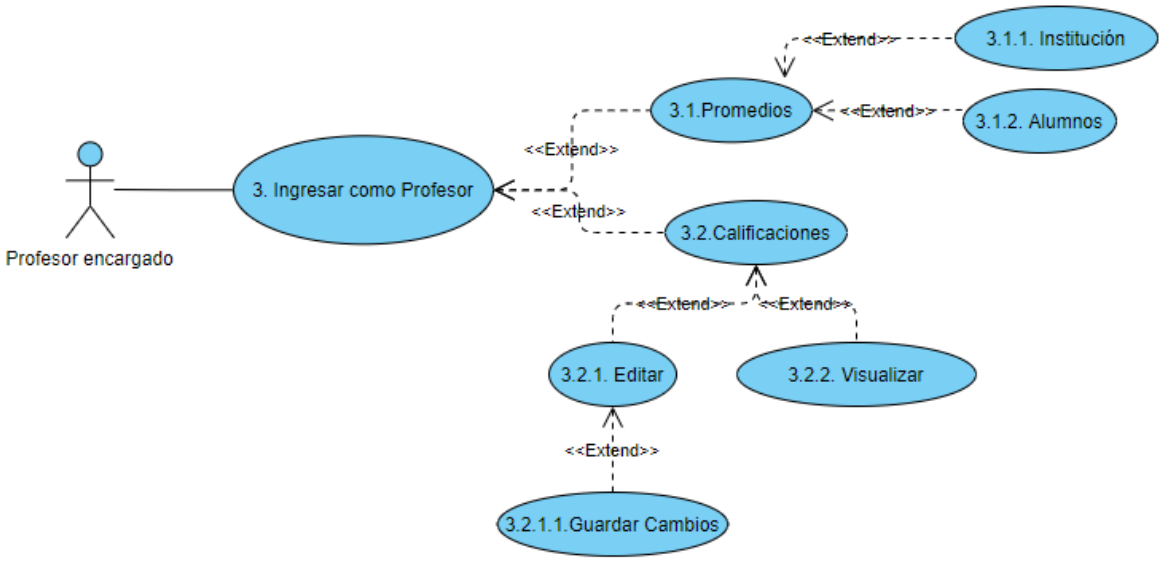
Actor	Administrador
Descripción	El administrador asigna un nombre de usuario y contraseña al profesor encargado (de acuerdo a la institución) para poder ingresar al sistema.
Precondiciones	El programa debe ser descargado en equipos de cómputo con sistemas operativos Windows 10, Linux o Mac. El usuario debe ejecutar el programa desde la carpeta donde haya decidido guardar el archivo. El administrador debe generar y proporcionar un nombre y contraseña correctos.
Postcondiciones	El profesor encargado se registra en el programa.
Subcaso	Ninguno

Caso de uso 2. Administrar usuarios



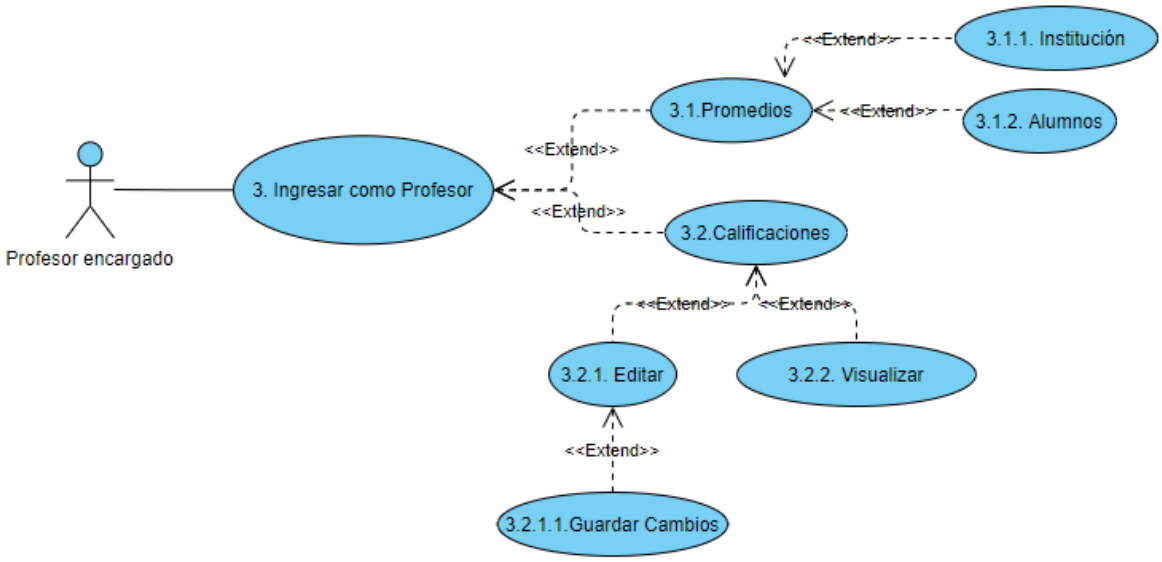
Actor	Administrador
Descripción	El administrador debe monitorear los usuarios de profesores que ingresan al sistema.
Precondiciones	Debe de haber al menos el usuario de un profesor
Postcondiciones	Es posible monitorear a los usuarios que ingresen al programa.
Subcaso	Ninguno.

Caso de uso 3. Ingresar como profesor.



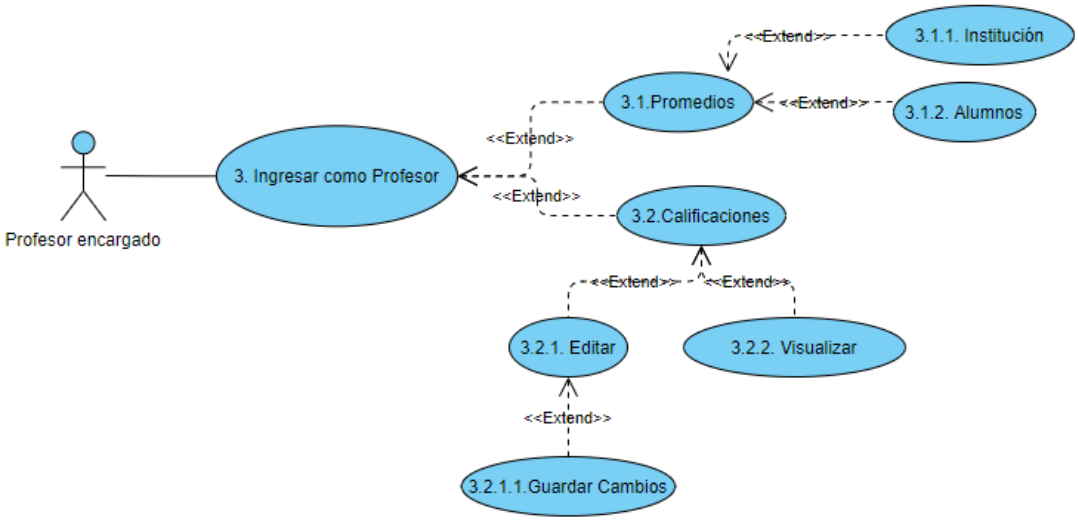
Actor	Profesor encargado.
Descripción	El usuario ingresará su nombre y contraseña para acceder a más acciones.
Precondiciones	El usuario debe estar registrado en el programa por el administrador
Postcondiciones	El usuario accede a más opciones correctamente sino hay problemas al loguear.
Subcaso	Promedios y Calificaciones

Caso de uso 3.1 Promedios



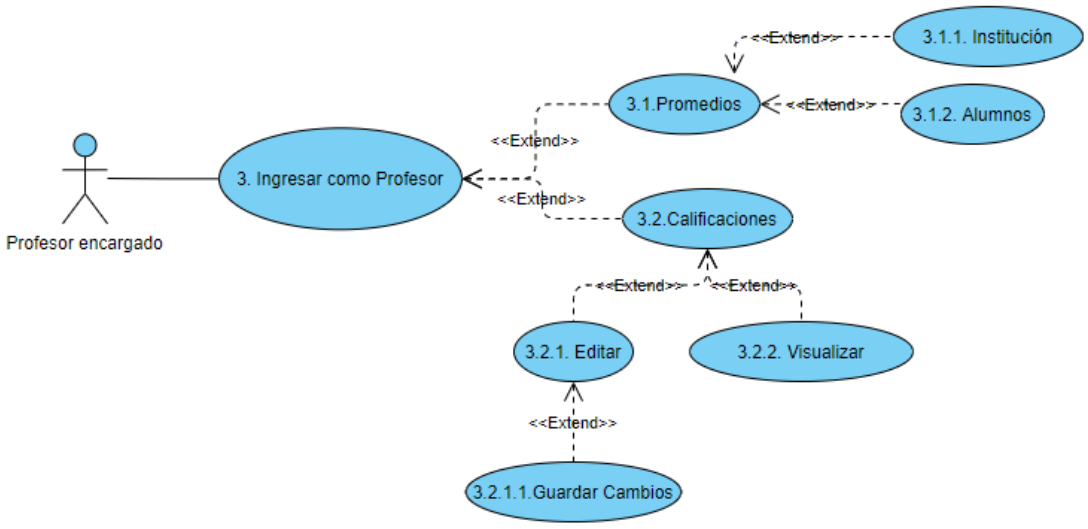
Actor	Profesor encargado.
Descripción	El usuario podrá acceder y obtener promedios teniendo como opciones: “Institución” y ”Alumnos”.
Precondiciones	Haberse logueado de manera exitosa. El usuario debe tener registrado datos que hayan sido guardados en archivos de tipo texto, para poder realizar análisis por parte del programa y visualizar información.
Postcondiciones	El usuario podrá visualizar y generar promedios de la institución y de los alumnos.
Subcaso	Institución y Alumnos.

Caso de uso 3.1.1 Institución



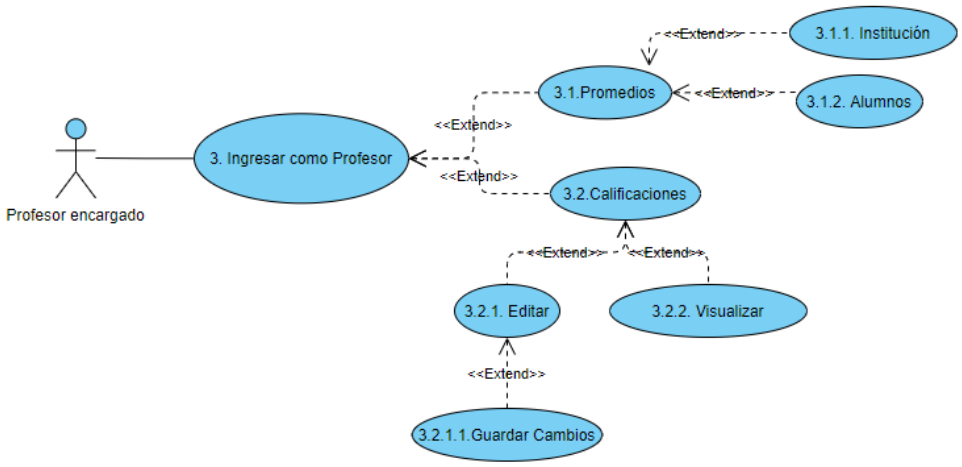
Actor	Profesor encargado.
Descripción	El usuario accede a los promedios generales de la institución de acuerdo a las materias, cursos y periodos (generado de acuerdo a los datos de los alumnos).
Precondiciones	Haberse logueado de manera exitosa. Debe haber información previamente almacenada para que el programa pueda hacer el proceso de generado de promedios.
Postcondiciones	Se visualizan los promedios de la Institución.
Subcaso	Ninguno.

Caso de uso 3.1.2. Alumnos



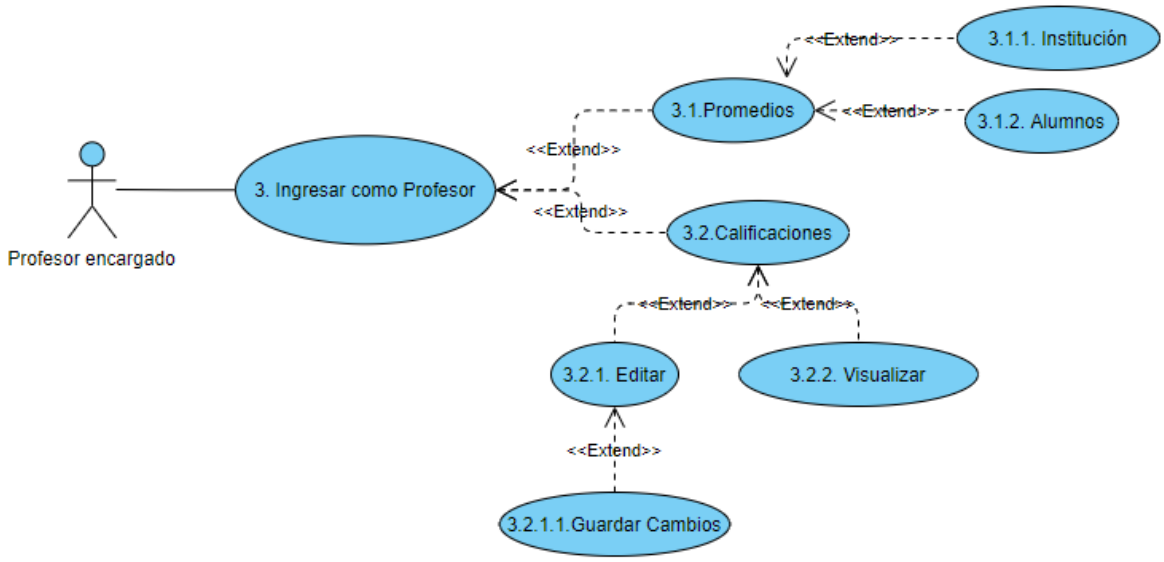
Actor	Profesor encargado.
Descripción	El usuario accede a los promedios generales de los Alumnos de acuerdo a las materias, cursos y periodos.
Precondiciones	Haberse logueado de manera exitosa. Debe haber información previamente almacenada para que el programa pueda hacer el proceso de generado de promedios de cada alumno.
Postcondiciones	Se visualizan y generan correctamente los promedios de los alumnos.
Subcaso	Ninguno.

Caso de uso 3.2. Calificaciones



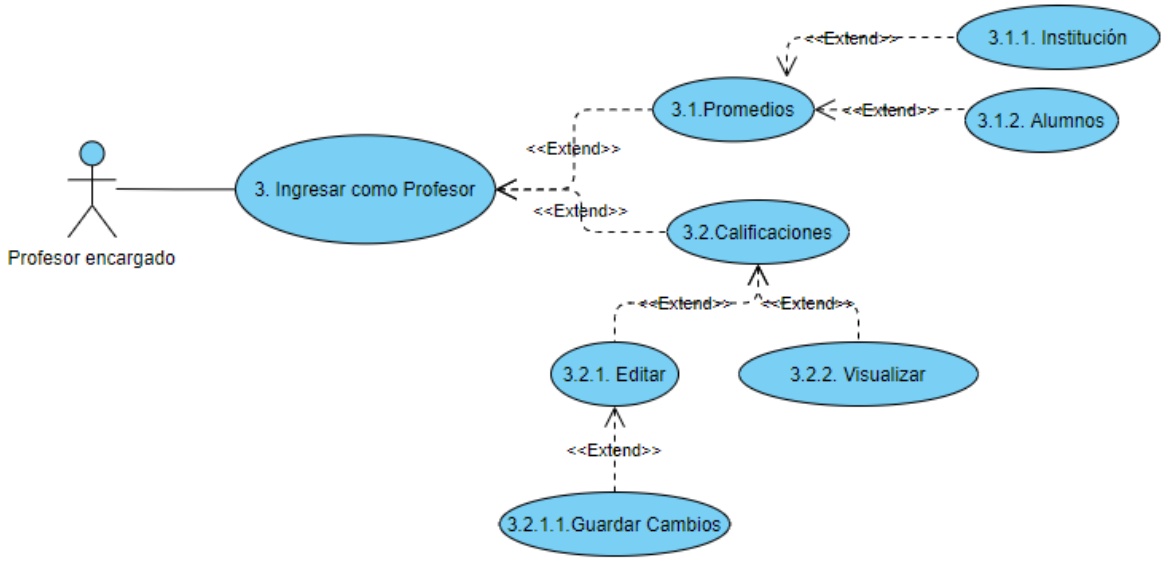
Actor	Profesor encargado.
Descripción	El usuario podrá acceder a las calificaciones de los alumnos para realizar acciones.
Precondiciones	Haberse logueado de manera exitosa.
Postcondiciones	Se accede a las calificaciones de los alumnos.
Subcaso	Editar y Visualizar.

Caso de uso 3.2.1 Editar



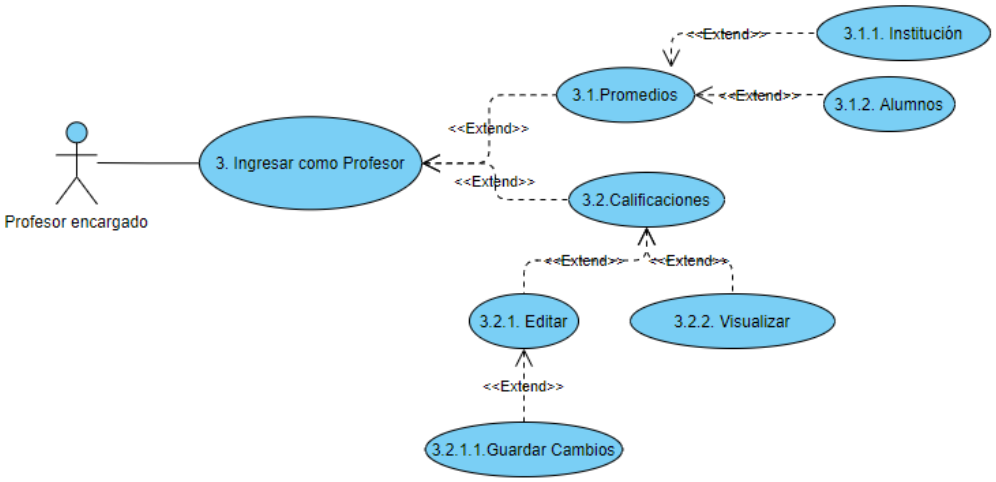
Actor	Profesor encargado.
Descripción	El usuario podra acceder al concetrado de alumnos para realizar modificaciones respecto a calificaciones de los alumnos.
Precondiciones	Haberse logueado de manera exitosa.
Postcondiciones	Podra agregar o eliminar el nombre de alumnos asi como sus respectivas calificaciones de acuerdo a su curso y materias.
Subcaso	Guardar cambios

Caso de uso 3.2.1.1 Guardar Cambios



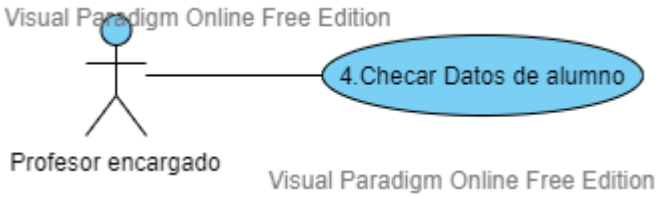
Actor	Profesor encargado.
Descripción	El usuario guardará los datos agregados o eliminados, un otra modificación una vez que este satisfecho con los cambios.
Precondiciones	Haberse logueado de manera exitosa.
Postcondiciones	Se almacenan de forma correcta los datos modificados.
Subcaso	Ninguno

Caso de uso 3.2.2 Visualizar



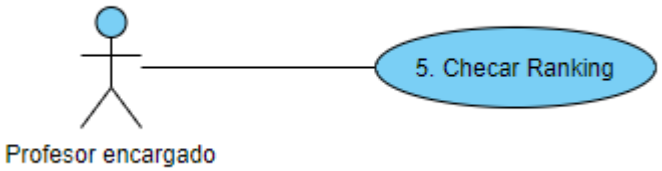
Actor	Profesor encargado.
Descripción	El usuario podra visualizar el concentrado de las calificaciones de los alumnos sin preocuparse de modificar algo por error.
Precondiciones	Haberse logueado de manera exitosa. Para visualizar una lista ha de haber datos registrados previamente.
Postcondiciones	Se visualizan las calificaciones sin problemas
Subcaso	Ninguno

Caso de uso 4. Checar datos de alumno



Actor	Profesor encargado
Descripción	El usuario podrá consultar datos rapidos que no son modificables sobre los alumnos, tales como nombre, calificaciones, materias y curso.
Precondiciones	Cargar datos de alumnos.
Postcondiciones	El programa deplega información de acuerdo al alumno buscado.
Subcaso	Ninguno

Caso de uso 5. Checar ranking



Actor	Profesor encargado
Descripción	El usuario podrá consultar los mejores 3 alumnos de cada curso de acuerdo a su institución. Ademas podrá consultar los 10 alumnos con promedios más altos del mismo nivel académico de todo el Estado de Yuctán.
Precondiciones	Cargar datos de alumnos.
Postcondiciones	El usuario consulta el ranking deseado correctamente.
Subcaso	Ninguno

Definición del estándar de codificación.

El estándar que utilizaremos ISO/IEC 9899:2017 del cual sacaremos la definición clara y específica sobre el estándar que se utilizará en el equipo de trabajo, este incluye uso de nombrado de variables, macros, funciones, archivos, bibliotecas, uso de comentarios, así como la información que contendrá cada bloque de comentarios.

Entornos de ejecución:

Se definen dos entornos de ejecución: autónomo y alojado. En ambos casos, el inicio del programa ocurre cuando una función C designada es llamada por el entorno de ejecución. Todos los objetos con una duración de almacenamiento estática se inicializarán (con sus valores iniciales) antes del inicio del programa. La forma y el momento de dicha inicialización no están especificados. La terminación del programa devuelve el control al entorno de ejecución.

- *Entorno independiente:*

  En un entorno independiente (en el que la ejecución del programa C puede tener lugar sin ningún beneficio de un sistema operativo), el nombre y el tipo de la función llamada en el inicio del programa están definidos. Cualquier biblioteca disponible para un programa independiente, aparte del conjunto mínimo requerido por la Cláusula 4, está definida por la implementación.
- *Entorno Alojado (IDE):*
  - *Inicio del programa:*

    La función llamada al inicio del programa se llama main. La implementación no declara ningún prototipo para esta función. Se definirá con un tipo de int de retorno y sin parámetros:

    ```
    int main(void) { /* ... */ }
    ```

    o con dos parámetros (referidos aquí como argc y argv, aunque se pueden usar nombres, ya que son locales a la función en la que se declaran):

    ```
    int main(int argc, char *argv[]) { /* ... */ }
    ```

    - El valor de argc no será negativo.
    - argv[argc] será un puntero nulo.
    - Si el valor de argc es mayor que cero, los miembros del array `argv[0]` a través de `argv[argc-1]` inclusive contendrán punteros a cadenas, a las que el entorno host les dará valores definidos por la implementación antes del inicio del programa. La intención es suministrar al programa información determinada antes del inicio del programa desde otro lugar del entorno alojado. Si el entorno host no es capaz de suministrar cadenas con letras tanto en mayúscula como en minúscula, la implementación garantizará que las cadenas se reciban en minúsculas.
    - Si el valor de argc es mayor que cero, la cadena apuntada por `argv[0]` representa el nombre del programa; `argv[0][0]` será el carácter nulo si el nombre del programa no está disponible en el entorno host. Si el valor de argc es mayor que uno, las cadenas apuntadas por `argv[1]` a través de `argv[argc-1]` representan los parámetros del programa.

- Los parámetros argc y argv y las cadenas apuntadas por el array argv serán modificables por el programa, y conservarán sus últimos valores almacenados entre el inicio y la terminación del programa.

Consideraciones ambientales:

- *Conjunto de caracteres:*
  - Tanto la fuente básica como los conjuntos de caracteres básicos de ejecución tendrán los siguientes miembros: las 26 letras mayúsculas del alfabeto latino

```
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z
```

  - las 26 letras minúsculas del alfabeto latino

```
a b c d e f g h i j k l m
n o p q r s t u v w x y z
```

  - los 10 dígitos decimales

```
0 1 2 3 4 5 6 7 8 9
```

  - los 29 caracteres gráficos siguientes

```
! " # % & ' ( ) * + , - . / :
; < = > ? [ \ ] ^ _ { | } ~
```

Palabras clave:

Syntax:

keyword: one of

auto	**extern**	**short**	**while**
break	**float**	**signed**	**_Alignas**
case	**for**	**sizeof**	**_Alignof**
char	**goto**	**static**	**_Atomic**
const	**if**	**struct**	**_Bool**
continue	**inline**	**switch**	**_Complex**
default	**int**	**typedef**	**_Generic**
do	**long**	**union**	**_Imaginary**
double	**register**	**unsigned**	**_Noreturn**
else	**restrict**	**void**	**_Static_assert**
enum	**return**	**volatile**	**_Thread_local**

Semántica:

Las fichas anteriores (sensibles a mayúsculas y minúsculas) están reservadas (en las fases de traducción 7 y 8) para su uso como palabras clave, y no se utilizarán de otro modo. La palabra `clave _Imaginary` está reservada para especificar imaginario.

Constante:

Syntax

constant:

integer-constant
floating-constant
enumeration-constant
character-constant

Limitaciones:

Cada constante tendrá un tipo y el valor de una constante estará en el rango de valores representativos de su tipo.

Descripcion:

Una constante entera comienza con un dígito, pero no tiene punto o parte exponente. Puede tener un prefijo que especifica su base y un sufijo que especifica su tipo.

Una constante flotante tiene una parte significante que puede ser seguida por una parte exponente y un sufijo que especifica su tipo. Los componentes de la parte significativa podrán incluir una secuencia de dígitos que represente la parte entera, seguida de un punto (.), seguida de una secuencia de dígitos que represente la parte de la fracción. Los componentes de la parte exponente son una e, E, p, o P seguidos por un exponente que consiste en una secuencia de dígitos firmada opcionalmente. Ya sea la parte del número entero o la parte de la fracción tiene que estar presente; para las constantes flotantes decimales, ya sea el período o la parte del exponente tiene que estar presente.

Declaración de funciones:

Limitaciones:

1. Un declarador de función no especificará un tipo de retorno que sea un tipo de función o un tipo de matriz.
2. Una lista identificadora en un declarador de funciones que no forme parte de una definición de esa función estará vacía.
3. Tras el ajuste, los parámetros de una lista de tipos de parámetros de un declarador de funciones que forme parte de una definición de esa función no deberán tener un tipo incompleto.

Semántica:

1. Una lista de tipos de parámetros especifica los tipos de los parámetros de la función y puede declarar identificadores para ellos.
2. La declaración de un parámetro como "tipo de retorno de función" se ajustará a "tipo de retorno de puntero a función".
3. El caso especial de un parámetro sin nombre de tipo void como el único elemento en la lista especifica que la función no tiene parámetros.
4. Si, en una declaración de parámetros, un identificador puede tratarse como un nombre de typedef o como un nombre de parámetro, se tomará como un nombre de typedef.
5. Si el declarador de función no forma parte de una definición de esa función, los parámetros pueden tener un tipo incompleto y utilizar la notación `[*]` en sus secuencias de especificadores de declarador para especificar tipos de array de longitud variable.

```
void addscalar(int n, int m,
      double a[n][n*m+300], double x);

int main()
{
      double b[4][308];
      addscalar(4, 2, b, 2.17);
      return 0;
}

void addscalar(int n, int m,
      double a[n][n*m+300], double x)
{
      for (int i = 0; i  <  n; i++)
            for (int j = 0, k = n*m+300; j  <  k; j++)
                  // a is a pointer to a VLA with n*m+300 elements
                  a[i][j] += x;
}
```

```
double maximum(int n, int m, double a[n][m]);
double maximum(int n, int m, double a[*][*]);
double maximum(int n, int m, double a[ ][*]);
double maximum(int n, int m, double a[ ][m]);
```

Inicializador

Syntax

initializer:

assignment-expression
{ *initializer-list* **}**
{ *initializer-list* **,** **}**

initializer-list:

$designation_{opt}$ *initializer*
initializer-list **,** $designation_{opt}$ *initializer*

designation:

designator-list **=**

designator-list:

designator
designator-list *designator*

designator:

[*constant-expression* **]**
. *identifier*

Semántica:

- Un inicializador especifica el valor inicial almacenado en un objeto.
- Si un objeto que tiene una duración de almacenamiento automática no se inicializa explícitamente, su valor es indeterminado. Si un objeto que tiene una duración de almacenamiento estática o de rosca no se inicializa explícitamente, entonces:
  - si tiene tipo de puntero, se inicializa a un puntero nulo;
  - si tiene tipo aritmético, se inicializa a cero (positivo o sin signo);
  - si se trata de un agregado, cada miembro se inicializa (recursivamente) de acuerdo con estas reglas, y cualquier relleno se inicializa a cero bits;
  - si se trata de una unión, el primer miembro nombrado se inicializa (recursivamente) de acuerdo con estas reglas, y cualquier relleno se inicializa a cero bits;
- Un array de tipo de carácter puede ser inicializado por una cadena de caracteres literal o UTF-8 literal, opcionalmente encerrada entre corchetes. Los bytes sucesivos de la cadena literal (incluyendo el carácter nulo de terminación si hay espacio o si el array es de tamaño desconocido) inicializan los elementos del array.
- Si un array de tamaño desconocido es inicializado, su tamaño es determinado por el elemento indexado más grande con un inicializador explícito. El tipo de array se completa al final de su lista de inicializadores.

EXAMPLE 2 The declaration

```
int x[] = { 1, 3, 5 };
```

EXAMPLE 3 The declaration

```
int y[4][3] = {
      { 1, 3, 5 },
      { 2, 4, 6 },
      { 3, 5, 7 },
};
```

is a definition with a fully bracketed initialization: 1, 3, and 5 initialize the first row of `y` (the array object `y[0]`), namely `y[0][0]`, `y[0][1]`, and `y[0][2]`. Likewise the next two lines initialize `y[1]` and `y[2]`. The initializer ends early, so `y[3]` is initialized with zeros. Precisely the same effect could have been achieved by

EXAMPLE 5 The declaration

```
struct { int a[3], b; } w[] = { { 1 }, 2 };
```

is a definition with an inconsistently bracketed initialization. It defines an array with two element structures: `w[0].a[0]` is 1 and `w[1].a[0]` is 2; all the other elements are zero.

EXAMPLE 7 One form of initialization that completes array types involves typedef names. Given the declaration

```
typedef int A[];  // OK - declared with block scope
```

the declaration

```
A a = { 1, 2 }, b = { 3, 4, 5 };
```

is identical to

```
int a[] = { 1, 2 }, b[] = { 3, 4, 5 };
```

due to the rules for incomplete types.

```
char s[] = "abc", t[3] = "abc";
```

defines "plain" **char** array objects `s` and `t` whose elements are initialized with character string literals. This declaration is identical to

```
char s[] = { 'a', 'b', 'c', '\0' },
     t[] = { 'a', 'b', 'c' };
```

The contents of the arrays are modifiable. On the other hand, the declaration

```
char *p = "abc";
```

defines `p` with type "pointer to **char**" and initializes it to point to an object with type "array of **char**" with length 4 whose elements are initialized with a character string literal. If an attempt is made to use `p` to modify the contents of the array, the behavior is undefined.

Declaraciones de selección:

Syntax

selection-statement:

if **(** *expression* **)** *statement*
if **(** *expression* **)** *statement* **else** *statement*
switch **(** *expression* **)** *statement*

Semántica:

- Una instrucción selection selecciona entre un conjunto de sentencias dependiendo del valor de una expresión controladora.
- Una instrucción selection es un bloque cuyo alcance es un subconjunto estricto del alcance de su bloque que lo encierra. Cada subcategoría asociada es también un bloque cuyo alcance es un subconjunto estricto del alcance de la instrucción de selección.

Instrucción IF

Limitación:

La expresión controladora de una instrucción if tendrá tipo escalar.

Semántica:

- En ambas formas, la primera subestación se ejecuta si la expresión se compara desigualmente a 0. En la otra forma, la segunda subestación se ejecuta si la expresión se compara igual a 0. Si la primera subestación se alcanza a través de una etiqueta, la segunda subestación no se ejecuta.
- Una otra se asocia con el léxico anterior más cercano si está permitido por la sintaxis.

Instrucción Switch

Limitación:

- La expresión controladora de una instrucción de conmutación tendrá un tipo entero.
- Si una instrucción switch tiene un caso asociado o una etiqueta predeterminada dentro del alcance de un identificador con un tipo modificado de forma variable, toda la instrucción switch estará dentro del alcance de dicho identificador.
- La expresión de cada etiqueta de caso será una expresión constante entera y no habrá dos de las expresiones constantes de caso en la misma instrucción de conmutación que tengan el mismo valor después de la conversión. Puede haber como mucho una etiqueta predeterminada en una instrucción switch. (Cualquier instrucción de conmutación cerrada puede tener una etiqueta predeterminada o expresiones constantes de mayúsculas con valores que duplican las expresiones constantes de mayúsculas en la instrucción de conmutación adjunta.

Semántica:

- Una instrucción `switch` hace que el control salte hacia, dentro o más allá de la instrucción que es el cuerpo del switch, dependiendo del valor de una expresión controlling, y de la presencia de una etiqueta predeterminada y los valores de cualquier etiqueta case en o en el cuerpo del switch. Una etiqueta de caso o predeterminada solo es accesible dentro de la instrucción de conmutación más cercana.
- Las promociones enteras se realizan en la expresión controladora. La expresión constante en cada etiqueta de caso se convierte al tipo promovido de la expresión controladora. Si un valor convertido coincide con el de la expresión de control promocionada, control salta a la instrucción que sigue a la etiqueta de caso coincidente. De lo contrario, si hay una etiqueta predeterminada, el control salta a la instrucción etiquetada. Si no coincide ninguna expresión constante de caja convertida y no hay una etiqueta predeterminada, no se ejecuta ninguna parte del cuerpo del conmutador.

EXAMPLE In the artificial program fragment

```
switch (expr)
{
      int i = 4;
      f(i);
case 0:
      i = 17;
      /* falls through into default code */
default:
      printf("%d\n", i);
}
```

the object whose identifier is `i` exists with automatic storage duration (within the block) but is never initialized, and thus if the controlling expression has a nonzero value, the call to the **`printf`** function will access an indeterminate value. Similarly, the call to the function `f` cannot be reached.

Declaraciones de iteración

Syntax

iteration-statement:

while (*expression*) *statement*
do *statement* **while** (*expression*) **;**
for ($expression_{opt}$ **;** $expression_{opt}$ **;** $expression_{opt}$) *statement*
for (*declaration* $expression_{opt}$ **;** $expression_{opt}$) *statement*

Limitaciones:

- La expresión controladora de una instrucción de iteración tendrá un tipo escalar.
- La parte de declaración de una declaración solo declarará identificadores para objetos que tengan auto o registro de clase de almacenamiento.

Semántica:

- Una instrucción iteración hace que una instrucción llamada cuerpo de bucle se ejecute repetidamente hasta que la expresión controladora se compare con 0. La repetición ocurre independientemente de si el cuerpo de bucle se introduce desde la instrucción iteración o por un salto.
- Una declaración de iteración es un bloque cuyo alcance es un subconjunto estricto del alcance de su bloque que lo encierra. El cuerpo del bucle es también un bloque cuyo alcance es un subconjunto estricto del alcance de la declaración de iteración.
- Una declaración de iteración puede ser asumida por la implementación para terminar si su expresión controladora no es una expresión constante) y ninguna de las siguientes operaciones se realizan en su cuerpo, la expresión controladora o (en el caso de una declaración para) su expresión -3:
    - operaciones de entrada/salida
    - acceder a un objeto volátil
    - sincronización o operaciones atómicas

La declaración mientras:

La evaluación de la expresión controladora tiene lugar antes de cada ejecución del cuerpo del bucle.

La declaración de hacer:

La evaluación de la expresión controladora tiene lugar después de cada ejecución del cuerpo del bucle.

La instrucción For:

```
for (clause-1; expression-2; expression-3) statement
```

La sentencia se comporta de la siguiente manera: La expresión-2 es la expresión controladora que se evalúa antes de cada ejecución del cuerpo del bucle. La expresión expresión-3 se evalúa como una expresión vacía después de cada ejecución del cuerpo del bucle. Si la cláusula-1 es una declaración, el alcance de cualquier identificador que declare es el resto de la declaración y el bucle completo, incluyendo las otras dos expresiones; se alcanza en el orden de ejecución antes de la primera evaluación de la expresión controladora. Si la cláusula-1 es una expresión, se evalúa como una expresión nula antes de la primera evaluación de la expresión controladora.

Declaraciones de salto

La instrucción Break

Limitaciones:

Una declaración de interrupción solo aparecerá en o como un cuerpo de conmutación o un cuerpo de bucle.

Semántica:

Una sentencia break finaliza la ejecución del interruptor o instrucción iteración más pequeño.

La declaración de respuesta

Limitaciones:

Una declaración de retorno con una expresión no aparecerá en una función cuyo tipo de retorno es nulo. Una declaración de retorno sin expresión sólo aparecerá en una función cuyo tipo de retorno es nulo.

Semántica:

- Una sentencia return termina la ejecución de la función actual y devuelve el control a su llamante. Una función puede tener cualquier número de sentencias return.

- Si se ejecuta una sentencia de retorno con una expresión, el valor de la expresión se devuelve al llamante como el valor de la expresión de llamada de la función. Si la expresión tiene un tipo diferente del tipo de retorno de la función en la que aparece, el valor se convierte como si por asignación a un objeto que tiene el tipo de retorno de la función.

Nombrado de variables:

Para el nombrado de las variables se utilizará el tipo de escritura Camel case, más específicamente lowerCamelCase donde la primera letra del nombre de la variable es minúscula y al inicio de cada palabra se escribirá en mayúscula. Ejemplo: ejemploDeLowerCamelCase. Dichos nombres deberán ser muy representativos con respecto a la función de las variables, siendo así fácilmente entendible su funcionamiento.

Archivos:

Los tipos de archivos a manejar serán de tipo .txt, dichos archivos se utilizarán para almacenar la información de las escuelas,las materias, los alumnos, los promedios y las calificaciones.

Bibliotecas a utilizar:

Libreria `<assert.h>`: La macro assert pone pruebas de diagnóstico en programas; se expande a una expresión vacía.

Librería `<stdio.h>`: Contiene los prototipos de las funciones, macros, y tipos para manipular datos de entrada y salida.

Librería `<stdlib.h>`: Contiene los prototipos de las funciones, macros, y tipos para utilidades de uso general.

Libreria `<string.h>`: Muy útil para el fácil uso de las cadenas de caracteres, pues elimina muchas de las dificultades que generan los char.

Libreria `<math.h>`: Una librería fundamental en este proyecto ya que contiene una gran variedad de operaciones matemáticas.

Librería `<windows.h>`: Librería sumamente importante al momento de declarar y utilizar a las funciones getch2() y gotoxy().

Comentarios:

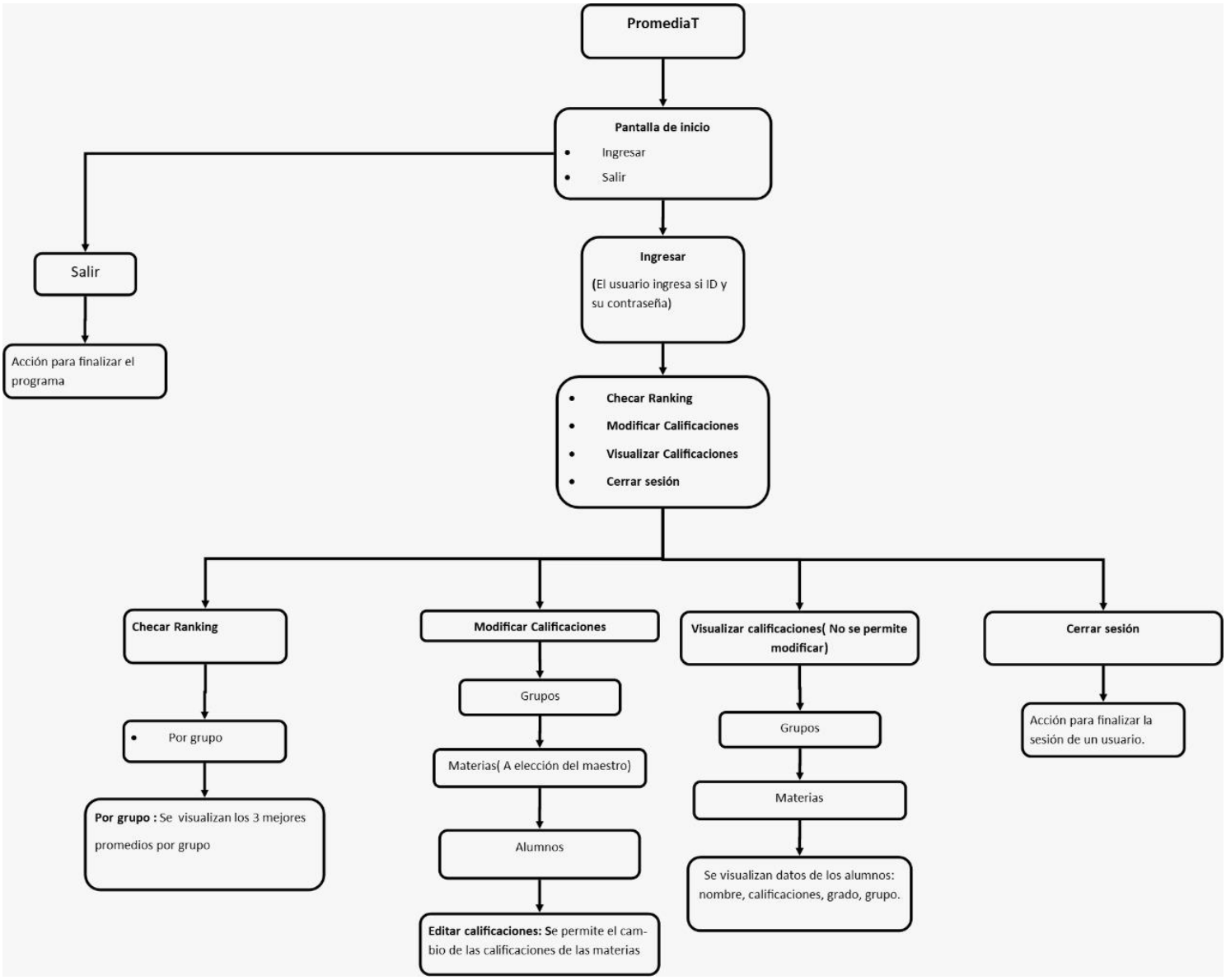
Excepto dentro de una constante de caracteres, una cadena literal o un comentario, los caracteres / introducen un comentario. El contenido de tal comentario se examina sólo para identificar caracteres multibyte y encontrar los caracteres */ que lo terminan.*

Excepto dentro de una constante de caracteres, una cadena literal o un comentario, los caracteres // introducen un comentario que incluye todos los caracteres multibyte hasta, pero no incluye, el siguiente carácter de nueva línea. El contenido de tal comentario se examina sólo para identificar caracteres multibyte y para encontrar el carácter de nueva línea que termina.

Se usará "//" para escribir comentarios breves donde se considere necesario en el código. En el caso de que se necesite explicar en profundidad una parte del código y se requiera el uso de más líneas se usará "/**/" de tal forma que no vea afectado el código. En los comentarios que deberán tener dichos bloques se deberá contemplar:

Funcionalidad que realiza tal fracción del código, ¿Qué guarda?(si lo hace),¿Se llama a alguna función?,¿Es parte de la entrada, del proceso o de la salida?.

Organización general del sistema



Proceso de desarrollo.

Descripción del proceso de desarrollo que se utilizará con herramientas, forma de organización, esquemas de monitoreo, bitácoras, medición del trabajo individual y de grupo.

Se agregará un marcador al aporte individual por cada punto realizado en cada entrega, los puntos se dividirán según su nivel de complejidad para repartirlos de manera justa y equitativa, para el aporte grupal se añadirá un marcador si aportó a los puntos más extensos los cuales se realizarán entre todos por medio de una llamada en equipo.

Al final se sumarán los marcadores de cada entrega para usarlos como divisor de los puntos realizados por cada uno para sacar su porcentaje de aportación

Ejemplo:

Integrantes	Entrega 1			Entrega 2			Entrega Final		
	Aporte Individual	Aporte Grupal	Porcentaje de aportación	Aporte Individual	Aporte grupal	Porcentaje de aportación	Aporte Individual	Aporte grupal	Porcentaje de Aportación
Alejandro Ake	**	*	3/9 = 33%	****	*	5/14 = 35%			
Andres Ortiz	*	*	2/9 = 22%	**	*	3/14 = 21%			
Jose Nahuat	*	*	2/9 = 22%	**	*	3/14 = 21%			
Rodrigo Euan	*	*	2/9 = 22%	**	*	3/14 = 21%			
Total	5	4	99%	10	4	98%			

Repositorio.

Dar clic aquí para visualizar el repositorio

Referencias

Alma. (s. f.). *Alma was built with your school in mind*. Alma. Recuperado de https://www.getalma.com/independent-charter

Aula 1. (s. f.). *Gestionar las calificaciones de tu centro con un software profesional*. Aula 1. Recuperado de https://www.aula1.com/calificaciones-software-profesional/

Flor. (2018, 20 de diciembre). *Gestión de calificaciones*. GQ dalya. Recuperado de https://www.gqdalya.com/gestion-calificaciones/