Informe del Proyecto: Sistema de Evaluación Basado en la Taxonomía de Bloom

---

1. Introducción

En este informe tiene como objetivo, describir el desarrollo de un sistema que se encarga de evaluaciones basado en la taxonomía de Bloom, implementado en el lenguaje de programación C++. Este algoritmo resuelve la necesidad de categorizar las preguntas de una prueba escrita según el nivel de taxonomía que evalúan. Esta categorización es clave para diseñar evaluaciones más efectivas y alineadas con los objetivos de aprendizaje.

---

2. Descripción de la Solución

2.1 Modelo de Clases y Arquitectura

El sistema fue diseñado utilizando principios de la programación orientada a objetos,en base a los contenidos . El modelo propuesto incluye cinco clases principales:

Clase Pregunta:
Esta clase representa una entidad abstracta que contiene los elementos comunes a todas las preguntas. Entre sus atributos se encuentran el enunciado de la pregunta y el nivel taxonómico que evalúa. Además, define métodos virtuales como mostrar()y mostrarNivelTaxonomico(), los cuales son redefinidos por las clases derivadas para adaptar su comportamiento.

Clase VerdaderoFalso:
Hereda de la clase Pregunta e implementa las particularidades de una pregunta de tipo verdadero/falso. Esta clase incluye atributos específicos como la respuesta esperada (booleano) y una justificación en caso de que la afirmación sea falsa. Su método mostrar() se describe al de la clase base para incluir estos elementos.

Clase Alternativa:
También hereda de Pregunta y representa preguntas de opción múltiple. Contiene un vector de alternativas y un índice que representa la respuesta correcta. Este tipo de pregunta requiere que el usuario indique cuántas opciones desea ingresar, lo cual se gestiona dinámicamente.

Clase Item:
Agrupa un conjunto de preguntas y representa una unidad evaluativa. Cada elemento posee un nombre y un vector de punteros a objetos Pregunta. Se incluyen métodos para agregar preguntas, mostrarlas y recuperar el conjunto de preguntas asociadas al ítem.

Clase Prueba:
Es la clase principal que orquesta la creación, actualización, eliminación y visualización de los ítems. También implementa un menú interactivo que permite al usuario controlar las operaciones del sistema desde la consola. La clase calcula el tiempo total estimado para resolver la prueba, calculando en el tipo de preguntas: 3 minutos por pregunta de verdadero/falso y 2 minutos por pregunta de opción múltiple.

2.2 Funcionalidades Clave del Sistema

El sistema permite crear una cantidad indefinida de artículos, cada uno con Múltiples preguntas. Durante la creación de un ítem o más de una prueba, el sistema solicita al usuario ingresar un nombre para el artículo y luego ingresar preguntas, seleccionando entre los dos tipos disponibles. En cada caso, también se solicita el nivel taxonómico correspondiente, entre 1 a 6, y la información asociada a cada tipo de pregunta.

Además de la creación, el sistema permite actualizar elementos existentes. El usuario puede seleccionar un item y reemplazar cualquiera de sus preguntas por una nueva, eligiendo nuevamente el tipo y la información correspondiente. Para mantener la integridad del sistema, se elimina manualmente la pregunta anterior antes de añadir la nueva, previniendo fugas de memoria.

El sistema también contempla la eliminación completa de un ítem. Esta operación libera la memoria asociada y remueve el ítem del vector que lo contiene. Asimismo, es posible visualizar todos los elementos creados, junto con sus preguntas, en un formato legible desde la consola.

Finalmente, se ha implementado una funcionalidad que calcula el tiempo total estimado que un estudiante podría tardar en responder una prueba completa. Esta estimación es útil para garantizar que las evaluaciones se ajusten a los tiempos definidos por el docente y al nivel de complejidad deseado.

2.3 Decisiones de diseño

Una de las decisiones más importantes fue usar jerarquía de clases basada en herencia. Esto permite encapsular el comportamiento común en la clase Pregunta, y delegar las particularidades de cada tipo de pregunta en sus respectivas subclases. Esta estrategia mejora la reutilización de código y facilita futuras ampliaciones del sistema, por ejemplo, si se desea incorporar más tipos de preguntas.

Para garantizar la manipulación eficiente de los objetos, se utiliza el contenedor vector. Gracias a esto, se pudo almacenar dinámicamente tanto las preguntas dentro de cada ítem como los ítems dentro de la prueba. Además, se optó por trabajar con punteros y liberar explícitamente la memoria para asegurar una gestión adecuada de los recursos del sistema.

El uso de métodos virtuales fue esencial para lograr el polimorfismo. Esto permite que, al recorrer las preguntas de un ítem, el sistema invoque automáticamente el método mostrar() adecuado a cada tipo de pregunta, mostrando cada elemento que nos han solicitado.

La interfaz con el usuario se diseñó íntegramente mediante la consola, respetando los requerimientos del enunciado. El menú presenta opciones claras y permite navegar entre las distintas operaciones disponibles, como crear, actualizar, eliminar y consultar elementos, además de calcular el tiempo estimado.

---

3. Conclusión

El desarrollo de este sistema permitió aplicar y consolidar conocimientos clave sobre la programación orientada a objetos en C++, tales como herencia, encapsulamiento, polimorfismo y manejo dinámico de memoria. Además, se abordó la problemática de estructurar evaluaciones educativas de manera significativa, utilizando como eje central la taxonomía de Bloom.

Se logró cumplir con todos los objetivos propuestos en el enunciado del trabajo: diseñar un modelo de clases coherente, implementar un sistema funcional que permita crear y gestionar evaluaciones, y calcular el tiempo estimado para cada pregunta. La consola actúa como interfaz principal, permitiendo al usuario interactuar con el sistema de forma sencilla y efectiva.