Instituto Tecnológico de Costa Rica

Unidad de Computación

1# proyecto de estructura de datos

Ryan Antonio Vargas Alvarado

Daniel Francisco Barrantes Zamora

Aaron Alonso Araya Guzman

Sede San Carlos

Fecha 23/09/2024

Para demostrar los conocimientos aprendidos hasta ahora, se nos asigno la realización de un proyecto, el cual consiste en implementar estructuras de datos dinámicas, en este caso, las listas, para dar una solución a problemas de diferentes tipos. Crearemos un total de 5 listas: “TiposTarea”, “Personas”, “ListaPendientes”,” tareasCompletadas”,” avanceTareas”, las cuales interactuaran entre ellas como un administrador de tareas, llevando un control de quien tiene x cantidad de tareas, el estado en las que se encuentran, el tipo de tarea, como además de separar las tareas entre finalizadas y en realización. Para ilustrar mucho mejor este trabajo, a continuación, se encuentra un diagrama que explica las variables que contendrá cada struct, junto con sus relaciones con otras listas (El diagrama además viene adjunto al documento, si se justa observar con mayor detalle):

Diagrama, Esquemático

Descripción generada automáticamente

Hay que tener en cuenta que cada una de las tareas insertadas en “listaPendientes” y “tareasCompletadas” tiene un enlace a lista “TiposTarea”. La lista “listaPendientes” solo usa el enlace hacia “avanceTareas” cuando el tipo de tarea sea estudio.

Hay que considerar además que hay que realizar un menú interactivo, en el cual se le puede pedir al algoritmo el realizar diferentes tipos de interacciones entre las listas. Estas peticiones son tanto como añadir o eliminar elementos a la lista, como además de procesos más complicados como separar las tareas por fecha y implimirlas.

**Análisis de resultados:**

Con respecto a los resultados finales, la siguiente tabla ilustrará el estado final del programa, con el fin de indagar en cada aspecto:

|  |  |  |
| --- | --- | --- |
| **Apartado** | **Estado completado** | **Comentarios** |
| Desarrollo de las listas | 100% | Las listas funcionan a la perfección y sin problemas. |
| **Actualización de información**  -  -Insertar tipo de tareas.  -Insertar personas.  -Borrar personas.  - Insertar tareas activas a una persona X (donde x es ingresado por el usuario).  - Modificar tareas activas, para que se puede reprogramar en otra fecha y hora.  - Borrar tareas activas.  - Insertar subtareas.  - Modificar el porcentaje de avance y estado de completado a una subtarea X.  - Insertar que una tarea se completó (insertando en la sublista de tareas realizadas y borradas de la sublista de tareas activas). | 100% | Todo perfecto. Un problema que se presentó fue que, al realizar una acción, esta saltaba hacia el otro item, sin volver al menú. Este problema parecía estar asociada a los “switch”, de todas formas este fue solucionado y el algoritmo funciona sin problemas. |
| **Consultas**  -  1. ¿Cuál es la persona que tiene más tareas activas?  2. ¿Cuál es la persona que tiene más tareas activas de un tipo X?  3. ¿Qué tipo de tarea es el más común? En caso de empate indicarlo.  4. ¿Cuál es la persona que tiene más tareas vencidas de un tipo X dado una fecha Y.  5. ¿Cuál es el tipo de tareas más común que se vence sin completarse, dado una fecha Y.  6. ¿Cuál es el tipo de *importancia* más usado por las personas?  7. ¿Qué es el tipo de tarea más común en tareas activas de importancia media?  8. ¿Qué es el tipo de tarea más común en tareas realizadas de importancia *Alta*? | 100% | Se utilizó la librería map, para simplificar el algoritmo de los item 3, 7 y 8. |
| **Reportes**  -  1. Imprimir la lista de tipos de tareas.  2. Imprimir las personas.  3. Imprimir las personas sin tareas activas.  4. Imprimir las tareas activas de una persona X, por orden de fecha y hora. En un rango de fechas ingresado por el usuario (fecha inicio y fecha final).  5. Imprimir las tareas próximo a vencer (menos de una semana) de una fecha X.  6. Imprimir todas las subtareas de una tarea X de una persona Y.  7. Imprimir las tareas realizadas por una persona X.  8. Imprimir las tareas realizadas al 100%. | 100% | Sin comentarios. |
| Menú | 100% | El menú es uno sencillo en terminal, con el cual se puede acceder a cada una de las funciones de actualización de datos, consultas y reportes. |
| Documentación interna | 100% | Cada función tiene documentada sus entradas como salidas, además de que se explica algunos apartados de código para mayor entendimiento. |
| Detalles menores | 100% | Partes como la función de cargar datos, arreglo de bugs y demás este hecho. |

**Conclusiones y recomendaciones:**

El trabajo esta realizado en un 100%, y realmente estamos satisfechos con el resultado. La interacción entre las listas cumple con lo que se pidió, así, por ejemplo, el añadir o eliminar elementos de la lista personas tiene repercusiones en otras interacciones, como el imprimir la lista completa. Por otro lado, para programar partes más complejas como el imprimir el tipo de tarea más usado entre las personas, se utilizó el #include <map>, el cual, gracias a su habilidad de almacenar elementos de forma mapeada con una clave, nos ayudó un montón con la simplificación de cierta parte del código. El apartado del menú fue pulido de tal forma que sea amigable y fácil de entender para el usuario, pudiendo realizar cualquier tarea en cualquier orden, sin problemas.

Existen un montón de librerías que acortan el proceso de programar un montón, por eso entre las recomendaciones, me gustaría recomendar el investigar cada cierto tiempo nuevas formas de realizar el código, pues como en este caso, si no hubiésemos visto las funciones de <map>, probablemente habríamos terminado haciendo un peor trabajo al actual.

Recomendamos aprovechar de mejor forma el tiempo en comparación a nosotros, si bien hemos podido terminar el trabajo a tiempo, no es sorpresa que probablemente se pudo haber realizado un mejor trabajo con una planificación distinta del tiempo. Esto además lo tendremos en cuenta para futuros proyectos.