Proyecto 2: El Problema de la Mochila

Viernes 19 de Septiembre

I. Descripción General

Se debe programar el algoritmo visto en clases para resolver el Problema General de la Mochila (*Knapsack*). La interfaz debe ser gráfica. La salida será un documento LaTeX que incluya el problema original, la tabla usada y su solución final detallada. Toda la programación debe realizarse en C sobre Linux. No se pueden cambiar las especificaciones de este documento.

II. Menú Principal

En el Proyecto 0 de este curso se desarrolló un menú principal que podía "lanzar" diversos algoritmos. El algoritmo solicitado en este proyecto será implementado como un **programa independiente**[1] que será ejecutado desde el menú mencionado. Se debe activar una "burbujita" con una descripción muy informativa de este algoritmo cuando el cursor se pose sobre la opción respectiva en el menú principal (*tooltip*). La interacción con el usuario se hará por medio de interfaces gráficas que deben ser desarrolladas con `GTK` y `Glade`. Debe haber un estilo uniforme en el *look & feel* de todas las interfaces.

III. Manejo de Archivos

Habrá un par de botones que permitan grabar archivos con los datos particulares de cada problema ingresado por el usuario (el usuario selecciona el nombre que desea para el archivo), y para que puedan ser cargados de nuevo y editados si así se desea. El formato de este archivo queda a discreción de cada grupo de trabajo. Se espera que, el día de la revisión, cada grupo cuente con bastantes archivos de pruebas para mostrar las capacidades de su proyecto.

IV. Problema de la Mochila

Usando los algoritmos vistos en clase, este programa resolverá el "problema de la mochila", en sus versiones 1/0, *bounded* y *unbounded*. Se espera que la interfaz gráfica sea lo más flexible posible. El usuario debe proporcionar la capacidad máxima de la mochila (entre 0 y 20), la cantidad de objetos a considerar (entre 1 y 10), y para cada objeto su costo, valor y cantidad disponible (incluido el caso de que esta cantidad fuera infinito).

Trabajo extra opcional 1: Manejar apropiadamente más de 10 objetos, no sólo en la interfaz sino también en el reporte final.

[1] No puede ser una función del programa de menú. La interfaz de este programa se desarrolla con Glade en un archivo independiente.

V. Reporte

Cuando el usuario considere que su problema ya está listo, presionará algún botón para arrancar la ejecución del algoritmo. Esto generará un documento con todo el proceso de solución.

Se debe generar un archivo "`.tex`" que será compilado con `pdflatex` o algún equivalente para que se produzca un archivo "`pdf`" que debe ser desplegado usando `evince` (o algún equivalente) en modo presentación.

La invocación de todos estos comandos debe realizarse internamente desde su programa, de manera transparente para el usuario. Deben quedar disponibles tanto el `pdf` como el fuente de LaTeX. Recuerde que la calidad final del documento es muy importante para la calificación de este proyecto.

El documento debe incluir por lo menos:

1. **Portada**: se identifica claramente a los miembros del grupo, el curso, el semestre y el nombre del proyecto.
2. **Problema de la Mochila**: explicar el problema clásico y sus variantes. Describir el algoritmo utilizado para resolverlo.
3. **Problema**: mostrar el problema ingresado, en notación matemática.
4. **Tabla de trabajo**: presentar la tabla de trabajo tal como se hizo en clases. Cada casilla mostrará el valor encontrado hasta ese punto, en color rojo o verde (o ambos si fuera un empate) y el valor de la variable de la columna correspondiente.
5. **Solución Óptima**: se muestra el valor óptimo de Z y el de todas las variables. Si hubiera varias soluciones óptimas, todas deben aparecer.

VI. Requisitos Indispensables

La ausencia de uno solo de los siguientes requisitos vuelve al proyecto "no revisable" y recibe un 0 de calificación inmediata:

- El programa se debe invocar desde el menú desarrollado en el Proyecto 0, el cual debe estar correcto (considerar cualquier observación que se les haya hecho).
- La colaboración entre grupos se considera fraude académico.
- Todo el código debe estar escrito en C
- Recuerde que usar IA para completar cualquier parte de su trabajo es fraude académico y tendrá muy serias consecuencias.
- El proyecto debe compilar y ejecutar en Linux. Todo debe estar **integrado**, explicaciones del tipo "*todo*

está bien pero no pudimos pegarlo"[2] provocan la cancelación automática de la revisión.

- Para el momento de la revisión, su programa no debe tener "prints" de depuración. Luce muy poco profesional.
- Todas las interfaces deben ser gráficas, robustas y de mucha calidad.
- Se debe usar `GTK` y `Glade`. Es inaceptable hacer una interfaz de texto desplegada en un *widget*.
- No debe dar "Segmentation Fault" bajo ninguna circunstancia.
- La demostración debe hacerse en una máquina que levante Linux de manera real (puede ser dual), es decir no usar máquinas virtuales. No se puede levantar de un disco o memoria externa. No se pueden prestar máquinas entre grupos.

VII. Fecha de Entrega

Revisiones a las 11:30am del **Viernes 19 de Septiembre** en la oficina del profesor. Mande además un `.tgz` con todo lo necesario (fuentes, makefile, readme, etc.) a `torresrojas.cursos.04@gmail.com`. Ponga como subject: `[IO] Proyecto 2 – Fulano – Mengano`, donde Fulano y Mengano son los miembros del grupo.

Mucha suerte...

[2] esto incluye los supuestos casos cuando alguien del grupo de trabajo no hizo su parte – el profesor no está interesado en sus problemas de organización.