PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

ALGORITMOS AVANZADOS

Primer Examen
(Segundo Semestre 2024)

Duración: 2h 45 min.

- **No puede utilizar apuntes, solo hojas sueltas en blanco.**
- En cada función el alumno deberá incluir, a modo de comentario, la forma de solución que utiliza para resolver el problema. De no incluirse dicho comentario, el alumno perderá el derecho a reclamo en esa pregunta.
- No puede emplear plantillas o funciones no vistas en los cursos de programación de la especialidad.
- Los programas deben ser desarrollados en el lenguaje C++. Si la implementación es diferente a la estrategia indicada o no la incluye, la pregunta no será corregida.
- Un programa que no muestre resultados coherentes y/o útiles será corregido sobre el 50% del puntaje asignado a dicha pregunta.
- Debe utilizar comentarios para explicar la lógica seguida en el programa elaborado. El orden será parte de la evaluación.
- Se utilizarán herramientas para la detección de plagios, por tal motivo si se encuentran soluciones similares, se anulará la evaluación a todos los implicados y se procederá con las medidas disciplinarias dispuestas por la FCI.
- **Solo está permitido acceder a la plataforma de PAIDEIA, cualquier tipo de navegación, búsqueda o uso de herramientas de comunicación se considera plagio por tal motivo se anulará la evaluación y se procederá con las medidas disciplinarias dispuestas por la FCI.**
- Para esta evaluación solo se permite el uso de las librerías **iostream, iomanip, climits cmath, fstream, vector, algorithm, string o cstring**
- Su trabajo deberá ser subido a PAIDEIA.
- **Es obligatorio usar como compilador NetBeans.**
- Los archivos deben llevar como nombre su código de la siguiente forma codigo_EX1_P# (donde # representa el número de la pregunta a resolver)

Pregunta 1 (10 puntos)

Dentro de la crianza de canarios la compra y venta de las aves es bastante rentable, ya que hay meses donde los ejemplares tienen un valor bajo y otros donde su precio sube debido a la demanda, por ejemplo, a inicios del año las aves tienen poco precio ya que aún no cuentan con sus plumas finales, mientras que antes de primavera su precio sube debido al inicio de la época reproductiva. Para realizar una adecuada compra y venta maximizando la ganancia, los criadores desean estimar cuanto ganarían al realizar **K** compras en un determinado rango de **N** fechas. Los procesos de compra y venta no pueden interceptarse. A continuación, se muestra un ejemplo:

Datos de Ingreso: N = 8

Mes 1	Mes 2	Mes 3	Mes 4	Mes 5	Mes 6	Mes 7	Mes 8
120	100	150	125	120	150	200	250

Para K = 1 compra

La ganancia máxima será 150 soles

El detalle de este resultado sería comprar el mes 2 y vender el mes 8.

Para K = 2 compras

La ganancia máxima será 180 soles

El detalle de este resultado sería un comprar el mes 2 y vender el mes 3, luego realizar una segunda compra el mes 5 y vender el mes 8.

Para resolver el problema se le solicita:

Utilizando **programación dinámica,** desarrolle un programe que le ayude a resolver este problema, que es calcular la máxima ganancia de al comprar y vender un canario. Recuerde que solo debe mostrar la ganancia, no es necesario mostrar el detalle de la asignación. Para esta pregunta debe usar los datos que se muestran en el ejemplo, **no** deben ingresarlos por el teclado o archivo.

Para que la solución sea válida se debe mostrar la matriz de soluciones parciales, recuerde que solo debe emplear iteraciones, si la respuesta emplea recursión o no muestra la matriz de soluciones parciales la pregunta queda anulada.

Pregunta 2 (10 puntos)

El problema de **Generación de Combinaciones de Paréntesis Balanceados** es un clásico de backtracking, donde el objetivo es generar todas las combinaciones posibles de paréntesis de apertura (y de cierre) para un número dado **n**, de tal manera que los paréntesis estén correctamente balanceados.

Dado un número ***n***, se debe generar todas las combinaciones de n pares de paréntesis bien balanceados. Un par de paréntesis está bien balanceado si:

- El número de paréntesis de cierre nunca supera al de apertura en ningún punto de la combinación.
- Al final de la combinación, el número de paréntesis de apertura es igual al de cierre.

A continuación, un ejemplo:

Si **n**=3, se debe generar todas las combinaciones posibles de 3 pares de paréntesis balanceados. Las combinaciones válidas serían:

Para cualquier valor de **n**, este proceso generaliza de la siguiente manera:

1. Empieza con una cadena vacía y va construyendo la combinación paso a paso.
2. Puede agregar un paréntesis de apertura si aún no ha llegado al límite de n aperturas.
3. Puede agregar un paréntesis de cierre si hay suficientes aperturas para balancearlos.
4. El proceso termina cuando la longitud de la cadena es 2*n, y se ha alcanzado un balance perfecto entre aperturas y cierres.

Construya un programa en C++ usando la técnica de **backtracking** que resuelva el problema e imprima como se mostró antes, el caso para n = 4

Pregunta 3 (10 puntos)

a) El problema de la cobertura de conjuntos (o Set Cover Problem) consiste en lo siguiente:
Dado un conjunto U (conjunto universal) y una colección de subconjuntos S_1, S_2, ..., S_m, donde cada subconjunto Si ⊆ U el objetivo es seleccionar un grupo de estos subconjuntos S_1, S_2, ..., S_m tal que:

1. La unión de los subconjuntos seleccionados cubra todos los elementos de U.
2. Se minimice el número de subconjuntos seleccionados.

En otras palabras, queremos cubrir todo el conjunto U utilizando la menor cantidad posible de subconjuntos de la colección dada.

A continuación, un ejemplo:

Supongamos que tenemos U = {1, 2, 3, 4, 5, 6, 7} y los subconjuntos:

$$S_1 = \{1, 2, 3\}, \quad S_2 = \{2, 4, 5\}, \quad S_3 = \{3, 5, 6\}, \quad S_4 = \{6, 7\}$$

Al seleccionar los subconjuntos

$S_1 = \{1, 2, 3\}$, $S_4 = \{6, 7\}$ y $S_2 = \{2, 4, 5\}$,

Hemos cubierto a U

Se pide desarrollar un programa que permita ingresar el conjunto U y los 4 subconjuntos anteriores y empleando **algoritmos voraces**, resuelva el problema planteado.

b) Se tiene *K* tareas como parte de una línea de producción, cada una de ellas realiza parte del proceso productivo de una fábrica demorando cada tarea una hora exactamente. Debido al tipo de producto que se desarrolla dentro del proceso, algunas tareas pueden realizarse hasta una determinada hora como máximo, pero dentro de un horario solo se puede ejecutar una sola tarea. Además, se sabe que cada tarea representa una ganancia para la empresa, por tal motivo la gerencia de producción desea seleccionar las tareas que le brinden la mayor ganancia. A continuación, algunos ejemplos:

Datos de Ingreso: K = 5

Tarea	A	B	C	D	E
Hora máxima	2	1	2	1	3
Ganancia	100	19	27	25	15

Solución:

Se consideran las tareas: c, a, e
Con una ganancia de: 142

Datos de Ingreso: K = 4

Tarea	A	B	C	D
Hora máxima	4	1	2	2
Ganancia	20	10	40	30

Solución:

Se consideran las tareas: d, c, a
Con una ganancia de: 90

Implemente un algoritmo voraz en C++ para seleccionar las tareas que nos brinden mayor ganancia, recuerde que **una hora solo puede tener una tarea asignada.**

Al finalizar el examen, comprima la carpeta de su proyecto empleando el programa Zip que viene por defecto en el Windows, **no se aceptarán los trabajos compactados con otros programas como RAR, WinRAR, 7zip o similares**. Luego súbalo a la tarea programa en Paideia para este examen.

Profesores del curso:

Manuel Tupia
Rony Cueva

San Miguel, 15 de octubre del 2024