

**Tecnología en Sistemas de Información**

**Proyecto Final FADA – Programación Dinámica**

**Elaborado por:**

Bryan Martínez - 1554623

Diego Fernando Nuñez Melo – 1554517

Juan Camilo Martínez - 1554324

**Universidad del Valle**

**Sede Palmira**

**Abril 2019**

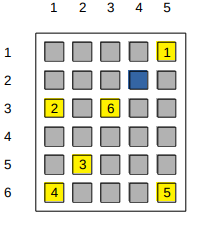
# Problema

Una empresa de mensajería local (solo hace entregas en la ciudad) necesita optimizar su proceso de entrega de encomiendas y para ello a acudido a los estudiantes de una prestigiosa universidad para que implementen un producto de software que permita especificar el orden en que deben ser entregadas las encomiendas.

Para mayor simplicidad del problema se tienen las siguientes restricciones:

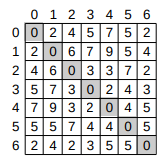
1. Las entregas serán realizadas por un único mensajero. En caso de haber más mensajeros, lo que se hace es que la ciudad se subdivide en regiones, cada una de ellas asignada a un mensajero (no se trabajará para efectos del proyecto). Además, cada encomienda tiene un valor que representa el dinero que el mensajero recibe por su entrega.

2. Se asumirá que la ciudad es una cuadricula de cuadras, con calles y carreras, donde el tiempo para desplazarse entre calles y carreras será el mismo para toda la ciudad. A continuación la representación de una ciudad de 5x6 manzanas (en amarillo las cuadras donde se deben entregar las encomiendas y en azul la ubicación de la oficina).



3. No se asumirán direcciones concretas (Calle X Cra Y ). Para efectos de especificar la dirección de entrega de una encomienda, se usará un sistema coordenado el cual especifica la manzana en donde se debe realizar una entrega. Por ejemplo, al encomienda No 4 se debe entregar en la manzana (6,1); la entrega 1 se realiza en la manzana (1,5). La manzana en azul (ubicación de la oficina) corresponde a la manzana (2,4).

4. A partir de la ubicación de las encomiendas, usted debe calcular la distancia entre la oficina y cada una de ellas, y entre ellas mismas. Para el ejemplo anterior, la matriz (simétrica) de distancias es:



## Problemas a Resolver

A continuación se listan los algoritmos que se deben diseñar:

1. Realizar la entrega de todos los paquetes en el menor tiempo posible. Para este caso, no es importante el valor que recibe el mensajero por realizar las entregas.

2. Realizar las entregas que le representen mayor beneficio al mensajero, en una cantidad de tiempo determinado (por ejemplo, diez unidades de tiempo).

3. OPCIONAL, hacer uso de técnicas voraces para resolver el problema de entregar todas las encomiendas, sin restricción de tiempo.

# Solución del problema

## Algoritmo recursivo (Pseudo-codigo)

Actual = Posición actual

Entregas [ ] = Agrego entregas

D [ ] [ ] = Matriz de distancia

**Entrega** ( actual, entregas [ ] , D [ ] [ ] ){

If ( entregas.size( ) == 1 ){

D [ actual ] [ entregas [ 0 ] ];

}

else {

min = min\_value;

For i = 0 to entregas.size do {

Encomienda e = entregas [ i ];

Copy [ ] = entregas [ ]

Copy.remove ( i );

Val = D [ actual ] [ e.getIndex ] + entrega ( e.getIndex , Copy [ ]

, D [ ] [ ] )

If ( val < min ){

min = val;

}

}

return min;

}

}

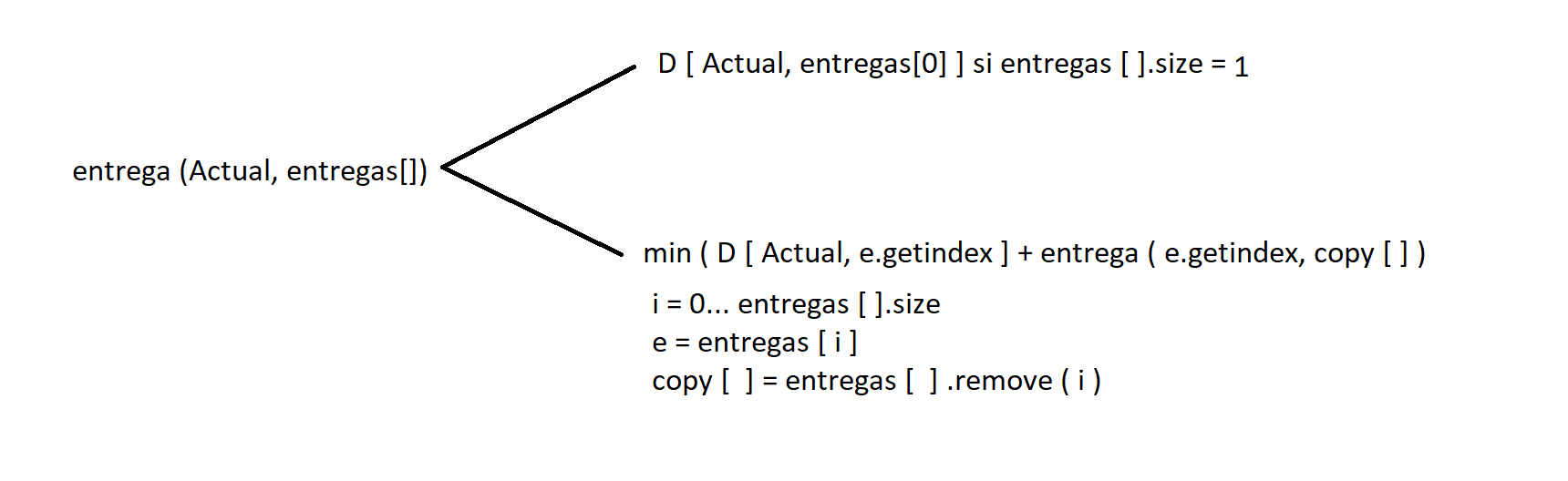
Consideraciones o modificaciones importantes al momento de realizar la implementación.

## Estructura optima

La especificación de la estructura óptima

Se realizó la estructura por medio de esta guía de recursividad: <Recursividad.pdf>

## Ecuación de recurrencia



## Algoritmo iterativo (Pseudo-codigo)

**Encomienda** (ArrayList<Encomienda> entregas, int[][] D) {

int S[][] = new int[D.length][D.length];

int E[][] = new int[D.length][D.length];

for (int i = 0; i < entregas.size(); i++) {

S[i][i] = D[i][i];

for (int j = 0; j < entregas.size(); j++) {

S[i][j] = +;

if (j == entregas.size() - 1) {

if (S[i][j] > D[i][j]) {

S[i][j] = D[i][j];

E[i][j] = j;

}

} else if (S[i][j] > D[i][j] + S[j][entregas.size() - 1]) {

S[i][j] = D[i][j] + S[j][entregas.size() - 1];

E[i][j] = j;

}

}

El algoritmo iterativo que calcula el costo óptimo (pseudo-código) y las consideraciones o modificaciones importantes al momento de realizar la implementación.

## Algoritmo (Pseudo-codigo)

El algoritmo (pseudo-código) que recostruye la solución óptima y las consideraciones o modificaciones importantes al momento de realizar la implementación.

## Análisis teórico de los algoritmos

Este algoritmo deberá de probar todas las posibles soluciones que puedan existir y deberá arrojar como resultado la ruta por la cual será más óptima la entrega de paquetes para el repartidor/mensajero.

Se prueba con un nodo inicial el cual al utilizarse se restara ya que si ya paso por este nodo no tendrá que volver a pasar por él, por lo tanto después de entregar x entrega se deberá de eliminar.

## Análisis experimental de los algoritmos

Al ingresar los valores para construir una matriz, se cargara dicha matriz y se procederá a realizar los cálculos correspondientes, se crea una nueva matriz la cual es la que se utiliza para conocer los costos óptimos y otra para conocer la solución óptima.

**(Gráfico con línea de tendencia y ecuación de la línea de tendencia)**

## Conclusiones

Conclusiones