

Laboratorio Nro. 4 Algoritmo Voraces



Objetivo: 1. Utilizar un algoritmo voraz para resolver un problema apropiado y determinar si la regla voraz conduce a una solución óptima o no. 2. Usar un enfoque heurístico para resolver un problema apropiado. 3. Describir las ventajas y desventajas entre una estrategia de fuerza bruta y una estrategia voraz.



Consideraciones: Lean y verifiquen las consideraciones de entrega,



Leer la Guía



Trabajar en
Parejas



Si tienen reclamos,
regístenlos en
<http://bit.ly/2g4TTKf>



Ver
calificaciones
en Eafit
Interactiva



Subir el informe pdf en la carpeta **informe**, el código del ejercicio 1 en la carpeta **codigo** y el código del 2 en la carpeta **ejercicioEnLinea**

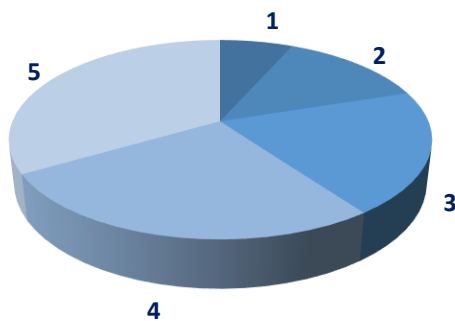


Hoy, plazo
de entrega



Si toman la respuesta de **alguna fuente**, deben referenciar según el **tipo de cita**.
Vean *Guía numerales 4.16 y 4.17*

Porcentajes y criterios de evaluación



- 1. Simulacro sustentación proyecto
- 2. Análisis de complejidad
- 3. Código de laboratorio
- 4. Simulacro de parcial
- 5. Ejercicio con juez en línea

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

1. Simulacro de Proyecto

Códigos para entregar en GitHub en la carpeta `codigo`

1 Simulacro de Proyecto

Códigos para entregar en GitHub en la carpeta `codigo`:



Vean Guía
numeral 3.4



Código de laboratorio en
GitHub. Vean Guía en
numeral 4.24



Documentación opcional. Si lo hacen, utilicen **Javadoc** o equivalente. No suban el HTML a GitHub.



No se reciben archivos en **.RAR** ni en **.ZIP**



Utilicen Java, C++ o Python



En la vida real, la documentación del software hace parte de muchos estándares de calidad como CMMI e ISO/IEC 9126



En la vida real, el problema del agente viajero se aplica para la construcción de circuitos electrónicos, recolectar monedas de teléfonos de monedas, entre otras. Léase más en <http://bit.ly/2i9JdIV>

- 1.1** La empresa FedEx tiene como misión entregar paquetes a sus clientes en el menor tiempo posible, como es retratada en la película “*Naúfrago (2000)*” protagonizada por Tom Hanks.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

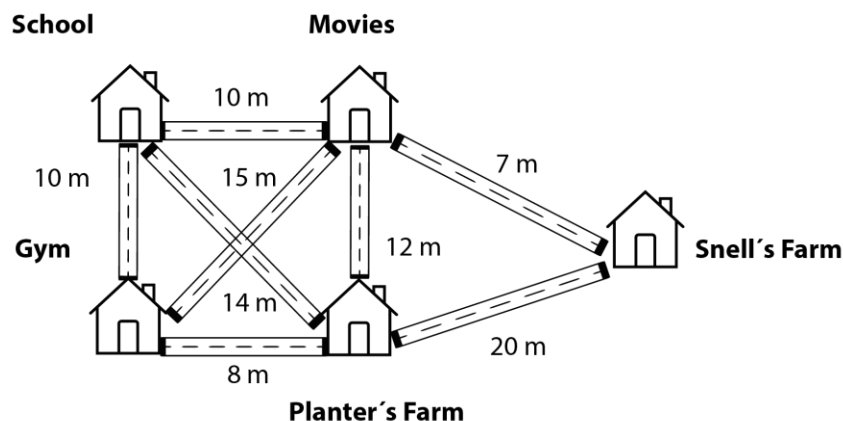
Código ST0247

Un camión de FedEx, carga todos los paquetes en un depósito, reparte los paquetes a cada uno de sus clientes y regresa al depósito. Para atender más rápido a sus clientes, la distancia total del recorrido debe ser mínima. **Este problema se conoce como el problema del agente viajero.**

▶ Implementen la solución al problema del agente viajero usando un algoritmo voraz.



Ejemplo 1, Para realizar una prueba, en *Github* encontrará el archivo *puentes_colgantes.txt* que contiene el ejemplo a continuación. También encontrará un mapa de la ciudad de Medellín.



Vertices. Formato: ID, coordenada x, coordenada y, nombre

```
10000 2.00000 0.00000 School
1 4.00000 1.00000 Movies
2 5.00000 2.00000 Snell
3 2.00000 5.00000 Planters
4 0.00000 2.00000 Gym
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

Arcos. Formato: ID, ID, distancia, nombre

```

10000 1 10.0 Calle 1
10000 3 14.0 desconocido
10000 4 10.0 desconocido
1 10000 10.0 Calle 2ª
1 2 7.0 desconocido
1 3 12.0 desconocido
1 4 15.0 desconocido
2 1 7.0 desconocido
2 3 20.0 desconocido
3 10000 14.0 desconocido
3 1 12.0 desconocido
3 2 20.0 desconocido
3 4 8.0 desconocido
4 10000 10.0 desconocido
4 1 15.0 desconocido
4 3 8.0 desconocido

```



En la vida real, una exigencia del mercado es que los ingenieros de sistemas conozcan la metodología de Desarrollo de Software dirigido por *testing* (en Inglés *TDD*), en la que primero se hacen los *tests* unitarios antes de empezar la programación

1.2

[Opc]

Realicen pruebas unitarias para el problema anterior usando JUnit o su equivalente en C++ o Python.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

2. Simulacro de Maratón de Programación sin documentación HTML, en GitHub, en la carpeta `ejercicioEnLinea`

2 Simulacro de Maratón de Programación sin documentación HTML, en GitHub, en la carpeta ejercicioEnLinea



Vean Guía
numeral 3.3



No entregar
documentación **HTML**



Utilicen Java, C++ o
Python



No se reciben archivos
en **.PDF**



No se reciben
archivos en **.RAR** ni
en **.ZIP**



Código del ejercicio en
línea en **GitHub**. Vean
Guía en numeral 4.24

2.1 En una ciudad hay n conductores de bus. También hay n rutas de bus en la mañana y n rutas de bus en la tarde con varias duraciones. Cada conductor es asignado una ruta en la mañana y una ruta en la noche.

Para cada conductor, si la duración total de su ruta por un día excede d , él tiene que ser pagado por el tiempo extra por cada hora después de su básico a una tarifa de r pesos por hora.



Su tarea es asignar una ruta en la mañana y una ruta en la tarde a cada conductor de tal forma que el tiempo total extra que haya que pagar por parte de la empresa sea el mínimo.



Entrada

La línea de cada caso de prueba tiene 3 enteros, n , d y r , como descriptores anteriormente. En la segunda línea, hay n enteros separados con espacios, que son las duraciones de la rutas de la mañana dadas en horas.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

De igual manera, en la tercera línea hay n enteros separados por espacio que son las duraciones, en horas, de las rutas de la tarde. Las duraciones son enteros positivos menores o iguales a 10.000. El fin de la entrada está dado por un caso con tres ceros.



Salida

Para cada caso de prueba, imprima el mínimo posible valor de horas extras que la empresa de transporte debe pagar.



Restricciones

- $1 < n < 100$
- $1 < d < 10000$
- $1 < r < 5$



Ejemplos de las entradas

```
2 20 5
10 15
10 15
2 20 5
10 10
10 10
0 0 0
```



Ejemplos de la salida

```
50
0
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

3. Simulacro de preguntas de sustentación de Proyecto en la carpeta informe

3 Simulacro de preguntas de sustentación de Proyecto en la carpeta informe



Vean **Guía**
numeral 3.4



Exportar y entregar informe
de laboratorio en **PDF**, en
español o Inglés



Si hacen el **informe**
en español, usen la
plantilla en español



No apliquen **Normas**
Icontec para esto

Si hacen **el informe**
en inglés, usen a
plantilla en inglés



En la vida real, las técnicas usadas para resolver muchos
problemas en Ingeniería de Sistemas son las mismas que las
existentes para las N reinas

Sobre el Simulacro del Proyecto

3.1



Expliquen con sus propias palabras la estructura de datos que utilizan para
resolver el problema del numeral 1.1 y cómo funciona el algoritmo.





Nota: Recuerden que deben explicar su implementación en el informe PDF

PhD. Mauricio Toro Bermúdez




Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

- 3.2**  Para resolver el problema del agente viajero, usando un algoritmo voraz, ¿Entrega siempre la solución óptima? ¿Qué debe cumplir el grafo para que el algoritmo, al menos, arroje una solución, así no sea óptima? ¿Por qué?
- 3.3**  ¿Cómo se puede adaptar la solución voraz del agente viajero para entregar domicilios en la ciudad de Medellín?
- ¿El objetivo sería pasar por todos los vértices del grafo o sólo por los puntos donde hay que entregar un domicilio? ¿Cómo calcular la distancia que hay entre los puntos donde hay que entregar un domicilio?

Sobre el simulacro de maratón de programación

- 3.4**  Expliquen con sus propias palabras la estructura de datos que utiliza para resolver el problema del numeral 2.1 y cómo funciona el algoritmo.
- 3.5**  Calculen la complejidad de los ejercicios en línea del numeral 2.1 y agréguenla al informe PDF
- 3.6**  Expliquen con sus palabras las variables (*qué es 'n', qué es 'm', etc.*) del cálculo de complejidad del numeral anterior



Ejemplo de esta respuesta:

“n es el número de elementos del arreglo”,
 “V es el número de vértices del grafo”,
 “n es el número de filas de la matriz y m el número de columnas”.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

4. Simulacro de parcial en informe PDF

4 Simulacro de Parcial en el informe PDF

Resuelvan los ejercicios



Para este simulacro, agreguen **sus respuestas** en el informe PDF.



El día del Parcial no tendrán computador, JAVA o acceso a internet.



Si hacen el **informe en español**, usen la **plantilla en español**



Exportar y entregar informe de laboratorio en **PDF**, en **español o inglés**

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



No apliquen **Normas Icontec** para esto

4.1 El problema de **selección de actividades** consiste en encontrar el máximo número de actividades que una persona o una máquina puede resolver, asumiendo que la persona o máquina sólo puede hacer una actividad al mismo tiempo.

Este problema es de interés en Ingeniería de Producción. Cada actividad tiene un tiempo de inicio y un tiempo de fin. Como un ejemplo, consideren las siguientes actividades:

Act.	a1	a2	a3	a4	a5	a6	a7	a8
inicio	1	0	1	4	2	5	3	4
fin	3	4	2	6	9	8	5	5

Nuestra tarea es encontrar el máximo número de actividades que se puedan ejecutar y que no estén en conflicto (es decir, que no ocurran dos al mismo tiempo).

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

Un algoritmo voraz para resolver el problema funciona de la siguiente forma:

- a) Ordenar las actividades de menor a mayor, según el tiempo de fin de la actividad
- b) Seleccionar la primera actividad
- c) Repetir hasta que no se puedan seleccionar más actividades:
 - Seleccionar una nueva actividad cuyo tiempo de inicio sea mayor igual al tiempo de fin de la actividad previamente seleccionado.

Para el ejemplo anterior, el algoritmo debe entregar la siguiente respuesta:

Actividad	Inicio	Fin
a3	1	2
a7	3	5
a6	5	8

A continuación, observen una implementación del algoritmo en Java:

```
class Actividad {
    public Actividad(String a,int b,int c) {
        id = a; inicio = b; fin = c;}
    String id;
    int inicio;
    int fin;
}

void seleccion(Actividad actividades[]) {
    int i, j;
    int n = actividades.length;
    Actividad temp;
    //paso 1
    for(i = 1; i < n; i++) {
        for(j = 0; j < n - 1; j++){
            if(actividades[j].fin > actividades[j+1].fin) {
                temp = actividades[j];
                actividades[j] = actividades[j+1];
                actividades[j+1] = temp;
            }
        }
    }
}
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

```

}
//paso 2
System.out.println("Actividad Inicio Fin");
System.out.println(actividades[0].id + " " +
                    actividades[0].inicio + " " +
                    actividades[0].fin);

//paso 3
i = 0;
for(j = 1; j < n; j++) {
    if(actividades[j].inicio >= actividades[i].fin) {
        System.out.println(actividades[j].id + " " +
                            actividades[j].inicio + " " +
                            actividades[j].fin);
    }
}

```



De acuerdo a lo anterior, completen el espacio faltante en la última línea

4.2 El problema del **agente viajero** consiste en responder la siguiente pregunta: Dada una lista de ciudades y las distancias entre cada par de ciudades, ¿cuál es la ruta posible más corta que visita a cada ciudad exactamente una vez y regresa a la ciudad de origen?

Un algoritmo voraz para solucionar este problema es el **algoritmo del vecino más cercano**. El algoritmo empieza en la primera ciudad y selecciona en cada iteración una nueva ciudad, no visitada, que sea la más cercana a la inmediatamente anterior.

A continuación, una implementación del algoritmo del vecino más cercano que recibe como parámetro un grafo representado como matriz de adyacencia.

La persona que hizo este código es un ingeniero matemático. En *Matlab* los índices empiezan en 1. Por esta razón, el camino que encuentra el algoritmo empieza con la ciudad numerada con 1 y trabaja con ciclos `while` en lugar de `for`.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

Adicionalmente, el programador inicia los índices en 1, por ejemplo $i = 1$. Por si fuera poco, la variable `min` no era necesaria inicializarla fuera del bloque `while` y `minFlag` hubiera sido mejor declararla dentro del bloque `while`.

Finalmente, el arreglo de visitados sería más eficiente haberlo hecho con tipo `boolean`. No obstante, el programa funciona en Java y tiene una complejidad de , que es la esperada para este algoritmo

```

01 public void tsp(int adjacencyMatrix[][]) {
02     Stack<Integer> stack = new Stack<Integer>();
03     int numberOfNodes = adjacencyMatrix[1].length - 1;
04     int[] visited = new int[numberOfNodes + 1];
05     visited[1] = 1;
06     stack.push(1);
07     int element, dst = 0, i;
08     int min = Integer.MAX_VALUE;
09     boolean minFlag = false;
10     System.out.print(1 + "\t");
11     while (!stack.isEmpty()) {
12         element = stack.peek();
13         i = 1;
14         min = Integer.MAX_VALUE;
15         while (i <= numberOfNodes) {
16             if (adjacencyMatrix[element][i] > 0
17                 && visited[i] == 0) {
18                 if (_____ > _____) {
19                     min = adjacencyMatrix[element][i];
20                     dst = i;
21                     minFlag = true;
22                 }
23             }
24             i++;
25         }
26         if (minFlag) {
27             visited[dst] = 1;
28             stack.push(dst);
29             System.out.print(dst + "\t");
30             minFlag = false;
31             continue;
32         }
33         stack.pop();
34     }
35 }

```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473



`Continue` es una palabra reservada en Java que permite terminar una iteración de un ciclo abruptamente y pasar a la siguiente iteración del ciclo.



De acuerdo a lo anterior, completen el espacio faltante en la línea 18

4.3 [Opc] El algoritmo de Dijkstra sirve para encontrar el camino más corto de un vértice a todos los demás de un grafo. A continuación, una implementación en Java.

```
int minVertex (int [] dist, boolean [] v) {
    int x = Integer.MAX_VALUE; //Infinity
    int y = -1;
    for (int i=0; i<dist.length; i++)
        if (!v[i] && dist[i]<x)
            y=i; x=dist[i];
    return y;
}

int [] dijsktra(Graph dg, int source) {
    int [] dist = new int [dg.size()];
    int [] pred = new int [dg.size()];
    boolean [] visited = new boolean [dg.size()];
    for (int i=0; i<dist.length; i++)
        dist[i] = Integer.MAX_VALUE;
    dist[source] = 0;
    for (int i=0; i<dist.length; i++) {
        int next = minVertex (dist, visited);
        visited[next] = true;
        ArrayList<Integer> n =
            dg.getSuccessors (next);
        for (int j=0; j<n.size(); j++) {
            int v = n.get(j);
            int d = dist[next] +
                dg.getWeight(next,v);
            if (dist[v] > d) {
                dist[v] = d;
            }
        }
    }
}
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

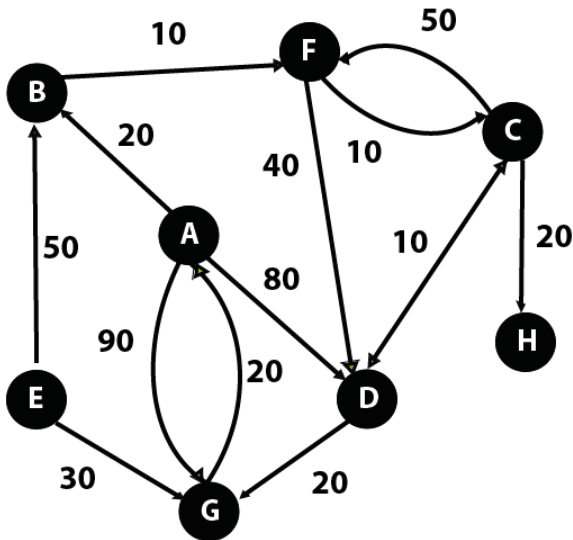
ESTRUCTURA DE DATOS 2
Código ST0247

```

    pred[v] = next;
  }}}
  return pred;
}

```

Consideren el siguiente grafo de *Dijkstra*:



▶ De acuerdo a lo anterior, resuelvan lo siguiente:

4.3.1 Completen, por favor, la siguiente tabla, usando el algoritmo de Dijkstra para encontrar el camino más corto del punto A a todos los demás. En la tabla, la palabra “a” significa “hasta”.

Paso	A	B	C	D	E	F	G	H
1	A	20,A	∞	80, A	∞	∞	90, A	∞
2	B	20,A	∞	80, A	∞	30,B	90,A	∞
3								
4								
5								
6								
7								
8								

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

4.3.2 ¿Cuál es el camino más corto de A a G?

- 4.4** Consideren el siguiente problema. El doctor le ha recomendado comer a vitaminas, b proteínas y c minerales. Ustedes pueden ir a un lugar donde hay infinitas cantidades de éstas tres cosas; además, ustedes no tienen que pagar por nada.

Sin embargo, uno puede ir sólo una vez por día y cada día que uno va al lugar tiene que tomar una de estas dos posibilidades:

- Tomar 3 cosas distintas (vitaminas, proteínas y minerales), es decir, pueden tomar una vitamina, una proteína y un mineral;
- Tomar 2 cosas del mismo tipo, es decir, 2 vitaminas, 2 proteínas o 2 minerales.

Como ustedes están muy enfermos, quieren saber cuál es la mínima cantidad de días que tiene gastar para ir a ese lugar para tener al menos a vitaminas, b proteínas y c minerales.



Como un ejemplo, Supongan que $a=3$, $b=4$, $c=7$. La solución óptima sería ir los tres primeros días por 3 cosas distintas. Luego los siguientes días ir por 2 cosas iguales.

Si se hace el cálculo se demostrará fácilmente que ustedes tienen que ir 3 días seguidos por dos cosas distintas. Por lo tanto, la solución sería 6 días. Es decir, un posible recorrido con el mínimo número de viajes es el siguiente:

1. Ir por 1 vitamina, 1 proteína y 1 mineral
2. Ir por 1 vitamina, 1 proteína y 1 mineral
3. Ir por 1 vitamina, 1 proteína y 1 mineral
4. Ir por 2 proteínas
5. Ir por 2 minerales
6. Ir por 2 minerales

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2

Código ST0247

```

01 int numeroDias(int a, int b, int c){
02     //Intentar primero tomando de a 3
03     int minimo = Math.min(a, Math.min(b, c));
04     //Hay que quitarlas a cada variable
05     a = a - minimo;
06     b = b - minimo;
07     c = c - minimo;
08     //Ahora tomemos de a 2 cosas.
09     int temp = a + b + c + 1;
10     temp = _____;
11     return _____;
12 }

```



Pista 1: La forma óptima de ir por tres cosas cada día es solo cuando tienen que ir por de las tres cosas, es decir, cuando $a > 0 \wedge b > 0 \wedge c > 0$.

La mejor forma de ir por dos cosas cada día es cuando solo tienen que ir por a lo sumo dos cosas, es decir, $a = 0 \vee b = 0 \vee c = 0$.



Pista 2: Verifiquen que $a = \text{minimo}$, $b = \text{minimo}$, $c = \text{minimo}$.



De acuerdo a lo anterior, completen en el espacio las siguientes líneas:

4.4.1 Completan la línea 10: _____

4.4.2 Completan la línea 11: _____

4.4.3 ¿Cuál es la complejidad del método anterior?

- a) $O(n)$
- b) $O(1)$
- c) $O(\log n)$
- d) $O(n \cdot \log n)$

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

4.5 [Opc] Durante la clase de Estructuras de Datos 2, el profesor propuso el siguiente problema para que se resolviera en parejas. Te daban n números y un entero $k, k \leq n$. La tarea era encontrar un subconjunto de exactamente k números cuya suma fuera mínima.

Sin embargo, para que el problema fuera más entretenido, el profesor prometió un *bonus* a la pareja que escribiera un algoritmo con una complejidad no superior a $O(n \log n)$. Para esto, las parejas 1 y 2 propusieron lo siguiente, respectivamente:

- Es imposible encontrar un algoritmo que funcione en complejidad $O(n \log n)$ o inferior porque es necesario intentar cada subconjunto de k números para determinar cuál de estos tiene suma mínima.
- Es posible encontrar un algoritmo que funcione con complejidad $O(n \log n)$ o inferior porque, si ordenamos los elementos de una manera específica, no es necesario intentar todos los posibles subconjuntos de k elementos.



De acuerdo a lo anterior:

4.5.1 Elijan la respuesta correcta:

- a) La pareja 1 tiene la razón, pero su justificación es falsa
- b) La pareja 1 tiene la razón y su justificación es verdadera
- c) La pareja 2 tiene la razón, pero su justificación es falsa
- d) La pareja 2 tiene la razón y su justificación es verdadera

4.5.2 Describan en pocas líneas el algoritmo que ustedes plantean para el problema anterior. Expliquen la complejidad de su algoritmo.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

4.6 Se está preparando la llegada del Rey a su castillo. Por seguridad se han ubicado N puestos de guardia, cada uno a $10m$ de distancia. Ya se han ubicado algunos guardias en posiciones estratégicas.

Se sabe que el guardia en la posición i protegerá al guardia en la posición j si $|j-i| \leq K$. En la posición 0 y en la posición $N-1$ siempre hay guardias, pero hay algunas posiciones que aún no se han cubierto.

La guardia real quiere saber cuál es la mínima cantidad de guardias que tiene que contratar, de tal manera que todos los guardias siempre se estén cuidando mutuamente y ¡haya más seguridad!

Las posiciones se entregan como un arreglo A donde la posición i del arreglo contiene el número $i+1$ si la posición i está ocupada por un guardia o un 0 si no hay un guardia en esa posición.



Ejemplos: Para $x_1 = \{1,2,0,0,5,0,0,8\}$, $K_1 = 2$, la respuesta sería 2. Para $x_2 = \{1,0,3,4\}$, $K_2 = 2$, la respuesta sería 0.

Para $x_3 = \{1,0,0,0,5\}$, $K_3 = 2$, la respuesta sería 1. Para el primer ejemplo, sería óptimo ubicar guardias en las posiciones 2 y 6. Así, todos los guardias se protegerán mutuamente.

```
int solucion(int[] x, int K){
    int last = 0; //ultimo guardia
    int res = 0; //respuesta
    int n = x.length;
    for (int i = 0; i < n; ++i) {
        if (x[i] == ..... ) last = i;
        if (i - last == K) {
            res = .....;
            last = .....;
        }
    }
    return res;
}
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 2
Código ST0247



De acuerdo a lo anterior:

4.6.1 Completen la línea 6 _____

4.6.2 Completen la línea 8 _____

4.6.3 Completen la línea 9 _____

4.6.4 Determinen la salida para $x_t = \{1,0,0,4,0,0,0,0,0,11\}$, $K_t=3$:

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



5. [Opcional]

Lecturas Recomendadas

5 [Opc] Lecturas recomendadas



Vean *Guía* en **numeral 3.5 y 4.20**



Exportar y entregar informe de laboratorio en **PDF**, en **español o Inglés**



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



"El ejercicio de una profesión requiere la adquisición de competencias que se sustentan en procesos comunicativos. Así cuando se entrevista a un ingeniero recién egresado para un empleo, una buena parte de sus posibilidades radica en su capacidad de comunicación; pero se ha observado que esta es una de sus principales debilidades..." *Tomado de <http://bit.ly/2gJKzJD>*



Lean a del "**R.C.T Lee et al., Introducción al análisis y diseño de Algoritmos. Capítulo 3. Páginas 71 - 115.**", y sumen puntos adicionales, así:



Hagan un mapa conceptual con los principales elementos teóricos.



Si desean una lectura adicional en español, consideren la siguiente "**John Hopcroft et al., Estructuras de Datos y Algoritmos, Sección 10.3. 1983**", que encuentran en biblioteca



Nota: Estas respuestas también deben incluirlas en el informe PDF

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

6. [Opcional] Trabajo en Equipo y Progreso Gradual

6 [Opc] Trabajo en Equipo y Progreso Gradual



Vean *Guía* en **numeral 3.6, 4.21, 4.22 y 4.23**



Exportar y entregar informe de laboratorio en **PDF, en español o Inglés**



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



El trabajo en equipo es imprescindible. "Algunos medios retratan la programación como un trabajo solitario, la realidad es que requiere de mucha comunicación y trabajo con otros. En la vida laboral serás parte de un equipo de trabajo y deberás comunicarte con otras personas". Tomado de <http://bit.ly/2qJKzJD>

- 6.1** ▶ Entreguen copia de todas las actas de reunión usando el *tablero Kanban*, con fecha, hora e integrantes que participaron
- 6.2** ▶ Entreguen el reporte de *git* con los cambios en el código y quién hizo cada cambio, con fecha, hora e integrantes que participaron
- 6.3** ▶ Entreguen el reporte de cambios del informe de laboratorio que se genera *Google docs* o herramientas similares
- !** **NOTA:** Estas respuestas también deben incluirlas en el informe PDF

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

7. [Opcional]

Laboratorio en Inglés con plantilla en Inglés

7 [Opc] Laboratorio en inglés



Vean *Guía* en **numeral 3.6, 4.21, 4.22 y 4.23**



Exportar y entregar informe de laboratorio en **PDF, en español o Inglés**



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



El inglés es un idioma importante en la Ingeniería de Sistemas porque la mayoría de los avances en tecnología se publican en este idioma y la traducción, usualmente se demora un tiempo.

Adicionalmente, dominar el inglés permite conseguir trabajos en el exterior que son muy bien remunerados. Tomado de goo.gl/4s3LmZ



Entreguen el código y el informe en inglés.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Resumen de ejercicios a resolver

1.1 Implementen la solución al problema del agente viajero usando un algoritmo voraz

1.2 Realicen pruebas unitarias para el problema anterior usando JUnit o su equivalente en C++ o Python.

2.1 Resuelvan el siguiente problema usando algoritmos voraces

3.1 Expliquen con sus propias palabras la estructura de datos que utiliza para resolver el problema del numeral 1.1 y cómo funciona el algoritmo.

3.2 Para resolver el problema del agente viajero, usando un algoritmo voraz, ¿Entrega siempre la solución óptima? ¿Qué debe cumplir el grafo para que el algoritmo, al menos, arroje una solución, así no sea óptima? ¿Por qué?

3.3. ¿Cómo se puede adaptar la solución voraz del agente viajero para entregar domicilios en la ciudad de Medellín? ¿El objetivo sería pasar por todos los vértices del grafo o sólo por los puntos donde hay que entregar un domicilio? ¿Cómo calcular la distancia que hay entre los puntos donde hay que entregar un domicilio?

3.4 Expliquen con sus propias palabras la estructura de datos que utiliza para resolver el problema del numeral 2.1 y cómo funciona el algoritmo.

3.5 Expliquen la complejidad de los ejercicios en línea del numeral 2.1 y agréguela al informe PDF

3.6 Expliquen con sus palabras las variables (*qué es 'n', qué es 'm', etc.*) del cálculo de complejidad del numeral anterior

4. Simulacro de Parcial

5. [Ejercicio Opcional] Lectura recomendada

6. [Ejercicio Opcional] Trabajo en Equipo y Progreso Gradual

7. [Ejercicio Opcional] Entreguen el código y el informe traducido al inglés.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ayudas para resolver los Ejercicios

Ayudas para el Ejercicio 1.....	<u>Pág. 32</u>
Ayudas para el Ejercicio 2.1.....	<u>Pág. 32</u>
Ayudas para el Ejercicio 3.2.....	<u>Pág. 32</u>
Ayudas para el Ejercicio 3.4.....	<u>Pág. 32</u>
Ayudas para el Ejercicio 4.0.....	<u>Pág. 32</u>
Ayudas para el Ejercicio 5.....	<u>Pág. 32</u>
Ayudas para el Ejercicio 6.1.....	<u>Pág. 33</u>
Ayudas para el Ejercicio 6.2.....	<u>Pág. 33</u>
Ayudas para el Ejercicio 6.3.....	<u>Pág. 33</u>



Ayudas para el Ejercicio 1



Pista: Vean Guía en numeral 4.1



Ayudas para el Ejercicio 2.1



Pista: Construyan un algoritmo voraz para resolver el problema



Ayudas para el Ejercicio 3.2



Pista: Lean <http://bit.ly/1ROGW0X>



Ayudas para el Ejercicio 3.4



Pista: Vean Guía en numeral 4.11



Ayudas para el Ejercicio 4.0



Pista: Vean Guía en numeral 4.18



Ayudas para el Ejercicio 5



Pista 1: Para que hagan el mapa conceptual se recomiendan herramientas como las que encuentran en <https://cacoo.com/> o <https://bit.ly/2QM9D3M>

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ayudas para el Ejercicio 6.1



Pista: Vean *Guía en numeral 4.21*

Ayudas para el Ejercicio 6.2



Pista: Vean *Guía en numeral 4.23*

Ayudas para el Ejercicio 6.3



Pista: Vean *Guía en numeral 4.22*

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez

Teléfono: (+57) (4) 261 95 00 Ext. 9473

Correo: mtorobe@eafit.edu.co

Oficina: 19- 627

Agenden una cita dando clic en la pestaña

-*Semana*- de <http://bit.ly/2gzVg10>