**Laboratorio Nro. 1  
Recursión**

**Objetivos:**

1. Identificar el caso base y el caso general de un problema definido recursivamente
2. Usar ecuaciones de recurrencia para determinar la complejidad en tiempo y espacio y definidos de forma recursiva
3. Usar la notación O para encontrar formalmente la complejidad asintótica en espacio y memoria de algoritmos

**Consideraciones iniciales**

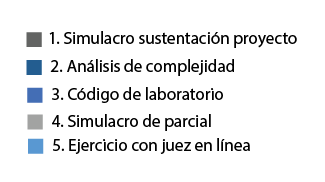
|  |  |
| --- | --- |
| guia-01**Leer la Guía** | Antes de comenzar a resolver el presente laboratorio, leer la ***“Guía Metodológica para la realización y entrega de laboratorios de Estructura de Datos y Algoritmos”*** que les orientará sobre los requisitos de entrega para este y todos los laboratorios, las rúbricas de calificación, el desarrollo de procedimientos, entre otros aspectos importantes. |
| **Registrar Reclamos**  boton-de-preguntas-frecuentes | En caso de tener **algún comentario** sobre la nota recibida en este u otro laboratorio, pueden **enviarlo** a través de [**http://bit.ly/2g4TTKf**](http://bit.ly/2g4TTKf), el cual será atendido en la menor brevedad posible. |
| **Traducción de Ejercicios**  Traducción-01 | En el GitHub del docente, encontrarán la traducción al español de los enunciados de los Ejercicios en Línea. |
| **Visualización de Calificaciones**  **examen-01** | A través de ***Eafit Interactiva*** encontrarán **un enlace** que les permitirá **ver un registro de las calificaciones** que **emite el docente** para cada taller de laboratorio y según las rubricas expuestas. ***Véase sección 3, numeral 3.7.*** |
| **GitHub** | Crear un repositorio en su cuenta de GitHub con el nombre st0245. **2.** Crear una carpeta dentro de ese repositorio con el nombre laboratorios. **3.** Dentro de la carpeta laboratorios, crear una carpeta con nombre lab01. **4.** Dentro de la carpeta lab01, crear tres carpetas: informe, codigo y ejercicioEnLinea. **5.**Subir el informe pdf a la carpeta infome, el código del ejercicio 1 a la carpeta codigo y el código del ejercicio en línea a la carpeta ejercicioEnLinea. Así:  st0245  laboratorios  lab01  informe  codigo  ejercicioEnLinea  lab02  ... |

**Intercambio de archivos**

Los archivos que **ustedes deben entregar** al docente son: **un archivo PDF** con el informe de laboratorio usando la plantilla definida, y **dos códigos**, uno con la solución al numeral 1 y otro al numeral 2 del presente. Todo lo anterior se entrega en **GitHub**.



**Porcentajes y criterios de evaluación para el laboratorio**

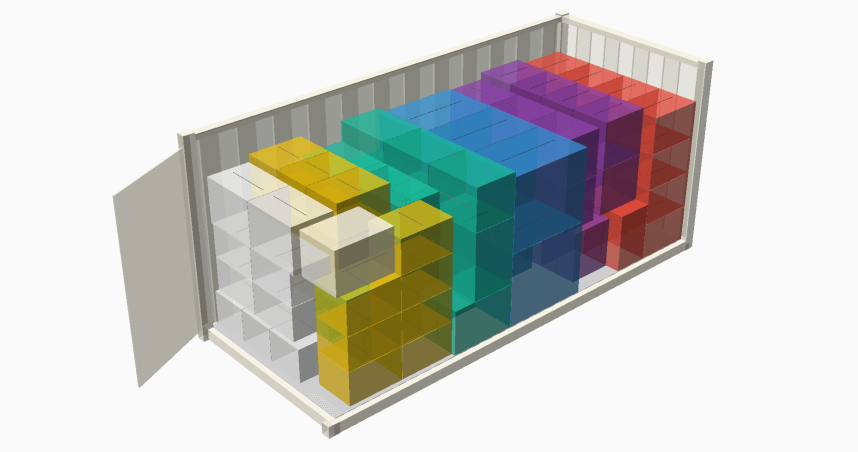


**Simulacro de Proyecto**

1. **Códigos para entregar en GitHub en la carpeta codigo:**

|  |  |
| --- | --- |
| ***Vida  Real-01*** | **En la vida real, la documentación de software hace parte de muchos estándares de calidad como CMMI e ISO/IEC 9126** |

|  |  |  |  |
| --- | --- | --- | --- |
| **guia-01** | Véase Guía ***en Sección 3, numeral 3.4*** |  | *Código de laboratorio en* ***GitHub.*** Véase Guía en **Sección 4, numeral 4.24** |
| **No HTML** | *Es opcional entregar documentación. Si lo hace, utilice* ***javadoc*** *o equivalente. No suba el HTML a GitHub.* | **no rar** | ***No******se reciben*** *archivos en* ***.RAR*** *ni en* ***.ZIP*** |

**1** Puerto Antioquia en Urabá es un proyecto en curso que traerá mucho desarrollo a la región. Su construcción está prevista para terminar a finales de 2020.

Por sus características y ubicación geográfica, Puerto Antioquia podría convertirse en el centro logístico de mayor importancia del país. Un problema que tendrán, una vez inicie la operación, es encontrar la forma óptima de empacar la mercancía en los contenedores.

Una versión simplificada de este problema es una versión en dos dimensiones: Hay un tablero de 2×*n* cuadrados y usted necesita saber de cuantas maneras se puede llenar el tablero usando rectángulos de 1×2.

**Implementen un algoritmo recursivo para encontrar de cuántas formas se puede llenar un rectángulo de 2xn cm2 con rectángulos de 1x2 cm².**

**[Ejercicio opcional] Implementen una interfaz gráfica para visualizar la respuesta del algoritmo anterior utilizando la librería JavaFX, Graphics, print o equivalente.**

**2) Simulacro de Maratón de Programación sin documentación HTML en GitHub**

|  |  |  |  |
| --- | --- | --- | --- |
| **guia-01** | Véase Guía en ***Sección 3, numeral 3.3*** | **No HTML** | *No se requiere documentación para los ejercicios en línea* |
| **C:\Users\Luisa Alzate\AppData\Local\Microsoft\Windows\INetCache\Content.Word\java.png** | *Entregar un archivo en* ***.JAVA*** | **no pdf** | ***No******se reciben*** *archivos en* ***.PDF ni .TXT*** |
| CodingBat-01 | ***Resolver*** *los problemas de* ***CodingBat*** *usando* ***Recursión*** |  | *Código del ejercicio en línea en* ***GitHub.*** Véase Guía en **Sección 4, numeral 4.24** |

|  |  |
| --- | --- |
| alerta-simbolo-dibujado-a-mano | **NOTA:** Recuerden que, **si toman la respuesta de alguna fuente**, deben **referenciar** según el **tipo de cita** correspondiente**.** Véase *Guía en* ***Sección 4, numerales 4.16 y 4.17*** |

**2.1 Resolver al menos 5 ejercicios del nivel *Recursion 1* de CodingBat**: [**http://codingbat.com/java/Recursion-1**](http://codingbat.com/java/Recursion-1)

**2.2 Resolver al menos 5 ejercicios del nivel *Recursion 2* de la página CodingBat:** [**http://codingbat.com/java/Recursion-2**](http://codingbat.com/java/Recursion-2)

|  |  |
| --- | --- |
| alerta-simbolo-dibujado-a-mano | **NOTA:** No está permitido el ejercicio ***GroupSum****.* |

**3) Simulacro de preguntas de sustentación de Proyecto**

|  |  |  |  |
| --- | --- | --- | --- |
| **guia-01** | Véase Guía en ***Sección 3, Numeral 3.4*** | **C:\Users\Luisa Alzate\AppData\Local\Microsoft\Windows\INetCache\Content.Word\pdf-simbolo-de-formato-de-archivo.png** | *Entregar informe de laboratorio*en **PDF** |
| **C:\Luisa Alzate\2. Laboral\Informal\Mauricio Toro\Iconos y Logos\papel.png** | *Usen la* ***plantilla*** *para responder laboratorios* | ***ICONTECr*** | ***No*** *apliquen* ***Normas Icontec*** *para esto* |

**SOBRE EL EJERCICIO 1**

**3.1 Calculen la complejidad asintótica para el peor de los casos del ejercicio 1**

**3.3 Tome tiempos para 20 tamaños del problema diferentes, genere una gráfica y analice los resultados.** Estime, ¿cuánto tiempo se demorará este algoritmo en calcular las formas que existen de llenar un rectángulo de 50x2 cm² con rectángulos de 1x2 cm² ?

**3.3** ¿La complejidad de este algoritmo hace que sea viable utilizarlo en Puerto Antioquia, en 2020, con contenedores que miden miles de centímetros?

**SOBRE EL EJERCICIO 2**

**3.4 Expliquen con sus propias palabras cómo funciona el ejercicio *GroupSum5***

|  |  |
| --- | --- |
| alerta-simbolo-dibujado-a-mano | **NOTA:** Recuerden que deben explicar su implementación en el informe PDF |

**3.5Calculen la complejidad de los Ejercicios en Línea de los numerales 2.1 y 2.2 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 ejercicio anterior. Ejemplos de su respuesta: “n es el número del 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’.**

**4) Simulacro de Parcial en el informe PDF**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Para este simulacro, agreguen ***sus respuestas* en el informe PDF.**  Resuelva, como mínimo, los ejercicios marcados con color rojo. |  | *El* ***día del Parcial no tendrán computador, JAVA o acceso a internet.*** |

|  |  |  |
| --- | --- | --- |
|  | clave | **PISTA 1:** *Véase* ***Guía en Sección 4, Numeral 4.18*** “Respuestas del Quiz” |
| **1.** Pepito escribió un algoritmo que, dado un arreglo de enteros, decide si es posible escoger un subconjunto de esos enteros, de tal forma que la suma de los elementos de ese subconjunto sea igual a target. El parámetro start funciona como un contador y representa un índice en el arreglo de números nums.  01 public boolean SumaGrupo(int start,  int[] nums, int target) {  02 if (start >= nums.length) return target == 0;  03 return SumaGrupo(start + 1, nums,  target - nums[start])  04 || SumaGrupo(\_\_\_\_,\_\_\_,\_\_\_);  05 }  ¿Qué parámetros colocarían en el llamado recursivo de la línea 4 para que el programa funcione?  \_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **2.** Pepito escribió el siguiente código usando recursión  private int b(int[ ] a, int x, int low, int high) {  if (low > high) return -1;  int mid = (low + high)/2;  if (a[mid] == x) return mid;  else if (a[mid] < x)  return b(a, x, mid+1, high);  else  return b(a, x, low, mid-1);  }  ¿Cuál ecuación de recurrencia describe el comportamiento del algoritmo anterior para el peor de los casos?   1. *T*(*n*)=*T*(*n*/2)+*C* 2. *T*(*n*)=2.*T*(*n*/2)+*C* 3. *T*(*n*)=2.*T*(*n*/2)+*Cn* 4. *T*(*n*)=*T*(*n*−1)+*C*   **3.** Dayla y Kefo están aquí de nuevo. En esta vez han traído un juego muy interesante, en el cual Kefo, en primer lugar, escoge un numero *n* (1≤*n*≤20) y, en segundo lugar, escoge tres números *a*,*b* y *c* (1≤*a*≤9,1≤*b*≤9,1≤*c*≤9).  Después, Kefo le entrega estos números a Dayla y Dayla le tiene que **decir a Kefo la cantidad máxima de números tomados de a, b y c (se puede tomar un número más de una vez) que al sumarlos dan el valor n**.  Como un ejemplo, si Kefo escoge *n*=14 y *a*=3,*b*=2,*c*=7. ¿Qué posibilidades hay de obtener 14 con *a*,*b* y *c*?   |  |  | | --- | --- | | 7+7=14 | cantidad es 2 | | 7+3+2+2=14 | cantidad es 4 | | 3+3+3+3+2=14 | cantidad es 5 | | … |  | | 2+2+2+2+2+2+2=14 | cantidad es 7 |   La cantidad máxima de números es 7. Esta sería la respuesta que da Dayla a Kefo. Como Dayla es muy astuta, ha diseñado un algoritmo para determinar la cantidad máxima de números y quiere que le ayudes a terminar su código.  Asuman que hay al menos una forma de sumar *n* usando los números *a*, *b* y *c* en diferentes cantidades, incluso si algunos de los números se suman 0 veces como sucede en el ejemplo anterior.  1 int solucionar (int n, int a, int b, int c) {  2 if (n == 0 || (n < a && n < b && n < c))  3 return 0;  4 int res = solucionar(\_\_\_\_\_\_\_\_\_\_\_\_) + 1;  5 res = Math.max(\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_);  6 res = Math.max(\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_);  7 return res; }  **3.1** Completen el espacio de la línea 04  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **3.2** Completen los espacios de la línea 05  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **3.3** Completen los espacios de la línea 06  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **4.** ¿Qué calcula el algoritmo desconocido y cuál es la complejidad asintótica en el peor de los casos del algoritmo desconocido?    01 public int desconocido(int[] a){  02 return aux(a, a.length-1); }  03  04 public int aux(int[] a, int n){  05 if(n < 1) return a[n];  06 else return a[n] + aux(a, n-1); }   1. La suma de los elementos del arreglo a y es 2. Ordena el arreglo a y es *O*(*n*.*log* *n*) 3. La suma de los elementos del arreglo a y es *O*(1) 4. El máximo valor de un arreglo a y es *O*(*n*) 5. La suma de los elementos del arreglo a y es *O*(*n*)   **5.** Hay un tablero de 2×*n* cuadrados y usted necesita saber de cuantas maneras se puede llenar el tablero usando rectángulos de 1×2. Se ha propuesto el siguiente algoritmo recursivo  1 public int formas(int n){  2 if(n <= 2) \_\_\_\_\_\_\_\_\_\_;  3 return formas(\_\_\_\_) +  4 formas(\_\_\_\_);  5 }  **5.1** Completen las líneas faltantes.  Linea 2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Linea 3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Linea 4. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **5.2** ¿Cuántas instrucciones ejecuta el algoritmo en el peor de los casos?  **a.** *T*(*n*)=*T*(*n*−1)+*C*  **b.** *T*(*n*)=*T*(*n*−1)+*T*(*n*−2)+*C*  **c.** *T*(*n*)=*T*(*n*/2)+*C*  **d.** *T*(*n*)=*T*(*n*+1)+*C*  **6.** Alek y Krish están jugando *Número*. Número es un juego en el que un jugador 1, entrega un numero *n* (1<=*n*<=10100) a un jugador 2 y el jugador 2 debe determinar la suma de todos los dígitos de *n*, exceptuando el caso en el que hay dos dígitos adyacentes (es decir, contiguos, seguidos) que son iguales.  Si hay dos dígitos adyacentes, no se suma ninguno de los dos números adyacentes. Entre Alek y Krish escribieron un código para hacer esto más rápido, pero se ha borrado una parte. ¿Podrían ayudarles a reconstruir el código a Alek y Krish?    1 public int suma(String n) {  2 return sumaAux(n, 0);  3 }  4  5 private int sumaAux(String n, int i){  6 if (i >= n.length()) {  7 return 0;  8 }  9 if(i + 1 < n.length() &&  n.charAt(i) == n.charAt(i + 1)){  10 return \_\_\_\_\_\_;  11 }  12 return (n.charAt(i) - '0') + \_\_\_\_\_;  13 }  La operación n.charAt(i) - ’0’ convierte un caracter en su equivalente en entero, por ejemplo, el caracter ’1’ lo transforma en el número 1.    **6.1** Completen la línea 10.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **6.2** Completen la línea 12.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **7)** Dado un conjunto *S* de *n* elementos se quiere determinar si existe un subconjunto *R* de *S* tal que la suma de los elementos de *R* es igual a *t*, con la condición que, si se toma un elemento par, el siguiente elemento no puede estar en *R*.    1 public boolean comb(int[] S, int i, int t){  2 if(i >= S.length){  3 return t == 0;  4 }  5 //par  6 if(S[i] % 2 == 0){  7 return comb(S, i + 2, t - S[i])  8 || comb(S, i + 1, t); }  9 return comb(\_\_\_\_\_\_\_\_\_\_\_) ||  10 comb(\_\_\_\_\_\_\_\_\_\_\_\_);  11 }  Al anterior código le faltan algunas líneas las cuales deben completar  **7.1** Línea 9  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **7.2** Línea 10  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **8)** El pequeño Polka fue a la tienda a comprar algunas cosas para su madre. El pequeño Polka pagó cierta cantidad de dinero y el tendero debe devolverle exactamente *K* pesos. El tendero tiene exactamente *n* diferentes tipos de monedas en su tienda, cada una de ellas con un valor vi. Como el tendero no es muy astuto en asuntos de matemáticas, te ha pedido el favor de decirle de cuantas maneras puede él devolverle tal cantidad a Polka. Por favor ayúdanos a completar el siguiente código.  **Ejemplo:**  Sea *n* = |*v*| = 3, *v* = {3,4,1}, *K*=7.  Las formas posibles de devolver 7 dólares serían 5:  1. 1 + 1 + 1 + 1 + 1 + 1 + 1  2. 3 + 1 + 1 + 1  3. 3 + 3 + 1  4. 3 + 4  5. 4 + 1 + 1 + 1  01 int cuantas(int K, int[] v, int n){  02 if(K == 0){  03 return 1;  04 }  05 boolean imposible;  06 imposible = n <= 0 && K >= 1;  07 imposible = imposible || K < 0;  08 if(imposible){  09 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  10 }  11 int ni = cuantas(K, v, n - 1);  12 int nj = cuantas(K - v[n-1], v, n);  13 int suma = \_\_\_\_\_\_\_\_\_\_\_\_\_;  14 return suma;  15 }  **8.1.** Complete Línea 9 ....................  **8.2.** Complete Línea 13 ....................  **9.** Considere el siguiente programa. ¿Cuál es la salida generada por *fun*(11,5)? Como un ejemplo: *fun*(10,3)=20.  int fun(int n, int m){  if(n % m == 2) return n;  return fun(n + m, n - m);  }  (a) 11  (b) 5  (c) 22  (d) 2  **10.** Considere el siguiente programa. ¿Cuál es la salida para *fun*(1,4)? Como un ejemplo: *fun*(1,2)=4.  int fun(int m,int n){  if(m==0){  return (n+1);  }  if(m>0 && n==0){  return fun(m-1,1);  }  int a=fun(m,n-1);  return fun(m-1,a);  }  (a) 4  (b) 6  (c) 5  *(d) 12* | | |

1. **[Ejercicio Opcional] Lectura recomendada**

|  |  |
| --- | --- |
| ***Vida  Real-01*** | **"Quienes se preparan para el ejercicio de una profesión requieren la adquisición de competencias que necesariamente 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**](http://bit.ly/2gJKzJD) |

|  |  |
| --- | --- |
| guia-01 | *Véase Guía en* ***Sección 3, numeral 3.5 y 4.20*** *de la Guía Metodológica, “Lectura recomendada” y “Ejemplo para realización de actividades de las Lecturas Recomendadas*”, respectivamente |

Posterior a la lectura del texto ***“Narasimha Karumanchi, Data Structures and Algorithms made easy in Java, (2nd edition), 2011. Chapter 3: Recursion and Backtracking”*** realicen las siguientes actividades que les permitirán sumar puntos adicionales:

1. Escriban un resumen de la lectura que tenga una longitud de 100 a 150 palabras
2. Hagan un mapa conceptual que destaque los principales elementos teóricos.

|  |  |
| --- | --- |
| alerta-simbolo-dibujado-a-mano | ***NOTA :*** Estas respuestas también deben incluirlas en el informe PDF |

1. **[Ejercicio Opcional] Trabajo en Equipo y Progreso Gradual**

|  |  |
| --- | --- |
| ***Vida  Real-01*** | **El trabajo en equipo es una exigencia actual del mercado. "Mientras algunos medios retratan la programación como un trabajo solitario, la realidad es que requiere de mucha comunicación y trabajo con otros. Si trabajas para una compañía, serás parte de un equipo de desarrollo y esperarán que te comuniques y trabajes bien con otras personas”**  Tomado de[**http://bit.ly/1B6hUDp**](http://bit.ly/1B6hUDp) |

|  |  |
| --- | --- |
| guia-01 | *Véase Guía en* ***Sección 3, numeral 3.6*** *y* ***Sección 4, numerales 4.21, 4.22*** *y* ***4.23*** *de la Guía Metodológica* |

1. Entreguen copia de todas las actas de reunión usando el *tablero Kanban*, con fecha, hora e integrantes que participaron
2. Entreguen el reporte de *git* con los cambios en el código y quién hizo cada cambio, con fecha, hora e integrantes que participaron
3. Entreguen el reporte de cambios del informe de laboratorio que se genera *Google docs* o herramientas similares

|  |  |
| --- | --- |
| alerta-simbolo-dibujado-a-mano | ***NOTA:*** Estas respuestas también deben incluirlas en el informe PDF |

1. **[Ejercicio Opcional] Laboratorio en inglés:**

|  |  |
| --- | --- |
| ***Vida  Real-01*** | **El inglés es un idioma muy 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 y es sólo un resumen de la información original.**  **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 traducido al inglés. Utilicen la plantilla dispuesta en este idioma para el laboratorio.**

**Ayudas para resolver los ejercicios**

|  |  |
| --- | --- |
| **Ayudas para el Ejercicio 1.1…………….** | [**Pág. 1**](#Ejer11)**6** |
| **Ayudas para el Ejercicio 2.1…………….** | [**Pág. 1**](#Ejer21)**6** |
| **Ayudas para el Ejercicio 2.2…………….** | [**Pág. 1**](#Ejer22)**7** |
| **Ayudas para el Ejercicio 2.4…………….** | [**Pág. 17**](#Ejer24) |
| **Ayudas para el Ejercicio 3.2…………….** | [**Pág. 18**](#Ejer32) |
| **Ayudas para el Ejercicio 3.4…………….** | [**Pág. 18**](#Ejer34) |
| **Ayudas para el Ejercicio 3.5…………….** | [**Pág. 19**](#Ejer35) |
| **Ayudas para el Ejercicio 3.6…………….** | [**Pág. 19**](#Ejer36) |
| **Ayudas para el Ejercicio 4…………….** | [**Pág. 19**](#Ejer4) |
| **Ayudas para el Ejercicio 5a…………….** | [**Pág. 19**](#Ejer5a) |
| **Ayudas para el Ejercicio 5b…………….** | [**Pág. 20**](#Ejer5b) |
| **Ayudas para el Ejercicio 6a…………….** | [**Pág. 20**](#Ejer6a) |
| **Ayudas para el Ejercicio 6b…………….** | [**Pág. 20**](#Ejer6b) |
| **Ayudas para el Ejercicio 6c…………….** | [**Pág. 20**](#Ejer6c) |

**Ayudas para el** **Ejercicio 1**

****

**Para todos los ejercicios**

****

**Pista 1:** Procedimiento sugerido

1. Implementen el algoritmo
2. Identifiquen qué representa el tamaño del problema para cada algoritmo
3. Identifiquen valores apropiados para tamaños del problema
4. Tomen los tiempos para los anteriores algoritmos con 20 tamaños del problema diferentes.
5. Hagan una gráfica en Excel para cada algoritmo y pasarla a Word luego
6. Entreguen el archivo de Word pasado a PDF.



**Pista 2:** Vean la sección 4.2 de la *Guía de Laboratorios* muestra cómo generar arreglos con números aleatorios

**Pista 3:** Vean la sección 4.3 de la *Guía de Laboratorios* muestra cómo aumentar el *heap* en Java y la sección 4.4 muestra cómo aumentar el tamaño de la pila.

**Pista 4:** Vean la sección 4.7 de la *Guía de Laboratorios* muestra cómo tomar el tiempo en Java

**Pista 5:** Vean la sección 4.6 de la *Guía de Laboratorios* muestra cómo cambiar la escala de una gráfica en Excel a escala logarítmica.

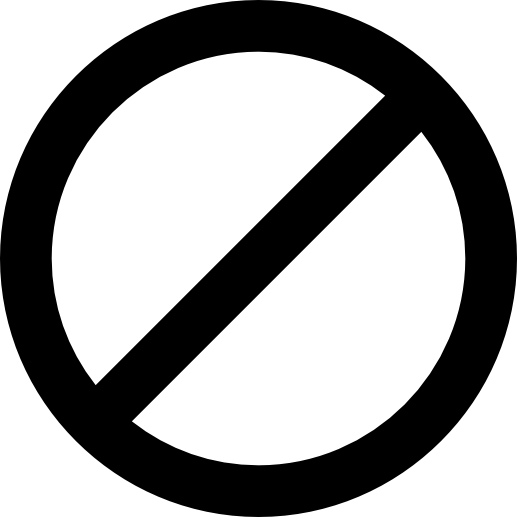
**Ayudas para el Ejercicio** **2.1**

|  |
| --- |
| **Error común** |
| n-ary-trees5 |

**Ayudas para el** **Ejercicio 2.2**

|  |  |
| --- | --- |
| clave | **PISTA 1:** El algoritmo *GroupSum* falla porque al llamarse recursivamente con el parámetro start se queda en una recursión infinita |

|  |  |
| --- | --- |
| clave | **PISTA 2:** El algoritmo *GroupSum* falla porque al llamarse recursivamente con el parámetro start-1 se sale del arreglo cuando start = 0. |
| clave | **PISTA 3:** El algoritmo *GroupSum* falla porque al llamarse recursivamente con el parámetro start se sale del arreglo cuando start = length-1 |

A continuación encontrarán algunos errores comunes que pueden ser aplicados con los ejercicios

public boolean groupSum(int start, int[] nums, int target) {

  if (start >= nums.length) return target == 0;

  return groupSum(start+1, nums, target - nums[start])

        ||groupSum(start, nums, target );

}

public boolean groupSum(int start, int[] nums, int target) {

  if (start >= nums.length) return target == 0;

  return groupSum(start+1, nums, target - nums[start])

        ||groupSum(start-1, nums, target );

}

public boolean groupSum(int start, int[] nums, int target) {

  if (start >= nums.length) return target == 0;

  return groupSum(start+1, nums, target - nums[start])

        ||groupSum(start+1, nums, target - nums[start - 1] );

**Ayudas para el** **Ejercicio 3.4**

|  |  |
| --- | --- |
| clave | **PISTA:** Véase***Guía en Sección 4, numeral 4.11*** *“Cómo escribir la complejidad de un ejercicio en línea”* |
| clave | **PISTA 2:** Véase***Guía en Sección 4, numeral 4.19 “****Ejemplos para calcular la complejidad de un ejercicio de CodingBat”* |

|  |  |  |
| --- | --- | --- |
| **Errores Comunes** | | |
| groupsum | 2llamados | On22n |

**Ayudas para el** **Ejercicio 3.5**

|  |  |
| --- | --- |
| clave | **PISTA 1:** *Véase* ***Guía sección 4, numeral 4.6*** *“Cómo usar la escalala logarítmica en Microsoft Excel”.* |
| clave | **PISTA 2:** Sitodos los tiempos de un algoritmo dan más de 5 minutos, realice otra tabla, para ese algoritmo, tomando tiempos para arreglos de e tamaño 1000, 10000 y 100000. |

**Ayudas para el** **Ejercicio 1**

|  |  |
| --- | --- |
| clave |  |
| alerta-simbolo-dibujado-a-mano | **PISTA 2:** Consideren llenar un tablero de 2×n. Si le quitamos la primera baldosa tenemos un tablero de 2×(n–1) (usando recursión). Si le quitamos 2 baldosas queda un tablero de 2×(n–2). Hagan el dibujo. ¿Se le pueden quitar 3 baldosas, o con lo anterior ya puedo formar el de 2×(n–3)? |

**Ayudas para el Ejercicio** **5a**

|  |  |
| --- | --- |
| clave | **PISTA 1:** En el siguiente enlace, unos consejos de cómo hacer un buen resumen[**http://bit.ly/2knU3Pv**](http://bit.ly/2knU3Pv) |
| clave | **PISTA 2:** [**Aquí**](http://bit.ly/28JXqIG) le explican cómo contar el número de palabras en Microsoft Word |

**Ayudas para el** **Ejercicio 5b**

|  |  |
| --- | --- |
| clave | **PISTA 1:** Para que hagan el mapa conceptual se recomiendan herramientas como las que encuentran en [**https://cacoo.com/**](https://cacoo.com/)o **[https://www.mindmup.com/#m:new-a-1437527273469](https://www.mindmup.com/" \l "m:new-a-1437527273469)** |

**Ayudas para el** **Ejercicio 6a**

|  |  |
| --- | --- |
| clave | **PISTA 1:** *Véase* ***Guía en Sección 4, Numeral 4.21*** “*Ejemplo de cómo hacer actas de trabajo en equipo usando Tablero Kanban”* |

**Ayudas para el** **Ejercicio 6b**

|  |  |
| --- | --- |
| clave | **PISTA 1: *Véase*** *Guía en Sección 4, Numeral 4.23*“**Cómo generar el historial de cambios en el código de un repositorio que está en svn”** |

**Ayudas para el** **Ejercicio 6c**

|  |  |
| --- | --- |
| clave | **PISTA 1: *Véase*** *Guía en Sección 4, Numeral 4.22* “***Cómo ver el historial de revisión de un archivo en Google Docs*”** |