Trabajo Práctico 0

Repaso Archivos, Scanner y Wrappers

Análisis de Algoritmos 2023

11 de agosto de 2023

Alumnos:

- Repetto Francisco Manuel (FAI -2548)
- Rivera Malena Anais (FAI-2516)

Respuestas Teóricas:

Ejercicio 2; Repaso de Algoritmia

1. Realiza detenidamente una traza al siguiente programa y muestra cuál sería la salida por pantalla:

```
ALGORITMO ej1
      VARIABLES
      suma i,j: ENTERO
      PARA i <-- 1 HASTA 4 HACER
            PARA j <--3 HASTA 0 PASO -1 HACER
                  suma <-- i*10+j
                  escribir(suma)
            FIN PARA
      FIN PARA
FIN ALGORITMO
```

TRAZA

i = 1 -> 2 -> 3 -> 4
j = 3 -> 2 -> 1 -> 0
suma = 13 -> 22 -> 31 -> 40

2. ¿Que imprime el siguiente programa? ´

```
1 class Ejercicio {
2     public static void main (String [] args){
3           char [] matriz ={'e','u','o','i','a'};
4           metodo(matriz);
5           for (int i = 0;i<matriz.length;i++){
6                 System.out.print(matriz[i];
7           }
8     }
9
2
10    public static void metodo (char [] vocales){
11          char aux;
12
13          for (int i=1;i<vocales.length;i++){
14                if (vocales[i-1]>vocales[i]){
15                      aux=vocales[i-1];
16                      vocales[i-1]=vocales[i];
17                      vocales[i]=aux;
18                }
19          }
20    }
21 }
```

Traza:

MAIN:

matriz = {'e','u','o','i','a'} -> {'e','o','i','a','u'}

i = 0 -> 1 -> 2 -> 3 -> 4

MÉTODO:

vocales = {'e','u','o','i','a'} -> {'e','o','u','i','a'} -> {'e','o','i','u','a'} -> {'e','o','i','a','u'}

i = 1 -> 2 -> 3 -> 4

aux = u -> u ->u

Imprime:

e

o

i

a

u

Ejercicio 5: Suponiendo n ≤ 1000000 y un usuario que siga de forma optima la lógica del juego y que quiera dar la menor cantidad de pasos hasta adivinar el numero: ¿Cual es el máximo número de intentos que puede necesitar el jugador hasta encontrar un número dentro del intervalo?

Respuesta:
La forma más óptima es dividir al número en 2 y dependiendo de la respuesta, dividir por 2. Por ejemplo: 1.000.000 preguntas si n es 500.000, si es menor entonces preguntas 250.000. Por lo tanto, el máximo número de intentos que puede necesitar el jugador hasta encontrar un número es de 20 pasos

Ejercicio 6: Liste y describa claramente los algoritmos para la resolucion del problema de busqueda que conoce. (tip: recordar distintas implementaciones de interfaz TablaDeBusqueda)

Respuesta:
Arreglos:
Búsqueda Secuencial: La búsqueda secuencial es un método de búsqueda en el que se recorre un conjunto de elementos uno por uno para encontrar un valor específico.
Búsqueda Binaria: La búsqueda binaria es un algoritmo de búsqueda más eficiente que solo se puede aplicar en listas ordenadas. Funciona dividiendo repetidamente la lista en mitades y comparando el elemento buscado con el elemento en el medio. Aquí está el proceso general de búsqueda binaria

Recorridos de Arboles
Búsqueda en Anchura : es un algoritmo de búsqueda que explora un grafo en capas. Comienza desde un nodo inicial y explora todos los nodos vecinos antes de avanzar a los nodos en la siguiente capa.
Búsqueda en Profundidad: algoritmo de búsqueda que explora un grafo tan profundamente como sea posible antes de retroceder. Comienza desde un nodo inicial y sigue un camino hasta llegar a un nodo sin nodos vecinos sin explorar, momento en el cual retrocede y explora otros caminos no explorados

Tabla de búsqueda con Hash
La Tabla Hash tiene una eficiencia en las operaciones de búsqueda, inserción y eliminación de orden constante, O(1), cuando se cuenta con una función hash adecuada para el tipo de datos que se almacena como clave. Luego, se deberá modificar la estructura de Tabla Hash de manera adecuada para permitir almacenar la información adicional de cada clave

Ejercicio 7: Al ordenar una lista de números enteros aplicando el algoritmo quicksort, como pivote se elige el primer elemento de la lista. ¿Qué pasaría si se selecciona otro pivote?

Respuesta:
Al seleccionar diferentes elementos como pivote en el algoritmo de ordenamiento QuickSort, se pueden tener resultados variados en términos de eficiencia y número de operaciones realizadas. La elección del pivote afecta el rendimiento del algoritmo, ya que determina cómo se divide la lista en subconjuntos y cuántas comparaciones y movimientos se requieren.

Si seleccionas el primer elemento de la lista como pivote en QuickSort, es posible que enfrentes el peor caso de rendimiento cuando la lista ya esté ordenada o esté casi ordenada. Esto sucede porque en cada iteración del algoritmo, el pivote (primer elemento) se comparará con los elementos restantes, y en un escenario ya ordenado, se producirían particiones muy desequilibradas.

Ejercicio 9: Se leen dos listas de números enteros, A y B de 100 y 60 elementos, respectivamente. Se desea resolver mediante procedimientos las siguientes tareas:

a) Ordenar aplicando un metodo de ordenacion distinto a cada una de las listas A y B

Aclaración:
Elegimos los métodos de ordenamiento Quick-sort y Merge-sort porque son los eficientes que conocemos junto a Heap-sort

b) Crear una lista C a partir de la mezcla de las listas A y B ya ordenadas.

Aclaración:
Para crear esta lista usamos Merge-sort entre ambas listas