**1)resumen**

El dia de hoy :

a)Se realizó ejercicio de simular un cajero automático

b)Estudiamos la forma de trabajar los arreglos

c)Se dedicó un tiempo a ver metodos de ordenamiento y búsqueda

**2)reflexión**

Hoy no podía ejecutar mi ejercicio y esto estaba pasando porque java se marea a veces con la instrucción .nextline ya que estaba erróneamente tomando todos los caracteres que se ingresan por pantalla lo que genera un error.

Luego del apoyo del profesor pude resolver el ejercicio.

**3) Estudio Extra**

Se investigó cómo funciona el método de ordenamiento quickSort.

Es el método de ordenamiento más eficiente de todos.

El método burbuja funciona pero es muy lento.

Se adjuntan ambos códigos

### **Ordenamiento**

**Objetivos: Mostrar distintos algoritmos de ordenamiento, comparando su eficiencia en tiempo de ejecución tanto desde el punto de vista del factor constante (por ejemplo este algoritmo es 3 veces más rápido que este otro) y del orden de magnitud (este algoritmo es más eficiente porque es O(log n)).**

**Temas:**

* **Ordenamiento.**
* **Ordenamiento por selección y reemplazo.**
* **Análisis del tiempo de ordenamiento por selección y reemplazo.**
* **Algoritmo de la burbuja.**
* **Quicksort**
* **Quicksort con arreglos nativos.**

#### **Ordenamiento**

**La búsqueda binaria funciona correctamente sólo si el arreglo está ordenado según el mismo criterio con el que se realiza la búsqueda. En las clases que vienen a continuación veremos distintos algoritmos para ordenar los elementos de un arreglo.**

**El objetivo de estudiar los algoritmos de ordenamiento es doble. Primero, porque permite ejemplificar la importancia del estudio de la eficiencia de los algoritmos, mostrando que no es intuitivo predecir cuanto tiempo de ejecución toman.**

**Y segundo, porque los computadores pasan típicamente 50% de su tiempo ordenando (de ahí que los españoles y franceses prefieren usar la palabra ordenador en vez de computador). Parece lógico entonces conocer cuáles son los algortimos que se usan para ordenar.**

**Hace un tiempo atrás era válido un tercer argumento: si en algún momento un programador necesitaba ordenar un arreglo, tenía que programarlo. A partir de 1999, la versión 1.2 de Java incorpora una clase de biblioteca para ordenar datos de manera eficiente.**