Nombre del alumno: Johanne Joaquín Arriaga Díaz

Nombre del profesor: Icel Bernardo Lepe Arriaga.

Nombre del trabajo: Cuadro sinóptico de unidad IV.

Materia: Algoritmos y estructuras de datos.

Grado: Quinto cuatrimestre

Grupo: ISC13SDC0119-F

Frontera Comalapa, Chiapas a 10 de Abril de 2021

UNIDAD IV
ORDENAMIENTO

- Relaciones de orden débiles.
  - Una relación de orden es para definir conjuntos o correspondencias, y la relación de ordenamiento puede ser elegida por conveniencia. Con un algoritmo podemos ordenar de manera automática
    - Asignatura de las relaciones de orden. Predicados binarios.

La expresión:	Usando: $\lhd$	Usando: $\trianglelefteq$
$a \lhd b$	$a \lhd b$	$!\ (b \trianglelefteq a)$
$a \trianglelefteq b$	$!\ (b \lhd a)$	$a \trianglelefteq b$
$a \rhd b$	$b \lhd a$	$!\ a \trianglelefteq b$
$a \trianglerighteq b$	$!\ a \lhd b$	$b \trianglerighteq a$
$\mathrm{cmp}(a,b)$	$(b \lhd a) - (a \lhd b)$	$(b \trianglelefteq a) - (a \trianglelefteq b)$
$a = b$	$!\ (a \lhd b \,\|\|\, b \lhd a)$	$a \trianglelefteq b \,\&\&\, b \trianglelefteq a$
$a \neq b$	$a \lhd b \,\|\|\, b \lhd a$	$(!\ a \trianglelefteq b) \,\|\|\, (!\ b \trianglelefteq a)$

La expresión:	Usando: $\rhd$	Usando: $\trianglerighteq$
$a \lhd b$	$b \rhd a$	$!\ a \trianglerighteq b$
$a \trianglelefteq b$	$!\ (b \rhd a)$	$b \trianglerighteq a$
$a \rhd b$	$a \rhd b$	$!\ b \trianglerighteq a$
$a \trianglerighteq b$	$!\ b \rhd a$	$a \trianglerighteq b$
$\mathrm{cmp}(a,b)$	$(a \rhd b) - (b \rhd a)$	$(a \trianglerighteq b) - (b \trianglerighteq a)$
$a = b$	$!\ (a \rhd b \,\|\|\, b \rhd a)$	$a \trianglerighteq b \,\&\&\, b \trianglerighteq a)$
$a \neq b$	$a \rhd b \,\|\|\, b \rhd a$	$(!\ a \trianglerighteq b) \,\|\|\, (!\ b \trianglerighteq a)$

La expresión:	Usando: $\mathrm{cmp}(\cdot,\cdot)$
$a \lhd b$	$\mathrm{cmp}(a,b) = -1$
$a \trianglelefteq b$	$\mathrm{cmp}(a,b) \leq 0$
$a \rhd b$	$\mathrm{cmp}(a,b) = 1$
$a \trianglerighteq b$	$\mathrm{cmp}(a,b) \geq 0$
$\mathrm{cmp}(a,b)$	$\mathrm{cmp}(a,b)$
$a = b$	$!\ \mathrm{cmp}(a,b)$
$a \neq b$	$\mathrm{cmp}(a,b)$

- Métodos de ordenamiento lentos
  - Llamamos “rápidos” a los métodos de ordenamiento con tiempos de ejecución menores o iguales a O(n log n). Al resto lo llamaremos “lentos”.
    - Método de la burbuja
      - 1) Presentamos las dos funciones con y sin función de comparación.
        2) Son templates sobre el tipo T de los elementos a ordenar.
        3) La que no tiene operador de comparación suele ser un “wrapper” que llama a la primera pasándole como función de comparación less.
        4) Las funciones no reciben un contenedor, sino un rango de iteradores, se puede referenciar a través del operador de dereferenciación *p. Así, donde normalmente pondríamos v[first+j] debemos usar *(first+j).
        5) Las operaciones aritméticas con iteradores son válidas ya que los contenedores son de acceso aleatorio. Las funciones presentadas son sólo válidas para vector<>.
        6) Notar la comparación puede ser con: comp() en vez de operator
        7) Se hara first, como si fuera 0. Es decir consideraremos que se está ordenando el rango [0,n) donde n=last-first
    - El método de inserción
      - En este método también hay un doble lazo (esta es una característica de todos los algoritmos lentos). En el lazo j se elige el menor del rango [j, N) y se intercambia con el elemento en la posición j.
    - El método de selección
      - En este método hay un doble lazo. En el lazo sobre j el rango [0, j) está ordenado e insertamos el elemento j en el rango [0, j), haciendo los desplazamientos necesarios. El lazo sobre k va recorriendo las posiciones desde j hasta 0 para ver donde debe insertarse el elemento que en ese momento está en la posición j.
- Ordenamiento indirecto
  - Si el intercambio de elementos es muy costoso podemos reducir el número de intercambios usando “ordenamiento indirecto”, se basa en ordenar un vector de cursores o punteros a los objetos reales lo cual es mucho más bajo que el intercambio de los objetos mismos.
    - Minimizar la llamada a funciones
      - Si la función de comparación se obtiene por composición, y la función de mapeo es muy costosa. Entonces mejor generar primero un vector auxiliar con los valores de la función de mapeo, ordenarlo y después aplicar la permutación resultante a los elementos reales. De esta forma el número de llamadas a la función de mapeo pasa de ser O(n 2) a O(n).
- El método de ordenamiento rápido, quick-sort
  - Se escoge un elemento del vector v llamado “pivote” y se “particiona” el vector de manera de dejar los elementos ≥ v a la derecha (rango [l, n), donde l es la posición del primer elemento de la partición derecha) y los < v a la izquierda (rango [0, l)). Está claro que a partir de entonces, los elementos en cada una de las particiones quedarán en sus respectivos rangos, ya que todos los elementos en la partición derecha son estrictamente mayores que los de la izquierda. Podemos entonces aplicar quick-sort recursivamente a cada una de las particiones.
    - Tiempo de ejecución. Casos extremos
      - El mejor caso es cuando podemos elegir el pivote de tal manera que las particiones resultan ser bien balanceadas, es decir n1 ≈ n2 ≈ n/2. Si además asumimos que el algoritmo de particionamiento y elección del pivote son O(n), es decirEl mejor caso es cuando podemos elegir el pivote de tal manera que las particiones resultan ser bien balanceadas, es decir n1 ≈ n2 ≈ n/2. Si además asumimos que el algoritmo de particionamiento y elección del pivote son O(n), es decir