PROCEDURE BinarySearch (A : anArray,

Size : anArraySize,

Key : INTEGER,

VAR Found : BOOLEAN;

VAR Index : anArrayIndex);

Var Low, High : anArrayIndex;

BEGIN

LOW := 1;

High := Size;

REPEAT

Index := (Low + High) DIV 2;

If Key < A[Index]

THEN High := Index - 1

ELSE Low := Index + 1

UNTIL (Low > High) OR (Key = A[Index]);

FOUND := (Low <= High)

END

***Errores :*** el array se pasa por valor, entonces se copia todo el vector

:= (Low + High) DIV 2 esto puede dar overflow

Se compara solo por menor, no encuentro un dato q este solo en el medio (CREO)

* while (imin < imax)
  + mid = imin + ((imax-imin) / 2)
  + If (V[imid] < key)
    - imin = imid + 1
  + else
    - imax = imid
* If (imax == imin) && (V[imin] == key)
  + return imin
* else
  + return KEY\_NOT\_FOUND

BUSQUEDA BINARIA INTERPOLADA

**public** **int** interpolationSearch(**int**[] sortedArray, **int** toFind){

*// Returns index of toFind in sortedArray, or -1 if not found*

**int** low = 0;

**int** high = sortedArray.length - 1;

**int** mid;

**while** (sortedArray[low] <= toFind && sortedArray[high] >= toFind) {

mid = low +

((toFind - sortedArray[low]) \* (high - low)) /

(sortedArray[high] - sortedArray[low]); *//out of range is possible here*

**if** (sortedArray[mid] < toFind)

low = mid + 1;

**else** **if** (sortedArray[mid] > toFind)

*// Repetition of the comparison code is forced by syntax limitations.*

high = mid - 1;

**else**

**return** mid;

}

**if** (sortedArray[low] == toFind)

**return** low;

**else**

**return** -1; *// Not found*

}

En promedio, la búsqueda de interpolación hace aproximadamente log (log ( *n* )) comparaciones (si los elementos se distribuyen de manera uniforme), donde *n* es el número de elementos que se debe buscar. En el peor de los casos (por ejemplo, cuando los valores numéricos de las teclas aumentan de forma exponencial) se puede hacer hasta [O](http://en.wikipedia.org/wiki/Big_O_notation) ( *n* ) comparaciones.

Merge de dos fases. Basandome en el problema de gradiance : ... un archivo de texto de 17gb, cada registro del archivo tiene 80 bytes. Se pueden usar 500mb de memoria. ...  
Cargo en memoria de a 500 mb, quedandome generados 34 particiones de 500mb. Hago merge a de 6 particiones (raiz de 34), quedandome generadas 5 particiones de 3 gb y una de 2 gb. Luego hago un merge de estas 6 particiones, generandome el archivo ordenado.

 La primer fase consta generar los archivos de 3Gb con los de 500Mb.   
La segunda fase, consta de generar el de 17Gb con los de 3Gb y 2Gb.





