Algoritmos y Estructuras de Datos

Clase de Práctica 14 - Búsqueda y ordenamiento

Comisión A y F

Docente de práctica:

Tomás Assenza

Universidad Tecnológica Nacional - Facultad Regional Santa Fe

Búsqueda lineal

```
int buscaElemento(int arreglo[], int tam, int elemento) {
  int i = -1;
  bool encontrado = false;

  while (!encontrado and i < tam) {
    i++;
    if (arreglo[i] == elemento) encontrado = true;
  }
  if (i == tam) i = -1;
  return i;
}
```

Búsqueda binaria (como entenderla)

```
a = ALGUIEN_QUE_NO_CUMPLA;
b = ALGUIEN_QUE_CUMPLA;
```

Búsqueda binaria (como entenderla)

```
a = ALGUIEN_QUE_NO_CUMPLA;
b = ALGUIEN_QUE_CUMPLA;
while (b-a > 1) {
    c = (a+b)/2;
```

Búsqueda binaria (como entenderla)

```
a = ALGUIEN_QUE_NO_CUMPLA;
b = ALGUIEN_QUE_CUMPLA;
while (b-a > 1) {
    c = (a+b)/2;
    if (c CUMPLE)
       b = c;
    else
       a = c;
}
// Llegados a este punto, a es el ultimo que no cumple, y b es el primero que cumple.
```

Búsqueda binaria

```
int busquedaBinaria(int arreglo[], int tam, int elemento) {
  a = 0;
  b = tam - 1;
  while (b-a > 1) {
      c = (a+b)/2;
      if (arreglo[c] >= elemento)
          b = c;
      else
          a = c;
  }
  return b;
}
```

Ordenamiento seleccion directa

```
void seleccion(int vec[], const int tl) {
  int i, j, min;

  for (i = 0; i < tl; i++) {
    min = i;
    for (j = i + 1; j < tl; j++)
      if (vec[j] < vec[min]) min = j;
    intercambio(vec[i], vec[min]);
  }
}
```

Ordenamiento inserción directa

```
void insercion(int vec[], const int tl) {
  int i, j;

  for (i = 1; i < tl; i++) {
    j = i;
    while ((j > 0) && (vec[j] < vec[j - 1])) {
      intercambio(vec[j], vec[j - 1]);
      j = j - 1;
    }
  }
}
```

Ordenamiento burbuja

```
void burbuja(int vec[], const int tl) {
  int pasada, k

  for (pasada = 1; pasada < tl; pasada++) {
    for (k = 0; k < tl - 1; k++)
      if (vec[k] > vec[k + 1])
        intercambio(vec[k], vec[k + 1]);
  }
}
```

Ejercicios

1) En un arreglo de 1500 elementos se guardan números enteros ordenados ascendentemente. Defina la función cantidadRepetidos() que recibe el vector y retorna la cantidad de números que se repiten más de 10 veces

Ejercicios

1) En un arreglo de 1500 elementos se guardan números enteros ordenados ascendentemente. Defina la función cantidadRepetidos() que recibe el vector y retorna la cantidad de números que se repiten más de 10 veces

```
int cantidadRepetidos(int vector[], int tl){
  int contGeneral=0, contParcial=1;
  for (int i=1; i<tl; i++){
    if (vector[i] == vector[i-1]) {
      contParcial++;
    }
    else{
      if (contParcial >= 10){
        contGeneral++;
      }
      contParcial=1;
    }
  }
  return contGeneral;
}
```

Ejercicios

1) En un arreglo de 1500 elementos se guardan números enteros ordenados ascendentemente. Defina la función cantidadRepetidos() que recibe el vector y retorna la cantidad de números que se repiten más de 10 veces

```
int cantidadRepetidos(int vec[], int tama){
  int i = 0, repetidos= 0, anterior = vec[0];
  while(i<tama){
     if (vec[i] == vec[i + 9]) {
        if (i == 0 or vec[i] != anterior) {
           repetidos++;
           i+=9;
           anterior = vec[i];
     }
     else i++;
    }
    else i++;
  }
  return repetidos;
}
```

Ejercicios

2) La función llamada elimOrdenar recibe como parámetros un arreglo A de 500 números enteros, el tamaño lógico TLA del arreglo y un valor entero X. El arreglo A está desordenado. La función deberá devolver el vector A actualizado de la siguiente forma:

- Se deben eliminar todos los valores que sean múltiplos de X.
- El vector debe quedar ordenado ascendentemente.
- La función deberá retornar la cantidad de elementos eliminados de A

Ejercicios

2) La función llamada elimOrdenar recibe como parámetros un arreglo A de 500 números enteros, el tamaño lógico TLA del arreglo y un valor entero X. El arreglo A está desordenado. La función deberá devolver el vector A actualizado de la siguiente forma:

- Se deben eliminar todos los valores que sean múltiplos de X.
- El vector debe quedar ordenado ascendentemente.
- La función deberá retornar la cantidad de elementos eliminados de A

Ejercicios

3) La función Uno() recibe dos arreglos de 500 números enteros (A y B), los tamaños lógicos de cada uno (TA y TB, TB <= TA) y un valor booleano (C). El arreglo A y el arreglo B están desordenados. La función Uno() debe retornar un vector V y su tamaño lógico TV, ordenado ascendentemente si C==True (descendentemente en otro caso) conteniendo solamente los valores de B que se encuentran en A, sin repeticiones. Los vectores A y B pueden contener elementos repetidos.