UNIVERSIDAD NACIONAL DEL ALTIPLANO
Facultad de ingenieria Esatdistica e Informatica
Actividad:Algoritmos de Ordenamiento

Anthony Contreras

November 2024

1 Ejercicio 1: Ordenamiento por Selecci´on

```
    #include <iostream>
using namespace std;

void selectionSort(int arr[], int n, int &comparaciones) {
    for (int i = 0; i < n - 1; i++) {
        int minIdx = i;
        for (int j = i + 1; j < n; j++) {
            comparaciones++;
        if (arr[j] < arr[minIdx]) {
            minIdx = j;
        }
        }
        swap(arr[i], arr[minIdx]);
    }
}

int main() {
    int arr[] = {580, 320, 760, 435, 520};
    int n = sizeof(arr) / sizeof(arr[0]);
    int comparaciones = 0;
    selectionSort(arr, n, comparaciones);
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl << "comparaciones totales: " << comparaciones;

    return 0;
}
```

2 Ejercicio 2: Ordenamiento por Burbuja

```
#include <iostream>
using namespace std;

void bubbleSort(int arr[], int n, int &comparaciones) {
    for (int i = 0; i < n - 1; i++) {
    for (int j = 0; j < n - i - 1; j++) {
            comparaciones++;
        if (arr[j] > arr[j + 1]) {
            swap(arr[j], arr[j + 1]);
        }
    }
    }
}

int main() {
    int arr[] = {125,90,150,105,80};
    int n = sizeof(arr) / sizeof(arr[0]);
    int comparaciones = 0;
    bubbleSort(arr, n, comparaciones);
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl << " comparaciones: " << comparaciones;
    return 0;
}
```

3 Ejercicio 3: Insercion Directa

```
#include <iostream>
using namespace std;
void insertionSort(int arr[], int n) {
     for (int i = 1; i < n; i++) {
     int key = arr[i];
     int j = i - 1;
     while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j--;
    }
arr[j + 1] = key;
    }
}
int main() {
```

```
 int arr[] = {250,120,300,95,210};
    int n = sizeof(arr)/sizeof(arr[0]);
    insertionSort(arr, n);
    for (int i = 0; i < n; i++) cout << arr[i] << " ";
return 0;
}
```

4 Ejercicio 4: Ordenamiento Rapido (Quicksort)

```
    #include<iostream>
using namespace std;

int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = low - 1;
    for (int j = low; j < high; j++) {
        if (arr[j] < pivot) {
            i++;
            swap(arr[i], arr[j]);
        }
    }
    swap(arr[i + 1], arr[high]);
    return i + 1;
}
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

int main() {
    int arr[] = {850,230,690,540,310};
    int n = sizeof(arr) / sizeof(arr[0]);
    quickSort(arr, 0, n - 1);
    for (int i = 0; i < n; i++)
 cout << arr[i] << " ";
    return 0;
}
```

5 Ejercicio 5: Mergesort

```
#include<iostream>
using namespace std;

void mergeSort(int arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr,left, mid, right);
    }
}
int main() {
   int arr[] = {30.5,22.3,45.6,15.2,28.4};
    int n = sizeof(arr) / sizeof(arr[0]);
    mergeSort(arr, 0, n - 1);
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    return 0;
}
```