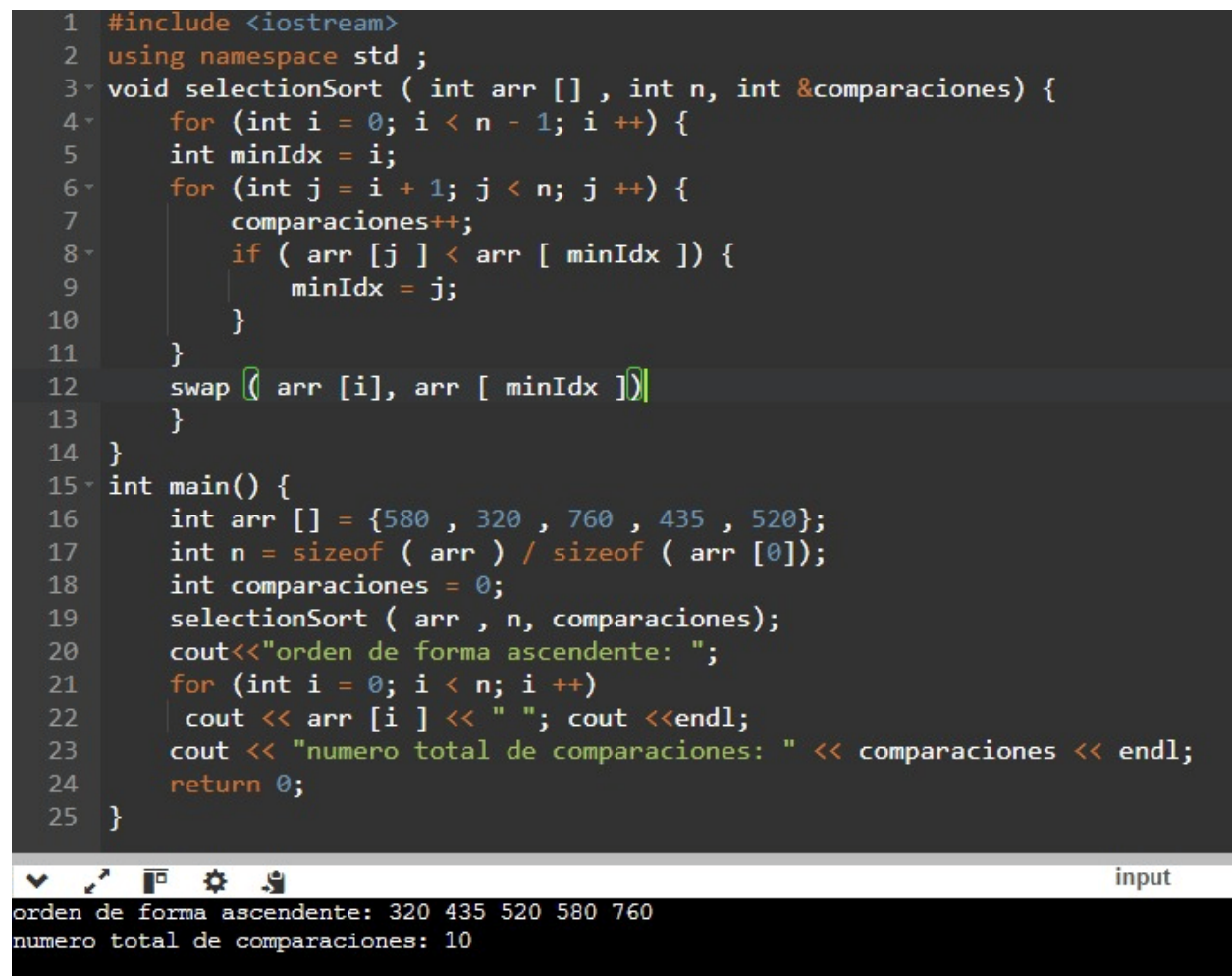


Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática
Docente: Fred Torres Cruz
Estudiante : Ruth Karina Apaza Solis

Trabajo Encargado: Algoritmos de ordenamiento

Ejercicio 1:

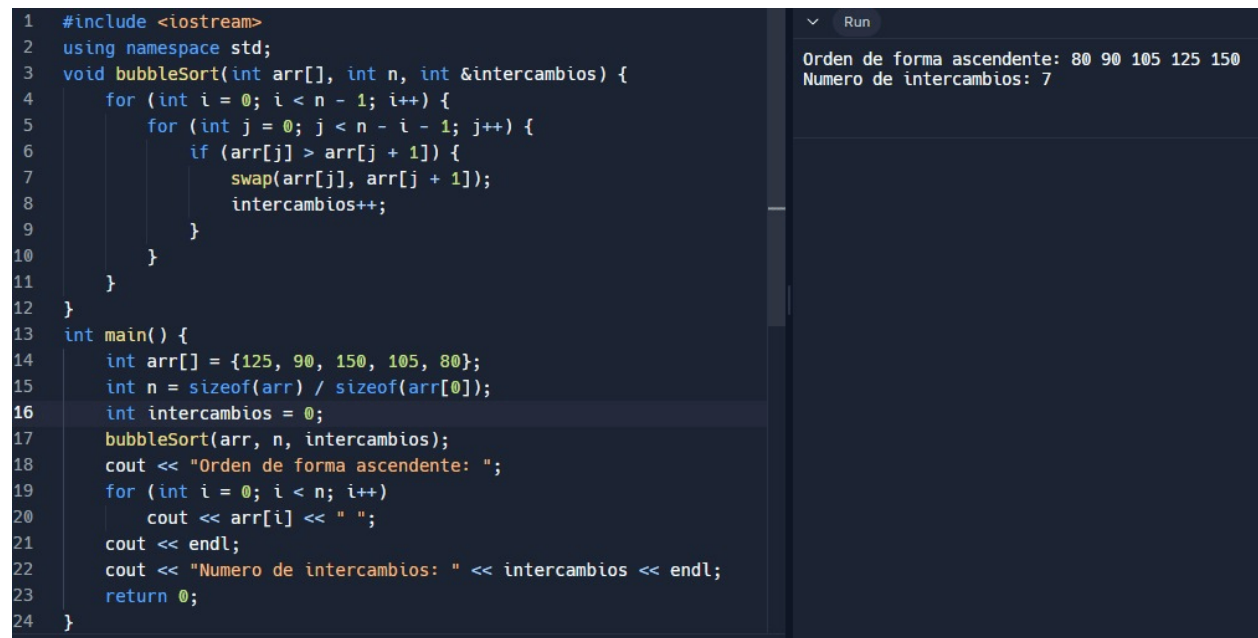
```
1  #include <iostream>
2  using namespace std ;
3  void selectionSort ( int arr [] , int n, int &comparaciones) {
4      for (int i = 0; i < n - 1; i ++ ) {
5          int minIdx = i;
6          for (int j = i + 1; j < n; j ++ ) {
7              comparaciones++;
8              if ( arr [j ] < arr [ minIdx ]) {
9                  minIdx = j;
10             }
11         }
12         swap ( arr [i], arr [ minIdx ])
13     }
14 }
15 int main() {
16     int arr [] = {580 , 320 , 760 , 435 , 520};
17     int n = sizeof ( arr ) / sizeof ( arr [0]);
18     int comparaciones = 0;
19     selectionSort ( arr , n, comparaciones);
20     cout<<"orden de forma ascendente: ";
21     for (int i = 0; i < n; i ++ )
22         cout << arr [i ] << " "; cout <<endl;
23     cout << "numero total de comparaciones: " << comparaciones << endl;
24     return 0;
25 }
```



orden de forma ascendente: 320 435 520 580 760
numero total de comparaciones: 10

Figura 1: ejercicio número 1

Ejercicio 2:



```
1  #include <iostream>
2  using namespace std;
3  void bubbleSort(int arr[], int n, int &intercambios) {
4      for (int i = 0; i < n - 1; i++) {
5          for (int j = 0; j < n - i - 1; j++) {
6              if (arr[j] > arr[j + 1]) {
7                  swap(arr[j], arr[j + 1]);
8                  intercambios++;
9              }
10         }
11     }
12 }
13 int main() {
14     int arr[] = {125, 90, 150, 105, 80};
15     int n = sizeof(arr) / sizeof(arr[0]);
16     int intercambios = 0;
17     bubbleSort(arr, n, intercambios);
18     cout << "Orden de forma ascendente: ";
19     for (int i = 0; i < n; i++)
20         cout << arr[i] << " ";
21     cout << endl;
22     cout << "Numero de intercambios: " << intercambios << endl;
23     return 0;
24 }
```

Run

Orden de forma ascendente: 80 90 105 125 150
Numero de intercambios: 7

Figura 2: ejercicio número 2

Ejercicio 3:

```
1  #include <iostream>
2  using namespace std ;
3  void insertionSort ( int arr [] , int n) {
4      for (int i = 1; i < n; i ++ ) {
5          int key = arr [i ];
6          int j = i - 1;
7          while (j >= 0 && arr [ j] > key ) {
8              arr [j + 1] = arr [j ];
9              j--;
10         }
11         arr [j + 1] = key;
12     }
13 }
14
15 int main () {
16     int arr [] = {250 , 120 , 300 , 95 , 210};
17     int n = sizeof ( arr ) / sizeof ( arr [0]);
18     insertionSort (arr , n);
19     cout << "orden de los valores en forma ascendente : ";
20     for (int i = 0; i < n; i ++ ) {
21         cout << arr [i] << " ";
22     }
23     return 0;
24 }
```

orden de los valores en forma ascendente : 95 120 210 250 300

...Program finished with exit code 0
Press ENTER to exit console.

Figura 3: ejercicio número 3

Ejercicio 4:

```
File include <iostream>
2 using namespace std ;
3 int partition (int arr [] , int low , int high ) {
4     int pivot = arr [ high ];
5     int i = low - 1;
6     for (int j = low ; j < high ; j ++ ) {
7         if ( arr [j ] < pivot ) {
8             i ++;
9             swap ( arr [i], arr [j ] );
10        }
11    }
12    swap ( arr [i + 1] , arr [ high ] );
13    return i + 1;
14 }
15 void quickSort ( int arr [] , int low , int high ) {
16     if ( low < high ) {
17         int pi = partition ( arr , low , high );
18         quickSort ( arr , low , pi - 1);
19         quickSort ( arr , pi + 1, high );
20     }
21 }
22 int main () {
23     int arr [] = {850 , 230 , 690 , 540 , 310};
24     int n = sizeof ( arr ) / sizeof ( arr [0]);
25     quickSort ( arr , 0, n - 1);
26     cout<<"tamaño de los archivos en forma ascendente: ";
27     for (int i = 0; i < n; i ++ ) {
28         cout << arr [i ] << " ";
29     }
30     return 0;
31 }
```

tamaño de los archivos en forma ascendente: 230 310 540 690 850

...Program finished with exit code 0
Press ENTER to exit console.

Figura 4: ejercicio número 4
Ejercicio 5:

```
1  #include <iostream>
2  using namespace std ;
3  void merge ( double arr [] , int l , int m , int r) {
4      int n1 = m - l + 1;
5      int n2 = r - m;
6      double L[ n1 ], R[ n2 ];
7      for (int i = 0; i < n1 ; i ++ ) L [i] = arr [l + i ];
8      for (int j = 0; j < n2 ; j ++ ) R [j] = arr [m + 1 + j ];
9      int i = 0, j = 0 , k = l;
10     while (i < n1 && j < n2 ) {
11         if (L [i] <= R[j ]) {
12             arr [k] = L[i ];
13             i ++;
14         } else {
15             arr [k] = R[j ];
16             j ++;
17         }
18         k ++;
19     }
20     while (i < n1 ) arr [k ++] = L[i ++];
21     while (j < n2 ) arr [k ++] = R[j ++];
22 }
23 void mergeSort ( double arr [] , int l , int r) {
24     if (l < r) {
```

Figura 5: ejercicio número 5


```
18     k ++,
19 }
20 while (i < n1 ) arr [k ++] = L[i ++];
21 while (j < n2 ) arr [k ++] = R[j ++];
22 }
23 void mergeSort ( double arr [] , int l , int r) {
24     if (l < r) {
25         int m = l + (r - l) / 2;
26         mergeSort ( arr , l , m );
27         mergeSort ( arr , m + 1, r );
28         merge ( arr , l , m , r );
29     }
30 }
31 int main () {
32     double arr [] = {30.5 , 22.3 , 45.6 , 15.2 , 28.4};
33     int n = sizeof ( arr ) / sizeof ( arr [0]);
34     mergeSort ( arr , 0, n - 1);
35     cout << "tiempo en forma ascendente: ";
36     for (int i = 0; i < n; i ++ ) {
37         cout << arr [i] << " ";
38     }
39     return 0;
40 }
```

tiempo en forma ascendente: 15.2 22.3 28.4 30.5 45.6

...Program finished with exit code 0
Press ENTER to exit console.

Figura 6: ejercicio número 5