

# Sorting

## Examples:

1. Selection Sort - ascending
2. Selection Sort - descending
3. Insertion Sort - ascending
4. Insertion Sort – descending
5. Sorting Algorithms Visualization

1. An array contains the elements shown below.  
Show the contents of the array after three passes of the  
**SELECTION SORT** algorithm.

15, 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

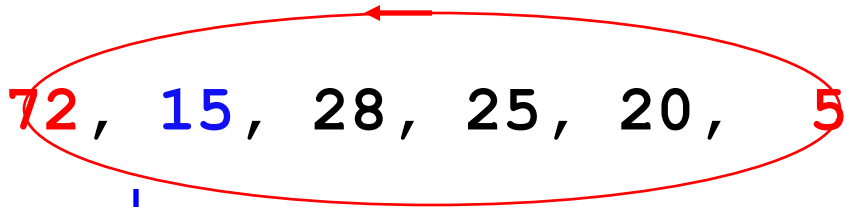
Selection sort algorithm.

15, 72, 2, 28, 25, 20, 5, 10, 60, 9, 50  
2, | 72, 15, 28, 25, 20, 5, 10, 60, 9, 50

## Selection sort algorithm.

15, 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

2, | 72, 15, 28, 25, 20, 5, 10, 60, 9, 50



2, 5, | 15, 28, 25, 20, 72, 10, 60, 9, 50

## Selection sort algorithm.

15, 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

2, | 72, 15, 28, 25, 20, 5, 10, 60, 9, 50

2, 5, | 15, 28, 25, 20, 72, 10, 60, 9, 50

2, 5, 9, | 28, 25, 20, 72, 10, 60, 15, 50

```

void selectionSortAscending(double ary[], int size)
{
    double temp;
    int     small;    // index of the smallest element

    for (int wall = 0; wall < size - 1; wall++)
    { // Look for the smallest value and find its location
        small = wall;
        for (int curr = wall + 1; curr < size; curr++)
        {
            if (ary[curr] < ary[small])
                small = curr;
        }
        // Exchange
        temp      = ary[wall];
        ary[wall] = ary[small];
        ary[small] = temp;
    }
}

```

## // Selection Sort – a calling statement

```
int main( void )
{
    int    size = 7;
    double ary[SIZE] = { 70.3, 30.1, 40.5, 50.2, 10.2,
                        80.7, 30.1 };

    selectionSortAscending( ary, size);

    return 0;
}
```

**2. Change the selection sort function to sort an array in descending order.**



```

void selectionSortDescending(double ary[], int size)
{
    double temp;
    int     large;    // index of the largest element

    for (int wall = 0; wall < size - 1; wall++)
    { // Look for the largest value and find its location
        large = wall;
        for (int curr = wall + 1; curr < size; curr++)
        {
            if (ary[curr] > ary[large])
                large = curr;
        }
        // Exchange
        temp      = ary[wall];
        ary[wall] = ary[large];
        ary[large] = temp;
    }
}

```

3. An array contains the elements shown below.  
Show the contents of the array after three passes of the  
**INSERTION SORT** algorithm.

15, 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

Insertion sort algorithm.

15, | 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

15, 72, | 2, 28, 25, 20, 5, 10, 60, 9, 50

The array does not change!

Insertion sort algorithm.

15, | 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

15, 72, | 2, 28, 25, 20, 5, 10, 60, 9, 50

2, 15, 72, | 28, 25, 20, 5, 10, 60, 9, 50

Insertion sort algorithm.

15, | 72, 2, 28, 25, 20, 5, 10, 60, 9, 50

15, 72, | 2, 28, 25, 20, 5, 10, 60, 9, 50

2, 15, 72, | 28, 25, 20, 5, 10, 60, 9, 50

2, 15, 28, 72, | 25, 20, 5, 10, 60, 9, 50

```
void insertionSortAscending(double ary[], int size)
{
    for (int curr = 1; curr < size; curr++)
    {
        // make a copy of the current element
        double temp = ary[curr];

        // shift elements in the sorted part of the list to make room
        int walk = curr - 1;
        while( walk >= 0 && temp < ary[walk] )
        {
            ary[walk + 1] = ary[walk];
            walk--;
        }

        // put temp back into the list
        ary[walk + 1] = temp;
    }
}
```

## // Insertion Sort – a calling statement

```
int main( void )
{
    int    size;
    double ary[SIZE];

    // ...
    insertionSortAscending( ary, size);

    return 0;
}
```

4. Change the insertion sort function to sort an array in descending order.



```
void insertionSortDescending(double ary[], int size)
{
    for (int curr = 1; curr < size; curr++)
    {
        // make a copy of the current element
        double temp = ary[curr];

        // shift elements in the sorted part of the list to make room
        int walk = curr - 1;
        while( walk >= 0 && temp > ary[walk] )
        {
            ary[walk + 1] = ary[walk];
            walk--;
        }

        // put temp back into the list
        ary[walk + 1] = temp;
    }
}
```

# Sorting

## Examples:

- ✓ 1. Selection Sort - ascending
- ✓ 2. Selection Sort - descending
- ✓ 3. Insertion Sort - ascending
- ✓ 4. Insertion Sort – descending
- ✓ 5. Sorting Algorithms Visualization