Dynamic Programming

Greedy Approach

# Selection sort

> In selection sort, we will select the optimal element for every index by comparing all other elements.

## Example

> In school prayer, all the student in a row should stand based on their height. i.e. the shortest person will stand in the beginning and the tallest one will stand in the end.

Assume that there are 5 student standing in the row.

student 1 height = 180cm

student 2 height = 165cm

student 3 height = 150cm

student 4 height = 170cm

student 5 height = 145cm

Let's sort the array by selecting correct person for each place.

## Given Details

Total students = 5

Student heights = {180cm,165cm,150cm,170cm,145cm}

## Goal

Make the row sorted.

Like below,

145cm, 150cm, 165cm, 170cm, 180cm

## Selection sort procedure

Take the first student height and compare the first student height with all other student who stands behind the first person.

if anyone has smaller height, interchange their positon.

Take second student height and compare the second student height with all other student who stands behind the second person.

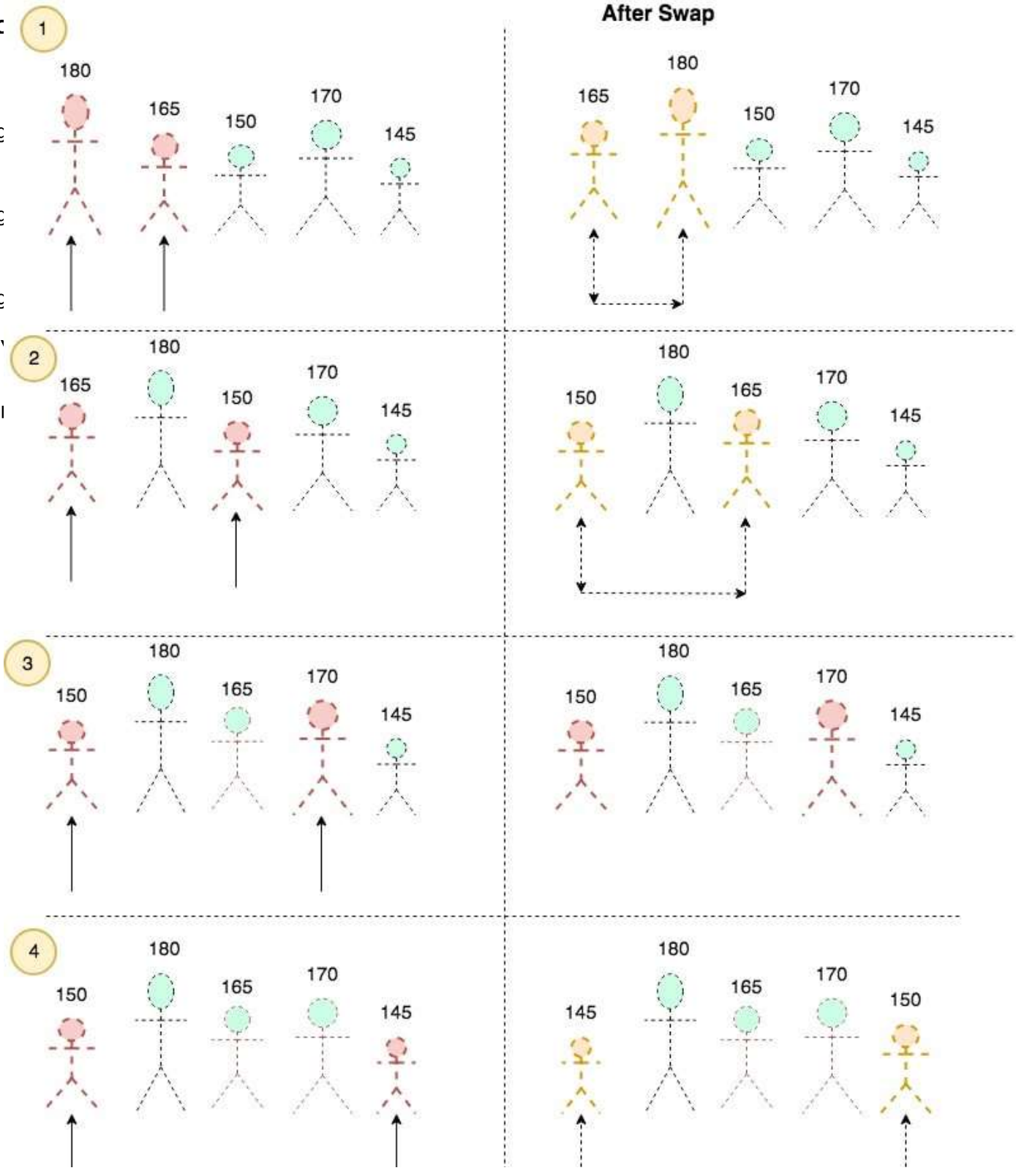
if anyone has smaller height, interchange their positon.

Do the above step for all the positions.

# Selection sort step by step

## Step 1

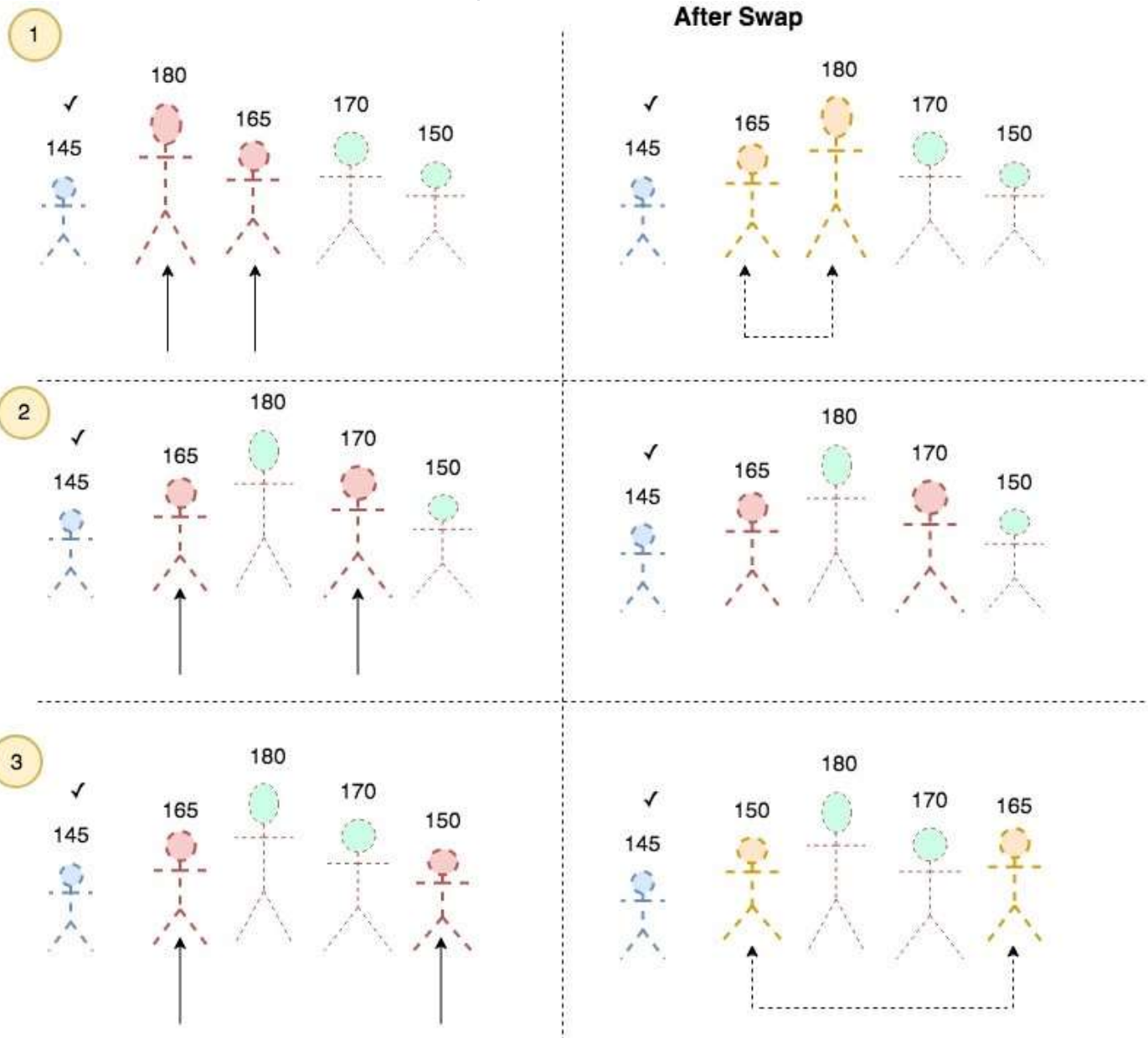
Searching



1. 165 < 180. Interchange their positon.
2. 150 < 165. Interchange their positon.
3. 170 > 150. Leave it as it is.
4. 145 < 150. Interchange their positon.

Selection Sort

Step 2



1. 165 < 180. Interchange their positon.
2. 170 > 165. Leave it as it is.
3. 150 < 165. Interchange their positon.



## Step 3

Searching

Sc

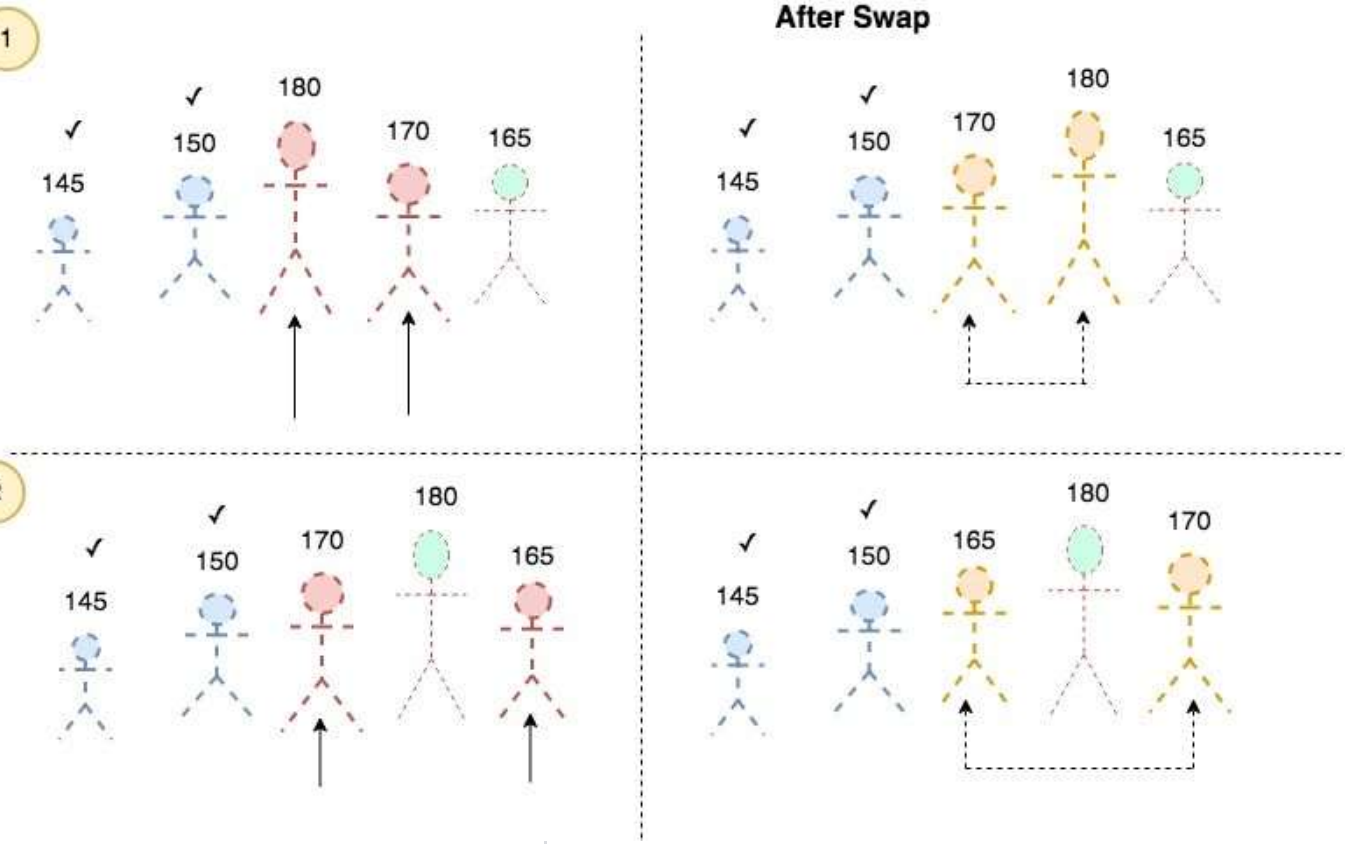
(/alg

(/alg

(/alg

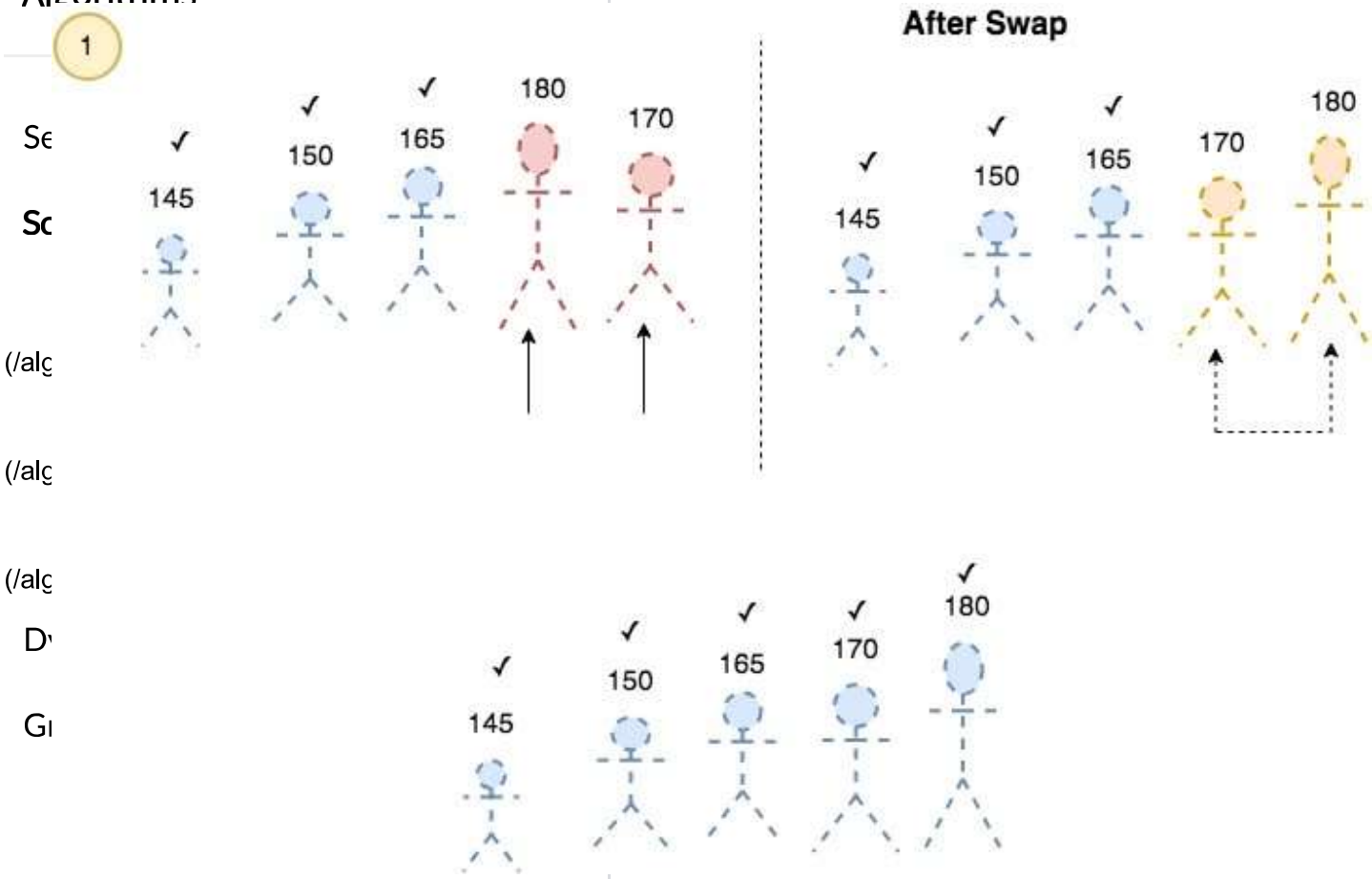
D

G




1. 170 < 180. Interchange their positon.
2. 165 < 170. Interchange their positon.

## Step 4




1. 170 < 180. Interchange their positon.

## Selection sort program in c

Example

```c
/*
 * Program : Selection sort
 * Language : C
 */

#include<stdio.h>
#define SIZE 5

void swap(int *x, int *y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

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

}

int main()
{
    int arr[SIZE] = {180,165,150,170,145},i;

    selectionSort(arr,SIZE);

    printf("After selection sort\n");
    for(i = 0; i < SIZE; i++)
        printf("%d ",arr[i]);

    return 0;
}
```

Run it (try-it-selection-sort.html)

(/) Courses

## Algorithms

PAGE -- (https://www.log2base2.com/algorithms/searching/open-hashing.html)

PAGE ++ (https://www.log2base2.com/algorithms/sorting/bubble-sort-algorithm-in-c.html)

**Sorting**

Selection Sort

(/algorithms/sorting/selection-sort.html)

Bubble Sort

(/algorithms/sorting/bubble-sort-algorithm-in-c.html)

Quicksort

(/algorithms/sorting/quick-sort.html)

Dynamic Programming

Greedy Approach