LABORATORIUM SISTEM INFORMASI
UNIVERSITAS TANJUNGPURA PONTIANAK

Gedung FMIPA Jl. Prof. Dr. Hadari Nawawi Pontianak

Hari/Tanggal: Kamis / 8 Maret 2025

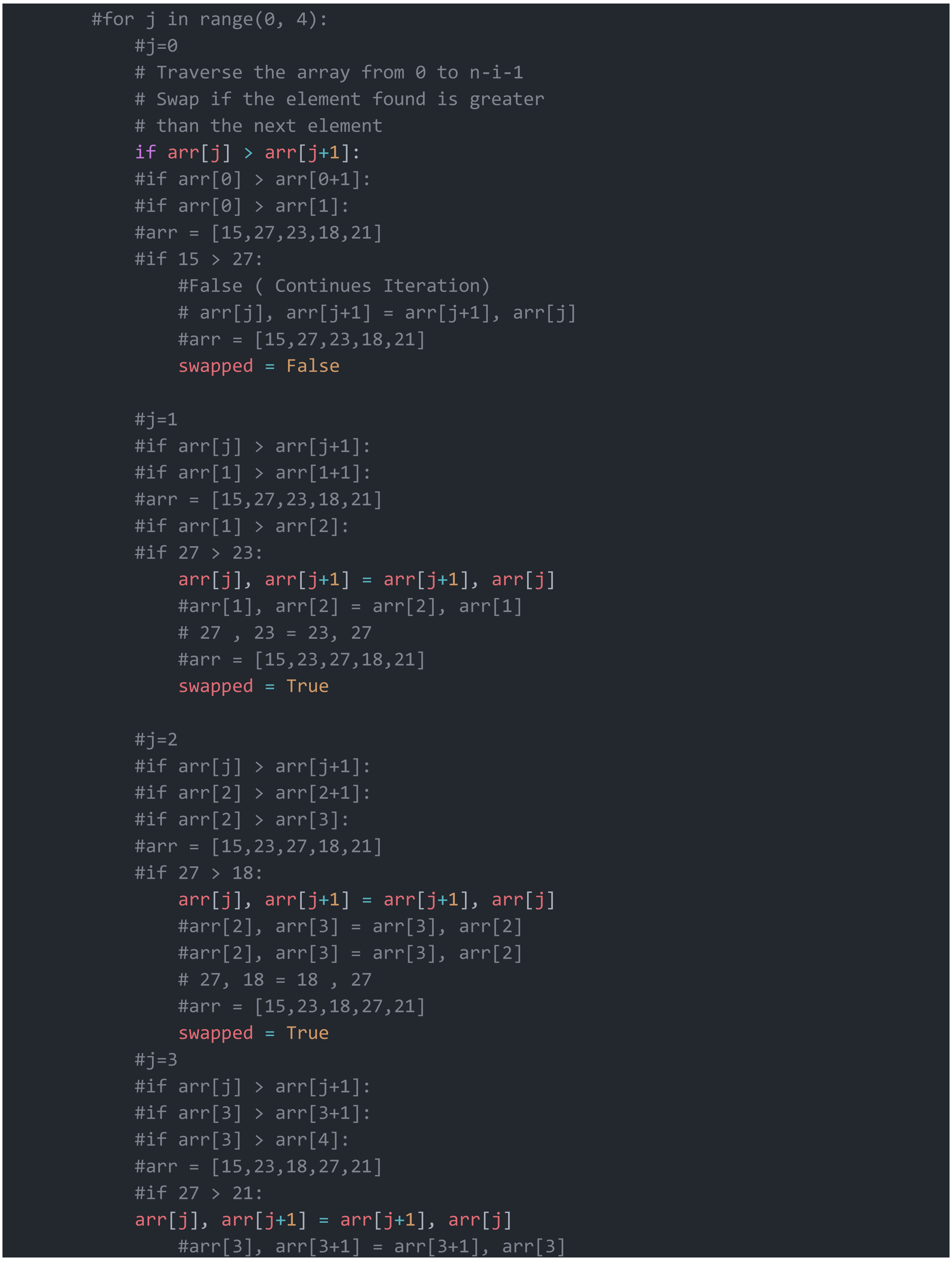
Nama Mahasiswa	Rafli Pratama	**Mata Kuliah Praktikum**	Algoritma dan Struktur
NIM	H1101241008	**Dosen Pengampu**	
Semester	2 Genap	**Paraf Dosen Pengampu**	
Kelas	Sistem Informasi A	**Asisten Praktikum**	
Nilai		**Paraf Asisten Praktikum**	

LEMBAR KERJA PRAKTIKUM

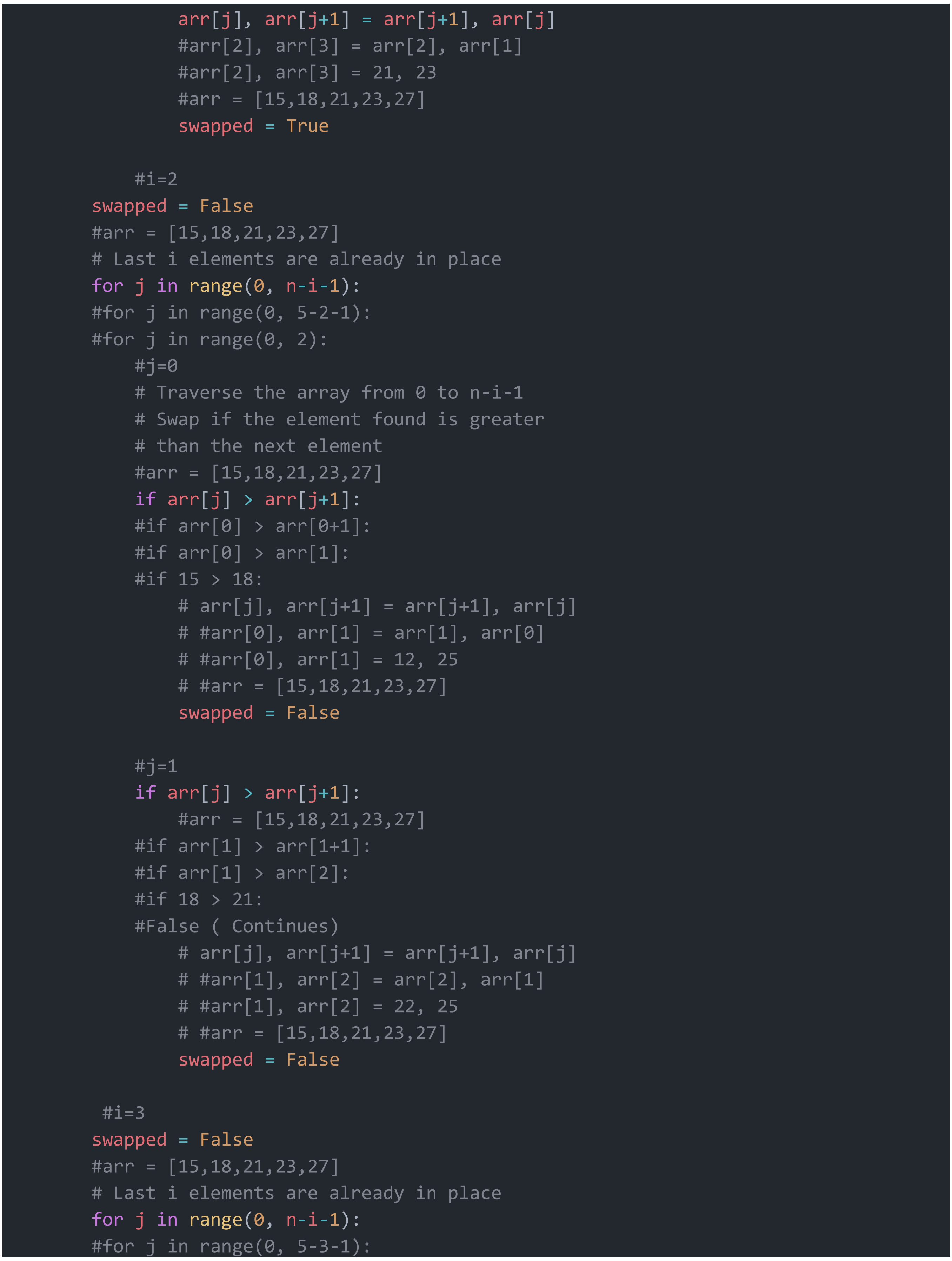
MATERI PRAKTIKUM : Sorting

Percobaan 1. Uji Coba BubbleSort

```python
# Optimized Python program for implementation of Bubble Sort

#arr = [15,27,23,18,21]
def bubbleSort(arr):
    n = len(arr)
    #n = len(arr)
    #n = 5


    # Traverse through all array elements
    for i in range(n):
    #for i in range(5):
    #i=0
        swapped = False

        # Last i elements are already in place
        for j in range(0, n-i-1):
        #for j in range(0, 5-0-1):
```



```
        #for j in range(0, 4):
            #j=0
            # Traverse the array from 0 to n-i-1
            # Swap if the element found is greater
            # than the next element
            if arr[j] > arr[j+1]:
            #if arr[0] > arr[0+1]:
            #if arr[0] > arr[1]:
            #arr = [15,27,23,18,21]
            #if 15 > 27:
                #False ( Continues Iteration)
                # arr[j], arr[j+1] = arr[j+1], arr[j]
                #arr = [15,27,23,18,21]
                swapped = False

            #j=1
            #if arr[j] > arr[j+1]:
            #if arr[1] > arr[1+1]:
            #arr = [15,27,23,18,21]
            #if arr[1] > arr[2]:
            #if 27 > 23:
                arr[j], arr[j+1] = arr[j+1], arr[j]
                #arr[1], arr[2] = arr[2], arr[1]
                # 27 , 23 = 23, 27
                #arr = [15,23,27,18,21]
                swapped = True

            #j=2
            #if arr[j] > arr[j+1]:
            #if arr[2] > arr[2+1]:
            #if arr[2] > arr[3]:
            #arr = [15,23,27,18,21]
            #if 27 > 18:
                arr[j], arr[j+1] = arr[j+1], arr[j]
                #arr[2], arr[3] = arr[3], arr[2]
                #arr[2], arr[3] = arr[3], arr[2]
                # 27, 18 = 18 , 27
                #arr = [15,23,18,27,21]
                swapped = True
            #j=3
            #if arr[j] > arr[j+1]:
            #if arr[3] > arr[3+1]:
            #if arr[3] > arr[4]:
            #arr = [15,23,18,27,21]
            #if 27 > 21:
            arr[j], arr[j+1] = arr[j+1], arr[j]
                #arr[3], arr[3+1] = arr[3+1], arr[3]
        #arr[3], arr[4] = arr[4], arr[3]
        #27, 21 = 21, 27
        #arr = [15,23,18,21,27]
      swapped = True

    #i=1
    swapped = False
    #arr = [15,23,18,21,27]
    # Last i elements are already in place
    for j in range(0, n-i-1):
    #for j in range(0, 5-1-1):
    #for j in range(0, 3):
      #j=0
      # Traverse the array from 0 to n-i-1
      # Swap if the element found is greater
      # than the next element
      #arr = [15,23,18,21,27]
      if arr[j] > arr[j+1]:
      #if arr[0] > arr[0+1]:
      #arr = [15,23,18,21,27]
      #if arr[0] > arr[1]:
      #if 15 > 23:
      # False ( Continue Iteration)
        # arr[j], arr[j+1] = arr[j+1], arr[j]
        #arr[0], arr[1] = arr[1], arr[0]
        #arr[0], arr[1] = 15, 23
        #arr = [15,23,18,21,27]
        swapped = False

      #j=1
      #if arr[j] > arr[j+1]:
        #arr = [15,23,18,21,27]
      #if arr[1] > arr[1+1]:
      #if arr[1] > arr[2]:
      #if 23 > 18:
        arr[j], arr[j+1] = arr[j+1], arr[j]
        #arr[1], arr[2] = arr[2], arr[1]
        #arr[1], arr[2] = 18, 23
        #arr = [15,18,23,21,27]
        swapped = True

      #j=2
      #if arr[j] > arr[j+1]:
      #arr = [15,18,23,21,27]
      #if arr[2] > arr[2+1]:
      #if arr[2] > arr[3]:
       #if 23 > 21:
```



```
                arr[j], arr[j+1] = arr[j+1], arr[j]
                #arr[2], arr[3] = arr[2], arr[1]
                #arr[2], arr[3] = 21, 23
                #arr = [15,18,21,23,27]
                swapped = True

            #i=2
        swapped = False
        #arr = [15,18,21,23,27]
        # Last i elements are already in place
        for j in range(0, n-i-1):
        #for j in range(0, 5-2-1):
        #for j in range(0, 2):
            #j=0
            # Traverse the array from 0 to n-i-1
            # Swap if the element found is greater
            # than the next element
            #arr = [15,18,21,23,27]
            if arr[j] > arr[j+1]:
            #if arr[0] > arr[0+1]:
            #if arr[0] > arr[1]:
            #if 15 > 18:
                # arr[j], arr[j+1] = arr[j+1], arr[j]
                # #arr[0], arr[1] = arr[1], arr[0]
                # #arr[0], arr[1] = 12, 25
                # #arr = [15,18,21,23,27]
                swapped = False

            #j=1
            if arr[j] > arr[j+1]:
                #arr = [15,18,21,23,27]
            #if arr[1] > arr[1+1]:
            #if arr[1] > arr[2]:
            #if 18 > 21:
            #False ( Continues)
                # arr[j], arr[j+1] = arr[j+1], arr[j]
                # #arr[1], arr[2] = arr[2], arr[1]
                # #arr[1], arr[2] = 22, 25
                # #arr = [15,18,21,23,27]
                swapped = False

         #i=3
        swapped = False
        #arr = [15,18,21,23,27]
        # Last i elements are already in place
        for j in range(0, n-i-1):
        #for j in range(0, 5-3-1):
```

```
        #for j in range(0, 1):
            #j=0
            # Traverse the array from 0 to n-i-1
            # Swap if the element found is greater
            # than the next element
            #arr = [15,18,21,23,27]
            if arr[j] > arr[j+1]:
            # if arr[0] > arr[0+1]:
            # if arr[0] > arr[1]:
            # if 15 > 18:


                if (swapped == False):
                    break

        #arr = [12, 22, 25, 34, 64]

# Driver code to test above
if __name__ == "__main__":
    arr = [15,27,23,18,21]

    bubbleSort(arr)

    print("Sorted array:")
    for i in range(len(arr)):
        print("%d" % arr[i], end=" ")
```

Output Program

Simulasi program bubbleSort

Percobaan 2. Uji Coba SelectionSort

```
# Python program for implementation of Selection
# Sort
#arr = [15,27,23,18,21]
def selection_sort(arr):
    n = len(arr)
    #n=5
    for i in range(n - 1):
    #for i in range(5 - 1):
```

```
    #for i in range(4):
        #i=0
        # Assume the current position holds
        # the minimum element
        min_idx = i
        #min_idx = 0

        # Iterate through the unsorted portion
        # to find the actual minimum
        for j in range(i + 1, n):
        #for j in range(0 + 1, 5):
        #for j in range(1, 5):
        #j=1
        #arr = [15,27,23,18,21]
            if arr[j] < arr[min_idx]:
            #if arr[1] < arr[0]:
            #if 27 < 15:

                # # Update min_idx if a smaller element is found
                # min_idx = j
                # #min_idx = 0

        #j=2
        #arr = [15,27,23,18,21
            if arr[j] < arr[min_idx]:
            #if arr[2] < arr[0]:
            #if 23 < 15:

                # Update min_idx if a smaller element is found
                min_idx = j
                #min_idx = 0

        #j=3
        #arr = [15,27,23,18,21]
            if arr[j] < arr[min_idx]:
            #if arr[3] < arr[0]:
            #if 18 < 15:
                # Update min_idx if a smaller element is found
                min_idx = j
                #min_idx = 0
        # Move minimum element to its
        # correct position

        #j=4
        #arr = [15,27,23,18,21
            if arr[j] < arr[min_idx]:
            #if arr[4] < arr[0]:
```

```
            #if 21 < 15:
                # Update min_idx if a smaller element is found
                min_idx = j
                #min_idx = 0
        # Move minimum element to its
        # correct position
        #arr = [15,27,23,18,21]


#i=1
        # Assume the current position holds
        # the minimum element
        min_idx = i
        #min_idx = 1

        # Iterate through the unsorted portion
        # to find the actual minimum
        for j in range(i + 1, n):
        #for j in range(1 + 1, 5):
        #for j in range(2, 5):
        #arr = [15,27,23,18,21]
        #j=2
            if arr[j] < arr[min_idx]:
            #if arr[2] < arr[1]:
            #if 23 < 27 :
                # Update min_idx if a smaller element is found
                min_idx = j
                #min_idx = 2

        #j=3
        #arr = [15,27,23,18,21]
            if arr[j] < arr[min_idx]:
            #if arr[3] < arr[2]:
            #if 18 < 23:
            #min_idx = j
            # min_idx = 3

        #j=4
        #arr = [15,27,23,18,21]
            if arr[j] < arr[min_idx]:
            #if arr[4] < arr[3]:
            #if 21 < 18:

        arr[i], arr[min_idx] = arr[min_idx], arr[i]
        #arr[i], arr[min_idx] = arr[min_idx], arr[i]
        #arr[i], arr[min_idx] = arr[1], arr[3]
        # 27, 18 = 18 , 27
```

```
    #arr[i], arr[min_idx] = 18, 27
    #arr = [15,18,23,27,21]

#i=2
    # Assume the current position holds
    # the minimum element
    min_idx = i
    #min_idx = 2

    # Iterate through the unsorted portion
    # to find the actual minimum
    for j in range(i + 1, n):
    #for j in range(2 + 1, 5):
    #for j in range(3, 5):
    #arr = [15,18,23,27,21]
    #j=3
        if arr[j] < arr[min_idx]:
        #if arr[3] < arr[2]:
        #if 27 < 23:
         min_idx = j
            #min_idx = 2

    #j=4
    #arr = [15,18,23,27,21]
        if arr[j] < arr[min_idx]:
        #if arr[4] < arr[2]:
        #if 21 < 23:
        #min_idx = 4

    arr[i], arr[min_idx] = arr[min_idx], arr[i]
    #arr[i], arr[min_idx] = arr[2], arr[4]
    #arr[i], arr[min_idx] = 21 , 23

    #arr = [15,18,21,27,23]

#i=3
    # Assume the current position holds
    # the minimum element
    min_idx = i
    #min_idx = 3

    # Iterate through the unsorted portion
    # to find the actual minimum
    for j in range(i + 1, n):
    #for j in range(3 + 1, 5):
    #for j in range(4, 5):
    #arr = [15,18,21,27,23]
```

```
        #j=4
        #arr = [15,18,21,27,23]
            if arr[j] < arr[min_idx]:
            #if arr[4] < arr[3]:
            #if 23 < 27:
            #min_idx = 4

                arr[i], arr[min_idx] = arr[min_idx], arr[i]
        #arr[3], arr[4] = arr[4], arr[3]
        #arr[3], arr[4] = 23,27

        #arr = [15,18,21,23,27]

def print_array(arr):
    for val in arr:
        print(val, end=" ")
    print()

if __name__ == "__main__":
    arr = [15, 27 ,23 ,18 , 21]

    print("Original array: ", end="")
    print_array(arr)

    selection_sort(arr)

    print("Sorted array: ", end="")
    print_array(arr)
```

Output Program

Simulasi program selection Sort

Percobaan 3. Simulasi Insertion Sort

```
# Python program for implementation of Insertion Sort

# Function to sort array using insertion sort
# Python program for implementation of Insertion Sort

# Function to sort array using insertion sort
def insertion_sort(arr):
    n = len(arr)
    # n = 5
    for i in range(1, n):
        # Iterasi i = 1
        key = arr[i]
        # key = arr[1] = 16
        j = i - 1
        # j = 1 - 1 = 0

        # Geser elemen yang lebih besar dari key ke kanan
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
        # arr setelah iterasi pertama: [11, 16, 13, 29, 12]

        # Iterasi i = 2
        key = arr[i]
        # key = arr[2] = 13
        j = i - 1
        # j = 2 - 1 = 1
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
        # arr setelah iterasi kedua: [11, 13, 16, 29, 12]

        # Iterasi i = 3
        key = arr[i]
        # key = arr[3] = 29
        j = i - 1
        # j = 3 - 1 = 2
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
        # arr setelah iterasi ketiga: [11, 13, 16, 29, 12]

        # Iterasi i = 4
        key = arr[i]
        # key = arr[4] = 12
        j = i - 1
        # j = 4 - 1 = 3
```

```python
        while j >= 0 and arr[j] > key:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
        # arr setelah iterasi keempat: [11, 12, 13, 16, 29]

# Utility function untuk mencetak array
def print_array(arr):
    for val in arr:
        print(val, end=" ")
    print()

# Driver method
if __name__ == "__main__":
    arr = [11, 16, 13, 29, 12]
    print("Original array: ", end="")
    print_array(arr)

    insertion_sort(arr)

    print("Sorted array: ", end="")
    print_array(arr)
```

Output Program

Simulasi program insertionSort