TUGAS INDIVIDU
ALGORITMA SORTING DI PYTHON

Disusun untuk memenuhi tugas
Mata Kuliah: Dasar Pemrograman
Dosen Pengampu: Indira Syawanodya, S.Kom., M.Kom.

Oleh:
Mohamad Firdiansyah (2402041)

PROGRAM STUDI REKAYASA PERANGKAT LUNAK
KAMPUS UPI CIBIRU
UNIVERSITAS PENDIDIKAN INDONESIA
2024

Menentukan Algoritma Sorting Tercepat di Python

1. Radix Sort

Code:

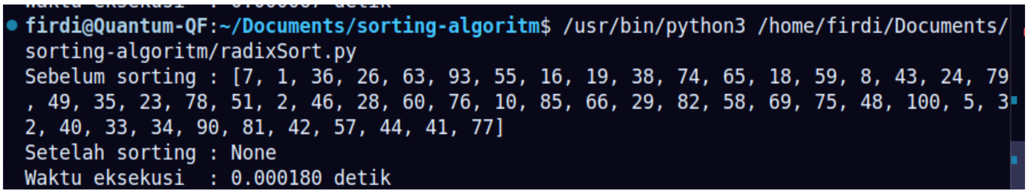
```
import time
startTime = time.time()

def sortByPlace(arr, place):
    size = len(arr)
    output = [0] * size
    count = [0] * 10
    for num in arr:
        index = (num // place) % 10
        count[index] += 1
    for i in range(1, 10):
        count[i] += count[i - 1]
    for i in range (size - 1, -1, -1):
        index = (arr[i] // place) % 10
        output[count[index] - 1] = arr[i]
        count[index] -= 1
    for i in range(size):
        arr[i] = output[i]

def radixSort(arr):
    maxNum = max(arr)
    place = 1
    while maxNum // place > 0:
        sortByPlace(arr, place)
        place *= 10

arrData =  [7, 1, 36, 26, 63, 93, 55, 16, 19, 38, 74, 65, 18, 59, 8, 43, 24, 79, 49, 35, 23, 78, 51, 2, 46, 28, 60, 76, 10, 85, 66, 29, 82, 58, 69, 75, 48, 100, 5, 32, 40, 33, 34, 90, 81, 42, 57, 44, 41, 77]
print("Sebelum sorting\t:", arrData)
print("Setelah sorting\t:", radixSort(arrData))
endTime = time.time()
print(f"Waktu eksekusi\t: {endTime - startTime:.6f} detik")
```

Screenshot output:



```
firdi@Quantum-QF:~/Documents/sorting-algoritm$ /usr/bin/python3 /home/firdi/Documents/
sorting-algoritm/radixSort.py
Sebelum sorting : [7, 1, 36, 26, 63, 93, 55, 16, 19, 38, 74, 65, 18, 59, 8, 43, 24, 79
, 49, 35, 23, 78, 51, 2, 46, 28, 60, 76, 10, 85, 66, 29, 82, 58, 69, 75, 48, 100, 5, 3
2, 40, 33, 34, 90, 81, 42, 57, 44, 41, 77]
Setelah sorting : None
Waktu eksekusi  : 0.000180 detik
```

2. Quick Sort

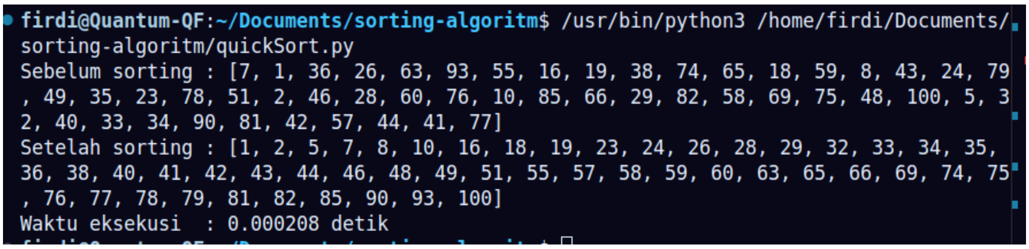
Code:

```
import time
startTime = time.time()

def quickSort(arr):
    if len(arr) <= 1:
        return arr
    else:
        pivot = arr[0]
        left = [x for x in arr[1:] if x <= pivot]
        right = [x for x in arr[1:] if x > pivot]
        return quickSort(left) + [pivot] + quickSort(right)

arrData =  [7, 1, 36, 26, 63, 93, 55, 16, 19, 38, 74, 65, 18, 59, 8, 43, 24, 79, 49, 35,
23, 78, 51, 2, 46, 28, 60, 76, 10, 85, 66, 29, 82, 58, 69, 75, 48, 100, 5, 32, 40, 33, 34,
90, 81, 42, 57, 44, 41, 77]
print("Sebelum sorting\t:", arrData)
print("Setelah sorting\t:", quickSort(arrData))
endTime = time.time()
print(f"Waktu eksekusi\t: {endTime - startTime:.6f} detik")
```

Screenshot output:



```
firdi@Quantum-QF:~/Documents/sorting-algoritm$ /usr/bin/python3 /home/firdi/Documents/
sorting-algoritm/quickSort.py
Sebelum sorting : [7, 1, 36, 26, 63, 93, 55, 16, 19, 38, 74, 65, 18, 59, 8, 43, 24, 79
, 49, 35, 23, 78, 51, 2, 46, 28, 60, 76, 10, 85, 66, 29, 82, 58, 69, 75, 48, 100, 5, 3
2, 40, 33, 34, 90, 81, 42, 57, 44, 41, 77]
Setelah sorting : [1, 2, 5, 7, 8, 10, 16, 18, 19, 23, 24, 26, 28, 29, 32, 33, 34, 35,
36, 38, 40, 41, 42, 43, 44, 46, 48, 49, 51, 55, 57, 58, 59, 60, 63, 65, 66, 69, 74, 75
, 76, 77, 78, 79, 81, 82, 85, 90, 93, 100]
Waktu eksekusi  : 0.000208 detik
```

3. Counting Sort

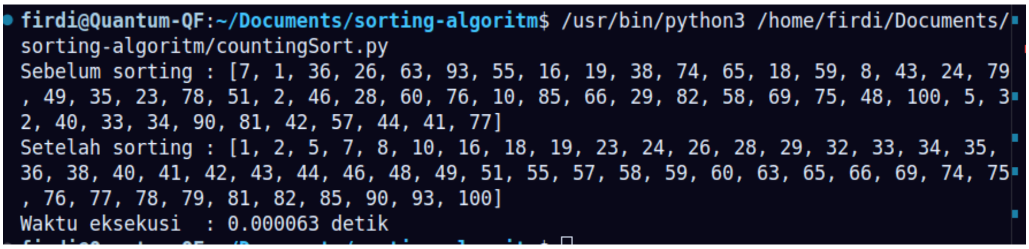
Code:

```
import time
startTime = time.time()

def countingSort(arr):
    n = len(arr)
    if n == 0:
        return arr
    k = max(arr)
    count = [0] * (k + 1)
    output = [0] * n
    for i in range(n):
        count[arr[i]] += 1
    for i in range(1, k + 1):
        count[i] += count[i - 1]
    for i in range(n - 1, -1, -1):
        output[count[arr[i]] - 1] = arr[i]
        count[arr[i]] -= 1
    return output

arrData =  [7, 1, 36, 26, 63, 93, 55, 16, 19, 38, 74, 65, 18, 59, 8, 43, 24, 79, 49, 35, 23, 78, 51, 2, 46, 28, 60, 76, 10, 85, 66, 29, 82, 58, 69, 75, 48, 100, 5, 32, 40, 33, 34, 90, 81, 42, 57, 44, 41, 77]
print("Sebelum sorting\t:", arrData)
print("Setelah sorting\t:", countingSort(arrData))
endTime = time.time()
print(f"Waktu eksekusi\t: {endTime - startTime:.6f} detik")
```

Screenshot output:



```
firdi@Quantum-QF:~/Documents/sorting-algoritm$ /usr/bin/python3 /home/firdi/Documents/
sorting-algoritm/countingSort.py
Sebelum sorting : [7, 1, 36, 26, 63, 93, 55, 16, 19, 38, 74, 65, 18, 59, 8, 43, 24, 79
, 49, 35, 23, 78, 51, 2, 46, 28, 60, 76, 10, 85, 66, 29, 82, 58, 69, 75, 48, 100, 5, 3
2, 40, 33, 34, 90, 81, 42, 57, 44, 41, 77]
Setelah sorting : [1, 2, 5, 7, 8, 10, 16, 18, 19, 23, 24, 26, 28, 29, 32, 33, 34, 35,
36, 38, 40, 41, 42, 43, 44, 46, 48, 49, 51, 55, 57, 58, 59, 60, 63, 65, 66, 69, 74, 75
, 76, 77, 78, 79, 81, 82, 85, 90, 93, 100]
Waktu eksekusi  : 0.000063 detik
```

KESIMPULAN

Algoritma sorting tercepat ialah Counting Sort dengan total waktu eksekusi 0,000063 detik, lalu di lanjut dengan Radix Sort, dan terakhir Quick Sort. Hal ini karena Counting Sort **tidak melakukan perbandingan elemen** seperti Quick Sort. Sebagai gantinya, algoritma ini memanfaatkan array untuk menghitung jumlah kemunculan setiap elemen. Kemudian Counting Sort menjalankan operasi dengan perhitungan dan penyusunan ulang elemen yang sederhana dan langsung, sehingga membuat algortima ini menjadi cepat untuk mengeksekusi pengurutan value di array tersebut.