Go Run Terminal Help

Belajar Python

PermutationDanCombination.py

Materi Python > PermutationDanCombination.py > ...

```python
from itertools import permutations, combinations, combinations_with_replacement

# Himpunan data yang akan diolah
data = [1, 2, 3]
panjang_r = 2 # Menentukan panjang subset yang akan diambil untuk sebagian besar contoh (r=2)

# --- 1. Permutasi (Urutan PENTING) ---

# Contoh 1: Permutasi dari semua elemen (r dihilangkan, P(3, 3))
print("### 1. Permutasi (P(3, 3)) - Semua Urutan Memungkinkan ###")
# permutations(data) menghasilkan semua kemungkinan susunan dari ketiga elemen
perm_all = permutations(data)
for i in perm_all:
    print(i)
# Hasil: (1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)

print("-" * 30)

# Contoh 2: Permutasi dengan Panjang Tertentu (P(3, 2))
print("### 2. Permutasi dengan Panjang Tertentu (P(3, 2)) ###")
# permutations(data, 2) menghasilkan semua urutan 2 elemen dari 3 elemen.
# (1, 2) dan (2, 1) dianggap berbeda.
perm_r = permutations(data, panjang_r)
for i in perm_r:
    print(i)
# Hasil: (1, 2), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2)

print("=" * 30)

# --- 2. Kombinasi (Urutan TIDAK Penting) ---

# Contoh 3: Kombinasi Panjang 2 (C(3, 2))
print("### 3. Kombinasi Tanpa Pengulangan (C(3, 2)) ###")
# combinations(data, 2) menghasilkan semua subset unik 2 elemen.
# (1, 2) dan (2, 1) dianggap sama, hanya (1, 2) yang muncul.
comb = combinations(data, panjang_r)
for i in comb:
```

```python
for i in comb:
    print(i)
# Hasil: (1, 2), (1, 3), (2, 3)

print("-" * 30)

# Contoh 4: Kombinasi dari Daftar yang TIDAK Diurutkan
# Menguji bahwa urutan input tidak memengaruhi output kombinasi yang unik
data_unsorted = [2, 1, 3]
print("### 4. Kombinasi dari Daftar [2, 1, 3] (C(3, 2)) ###")
# combinations mempertahankan urutan relatif elemen dalam subset,
# tetapi menghindari duplikat berdasarkan konten.
comb_unsorted = combinations(data_unsorted, panjang_r)
for i in comb_unsorted:
    print(i)
# Hasil: (2, 1), (2, 3), (1, 3)

print("-" * 30)

# Contoh 5: Kombinasi dengan Pengulangan
print("### 5. Kombinasi Dengan Pengulangan (H(3, 2)) ###")
# combinations_with_replacement(data, 2) memungkinkan elemen yang sama dipilih lebih dari sekali.
comb_rep = combinations_with_replacement(data, panjang_r)
for i in comb_rep:
    print(i)
# Hasil: (1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3)
```

```
ion.py
### 1. Permutasi (P(3, 3)) - Semua Urutan Memungkinkan ###
(1, 2, 3)
(1, 3, 2)
(2, 1, 3)
(2, 3, 1)
(3, 1, 2)
(3, 2, 1)
----------------------------
### 2. Permutasi dengan Panjang Tertentu (P(3, 2)) ###
(1, 2)
(1, 3)
(2, 1)
(2, 3)
(3, 1)
(3, 2)
============================
### 3. Kombinasi Tanpa Pengulangan (C(3, 2)) ###
(1, 2)
(1, 3)
(2, 3)
----------------------------
### 4. Kombinasi dari Daftar [2, 1, 3] (C(3, 2)) ###
(2, 1)
(2, 3)
(1, 3)
----------------------------
### 5. Kombinasi Dengan Pengulangan (H(3, 2)) ###
(1, 1)
(1, 2)
(1, 3)
(2, 2)
(2, 3)
(3, 3)
PS C:\Program Files\Dev\Belajar Python>
```