

# **LAPORAN PRAKTIKUM**

## **“POST TEST 8: Kombinatorik”**

Diajukan untuk memenuhi salah satu praktikum Mata Kuliah Matematika Diskrit yang di  
ampu oleh:

Nur Rochmah Dyah PA, S.T., M.Kom



Disusun Oleh:

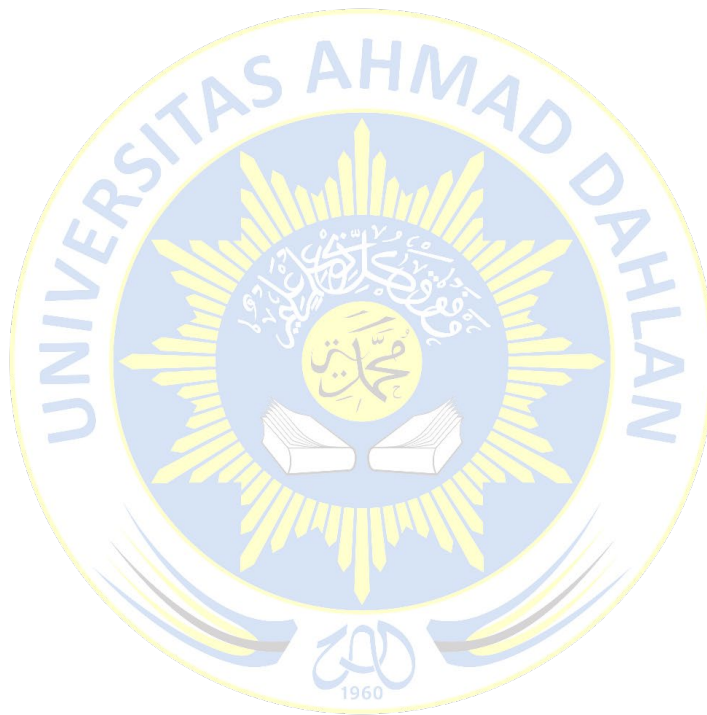
Mohammad Farid Hendianto 2200018401

Selasa 12.00-13.30

**PROGRAM STUDI INFORMATIKA**  
**UNIVERSITAS AHMAD DAHLAN**  
**FAKULTAS TEKNOLOGI INDUSTRI**  
**TAHUN 2023**

## DAFTAR SOAL

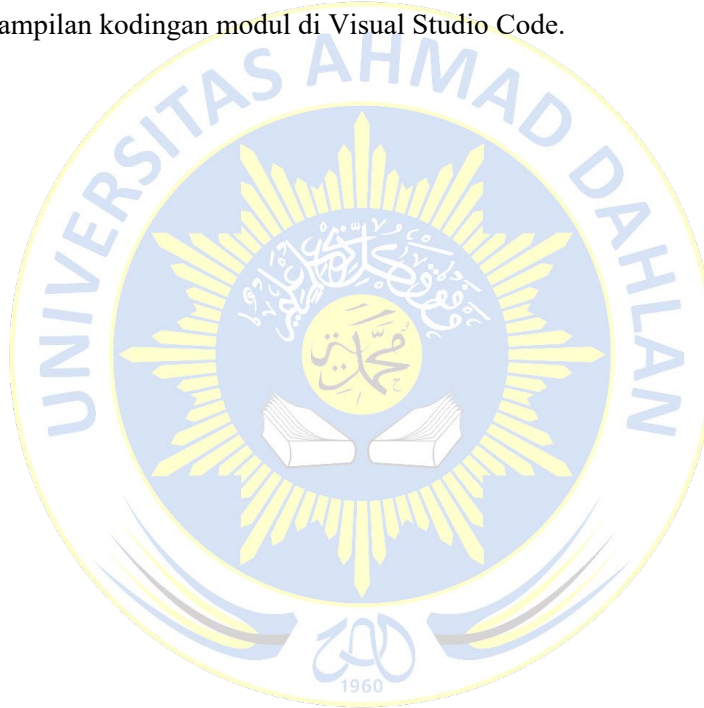
- 1) Modifikasi program permutasi dan kombinasi yang ada dimodul dengan menggunakan fungsi dan di main hanya ada pemanggilan satu fungsi..... 3

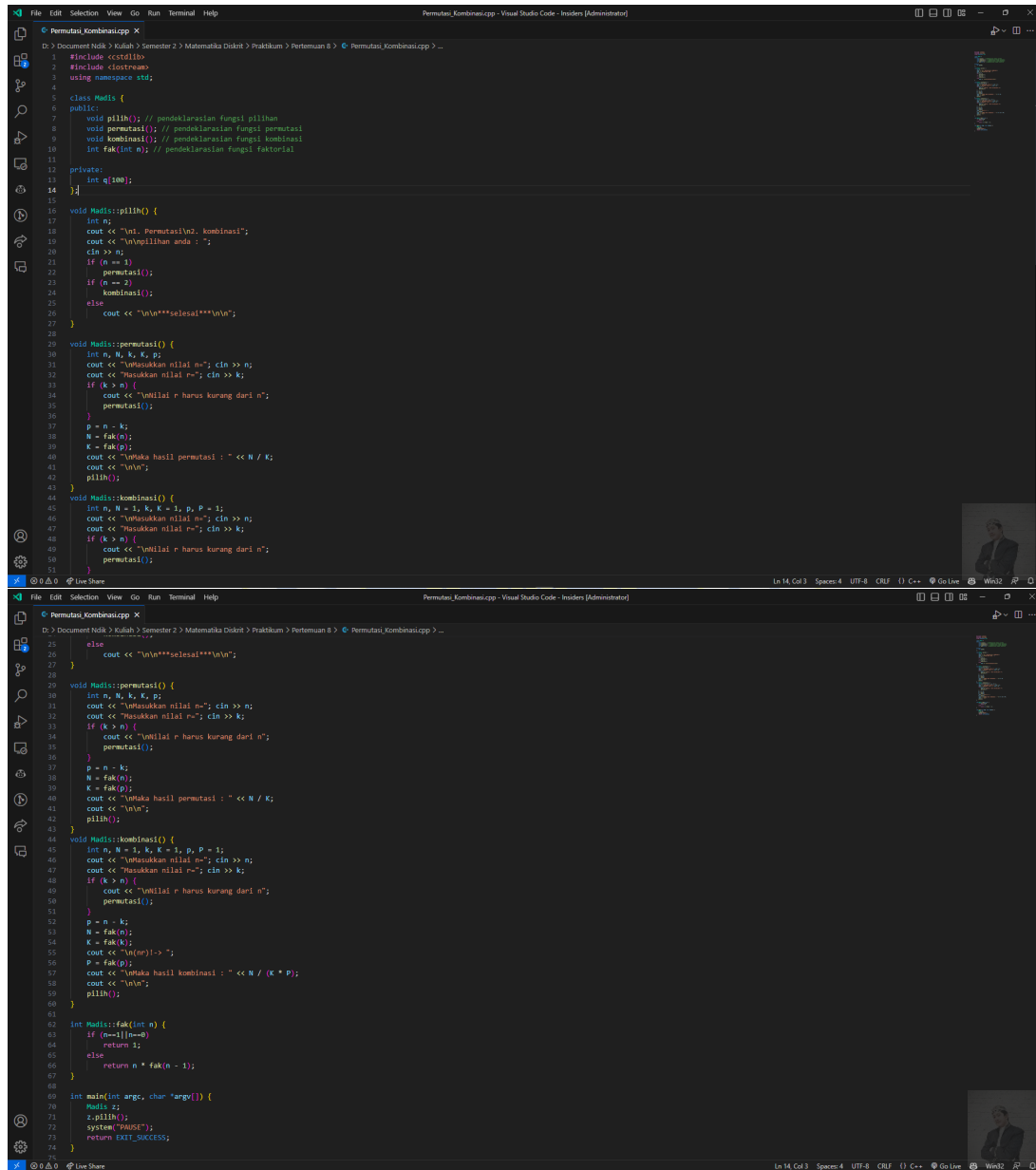


- 1) Modifikasi program permutasi dan kombinasi yang ada dimodul dengan menggunakan fungsi dan di main hanya ada pemanggilan satu fungsi.

kodingan yang ada di modul sebenarnya sudah menggunakan fungsi dan hanya terdapat pemanggilan satu fungsi di dalam main. Fungsi-fungsi yang digunakan yaitu pilih(), permutasi(), kombinasi(), dan fak(). Semua fungsi tersebut dideklarasikan dalam kelas Madis. Di dalam fungsi pilih(), terdapat penggunaan percabangan untuk memilih antara permutasi atau kombinasi. Kemudian, di dalam masing-masing fungsi permutasi dan kombinasi, rumus matematika dihitung menggunakan fungsi fak() yang menghitung nilai faktorial. Setelah selesai menjalankan perhitungan pada fungsi permutasi atau kombinasi, maka fungsi pilih() dipanggil lagi untuk mengulang prosesnya. Pada akhirnya, program akan berakhir setelah pengguna memilih keluar dari program.

Berikut adalah tampilan kodingan modul di Visual Studio Code.





```

1 #include <cstdlib>
2 #include <iostream>
3 using namespace std;
4
5 class Madis {
6 public:
7     void pilih(); // deklarasi fungsi pilihan
8     void permutasi(); // deklarasi fungsi permutasi
9     void kombinasi(); // deklarasi fungsi kombinasi
10    fak(int n); // deklarasi fungsi faktorial
11
12 private:
13     int q[100];
14 }
15
16 void Madis::pilih() {
17     int n;
18     cout << "Masukkan n: ";
19     cin >> n;
20     if (n > 0) {
21         permutasi();
22     }
23     else {
24         kombinasi();
25     }
26     cout << "Selesai";
27 }
28
29 void Madis::permutasi() {
30     int n, k, p;
31     cout << "Masukkan nilai n: ";
32     cin >> n;
33     cout << "Masukkan nilai k: ";
34     cin >> k;
35     if (k > n) {
36         cout << "Nilai k harus kurang dari n";
37         permutasi();
38     }
39     p = n - k;
40     n = fak(n);
41     k = fak(k);
42     cout << "Maka hasil permutasi : " << n / k;
43     cout << "n";
44     pilih();
45 }
46
47 void Madis::kombinasi() {
48     int n, k, p;
49     cout << "Masukkan nilai n: ";
50     cin >> n;
51     cout << "Masukkan nilai k: ";
52     cin >> k;
53     if (k > n) {
54         cout << "Nilai k harus kurang dari n";
55         kombinasi();
56     }
57     p = n - k;
58     n = fak(n);
59     k = fak(k);
60     p = fak(p);
61     cout << "Maka hasil kombinasi : " << n / (k * p);
62     cout << "n";
63     pilih();
64 }
65
66 int Madis::fak(int n) {
67     if (n == 1 || n == 0)
68         return 1;
69     else
70         return n * fak(n - 1);
71 }
72
73 int main(int argc, char *argv[]) {
74     Madis z;
75     z.pilih();
76     system("PAUSE");
77     return EXIT_SUCCESS;
78 }

```

Gambar 1 Tampilan kode yang berada di Visual Studio Code. (Sumber: Penulis)

Sayangnya kodingnya pada modul masih kurang efektif. Berikut adalah kodingan yang ada dimodul dengan menggunakan fungsi saja (tidak perlu menggunakan OOP) dan di main hanya ada pemanggilan satu fungsi. Kemudian, kodingan sudah teroptimasi dan menjadi lebih singkat.

```

1 #include <iostream>
2 using namespace std;
3
4 int factorial(int n) {
5     return (n <= 1) ? 1 : n * factorial(n - 1);
6 }
7
8 bool isValid(int n, int r) {
9     if (r > n) {
10        cout << "Nilai r harus kurang dari n";
11        return false;
12    }
13    if (n < 0 || r < 0) {
14        cout << "Nilai n dan r harus positif";
15        return false;
16    }
17    return true;
18 }
19
20 void getInput(int &n, int &r) {
21     cout << "Masukkan nilai n: "; cin >> n;
22     cout << "Masukkan nilai r: "; cin >> r;
23     if (!isValid(n, r)) getInput(n, r);
24 }
25
26 void permutation(int n, int r) {
27     int result = factorial(n) / factorial(n - r);
28     cout << "Maka Hasil permutasi: " << result << "\n\n";
29 }
30
31 void combination(int n, int r) {
32     int result = factorial(n) / (factorial(r) * factorial(n - r));
33     cout << "Maka hasil kombinasi: " << result << "\n\n";
34 }
35
36 void choose() {
37     system("cls");
38     int choice, n, r;
39     bool isUseStep;
40     cout << "\n1. Permutasi\n2. kombinasi\n3. Keluar";
41     cout << "\n\nPilihan anda : ";
42     cin >> choice;
43
44     switch (choice) {
45         case 1:
46             cout << "\n\nPermutasi\n\n";
47             getInput(n, r);
48             permutation(n, r);
49             break;
50         case 2:
51             cout << "\n\nkombinasi\n\n";
52             combination(n, r);
53             break;
54         case 3:
55             cout << "\n\n***selesai***\n\n";
56             break;
57         default:
58             cout << "\n\nPilihan tidak tersedia\n\n";
59             system("pause");
60             choose();
61             break;
62     }
63     system("pause"); system("cls");
64     choose();
65 }
66
67 int main() {
68     choose();
69     return 0;
70 }

```

Gambar 2 Kodingan yang sudah dimodifikasi lebih efisien dan hanya menggunakan 1 fungsi di mainnya.

Kodingan ini adalah sebuah program konsol sederhana yang dapat menghitung nilai permutasi dan kombinasi berdasarkan input nilai  $n$  dan  $r$  yang dimasukkan oleh pengguna. Program ini terdiri dari beberapa fungsi yang masing-masing memiliki tugas dan cara kerja yang berbeda.

**Fungsi factorial(int n)**

Fungsi ini merupakan fungsi rekursif yang digunakan untuk menghitung nilai faktorial dari suatu bilangan bulat positif  $n$ . Faktorial didefinisikan sebagai perkalian semua bilangan bulat positif dari 1 hingga  $n$ . Jadi,  $\text{factorial}(n) = 1 * 2 * 3 * \dots * (n-1) * n$ .

Cara kerja fungsi ini adalah dengan memeriksa apakah nilai  $n$  kurang dari atau sama dengan 1. Jika benar, maka akan mengembalikan nilai 1 karena faktorial dari 0 dan 1 adalah 1. Jika tidak, maka fungsi akan memanggil dirinya sendiri dengan parameter  $n-1$  dan akan mengalikan hasilnya dengan nilai  $n$ .

**Fungsi isValid(int n, int r)**

Fungsi ini digunakan untuk memeriksa validitas input nilai  $n$  dan  $r$ . Fungsi ini akan mengembalikan nilai true jika nilai  $n$  dan  $r$  valid, dan false jika tidak valid. Validitas input ditentukan oleh dua kondisi:

- $r$  harus kurang dari atau sama dengan  $n$
- $n$  dan  $r$  harus bernilai positif

Jika salah satu atau kedua kondisi tidak terpenuhi, maka fungsi akan menampilkan pesan kesalahan dan mengembalikan nilai false.

**Fungsi getInput(int &n, int &r)**

Fungsi ini digunakan untuk meminta input nilai  $n$  dan  $r$  dari pengguna. Fungsi ini mengambil dua argumen, yaitu variabel  $n$  dan  $r$  yang akan diisi dengan nilai input dari pengguna. Fungsi ini juga memanggil fungsi  $\text{isValid}(n, r)$  untuk memeriksa validitas input. Jika input tidak valid, maka fungsi akan memanggil dirinya sendiri kembali untuk meminta input yang baru.

**Fungsi permutation(int n, int r)**

Fungsi ini digunakan untuk menghitung nilai permutasi dari  $n$  dan  $r$ . Permutasi didefinisikan sebagai jumlah cara-cara pemilihan  $r$  objek dari  $n$  objek yang berbeda. Dalam matematika, notasi permutasi dituliskan sebagai  $P(n, r) = n! / (n-r)!$ .

Cara kerja fungsi ini adalah dengan memanggil fungsi  $\text{factorial}(n)$  dan  $\text{factorial}(n-r)$  untuk menghitung faktorial dari  $n$  dan  $n-r$ . Lalu, hasil faktorial  $n$  akan dibagi dengan hasil faktorial  $n-r$  untuk menghasilkan nilai permutasi. Hasil permutasi kemudian akan ditampilkan ke layar.

**Fungsi combination(int n, int r)**

Fungsi ini digunakan untuk menghitung nilai kombinasi dari n dan r. Kombinasi didefinisikan sebagai jumlah cara-cara pemilihan r objek dari n objek yang berbeda tanpa memperhatikan urutan. Dalam matematika, notasi kombinasi dituliskan sebagai  $C(n,r) = n! / (r!(n-r)!)$ .

Cara kerja fungsi ini hampir sama dengan fungsi permutation(). Fungsi ini juga akan memanggil fungsi factorial(n) dan factorial(r) untuk menghitung faktorial dari n dan r. Namun, hasil faktorial r juga akan dikalikan dengan hasil faktorial (n-r) sebelum membagi hasil faktorial n agar menghasilkan nilai kombinasi. Hasil kombinasi kemudian akan ditampilkan ke layar.

**Fungsi choose()**

Fungsi ini adalah fungsi utama yang menampilkan menu pilihan perhitungan permutasi atau kombinasi. Fungsi ini akan terus berjalan sampai pengguna memilih opsi keluar pada menu pilihan.

Cara kerja fungsi ini adalah sebagai berikut:

- Menggunakan fungsi system("cls") untuk membersihkan layar console.
- Menampilkan menu pilihan perhitungan permutasi atau kombinasi ke layar.
- Meminta input pilihan dari pengguna menggunakan cin.
- Menggunakan switch case untuk memilih perhitungan yang diinginkan oleh pengguna.
- Jika pengguna memilih permutasi atau kombinasi,

**Fungsi main()**

Fungsi main() adalah titik awal program. Pada fungsi main(), hanya terdapat pemanggilan fungsi choose() untuk menampilkan menu pilihan perhitungan permutasi atau kombinasi.



Perbandingan kodingan yang ada di modul dan kodingan kedua:

### **Fungsi-fungsi:**

Kodingan pertama menggunakan fungsi-fungsi yang dideklarasikan dalam kelas, sedangkan kodingan kedua menggunakan fungsi-fungsi yang dideklarasikan di luar kelas.

Pada kodingan kedua, terdapat fungsi `isValid()` yang memeriksa apakah nilai input `n` dan `r` valid atau tidak. Sedangkan pada kodingan pertama, pemeriksaan dilakukan langsung di dalam fungsi permutasi dan kombinasi.

### **Input:**

Pada kodingan pertama, input dimasukkan langsung di dalam fungsi-fungsi permutasi dan kombinasi. Sedangkan pada kodingan kedua, ada fungsi `getInput()` yang meminta input dari pengguna sebelum masuk ke fungsi permutasi atau kombinasi.

### **Perulangan:**

Pada kodingan pertama, perulangan terjadi di dalam fungsi `pilih()` dengan memanggil dirinya sendiri untuk mengulang proses pemilihan. Sedangkan pada kodingan kedua, perulangan terjadi di dalam fungsi `choose()` dengan bantuan rekursi.

### **Penanganan kesalahan:**

Pada kodingan pertama, penanganan kesalahan dilakukan dengan menampilkan pesan kesalahan di dalam fungsi-fungsi permutasi dan kombinasi, kemudian mengulang pemilihan dengan memanggil fungsi `pilih()`.

Pada kodingan kedua, penanganan kesalahan dilakukan dengan menggunakan fungsi `isValid()` yang memeriksa apakah nilai input `n` dan `r` valid atau tidak. Jika tidak valid, maka fungsi `getInput()` dipanggil lagi untuk meminta input yang baru.

### **Tampilan:**

Pada kodingan kedua, terdapat fungsi `system("cls")` yang digunakan untuk membersihkan tampilan layar sebelum menampilkan menu. Namun, pada kodingan pertama, tidak ada penggunaan fungsi ini sehingga menu akan muncul di bawah hasil perhitungan sebelumnya.

Secara keseluruhan, kodingan kedua lebih mudah dipahami karena menggunakan fungsi-fungsi yang dideklarasikan di luar kelas dan memiliki alur logika yang lebih jelas. Selain itu, penanganan kesalahan juga lebih baik karena terdapat fungsi `isValid()` yang memeriksa validitas input sebelum masuk ke fungsi permutasi atau kombinasi. Namun, kodingan pertama juga dapat dianggap baik karena sudah



memenuhi syarat dari soal yaitu menggunakan beberapa fungsi dan hanya satu pemanggilan fungsi di dalam main.

Berikut adalah contoh outputnya.

```

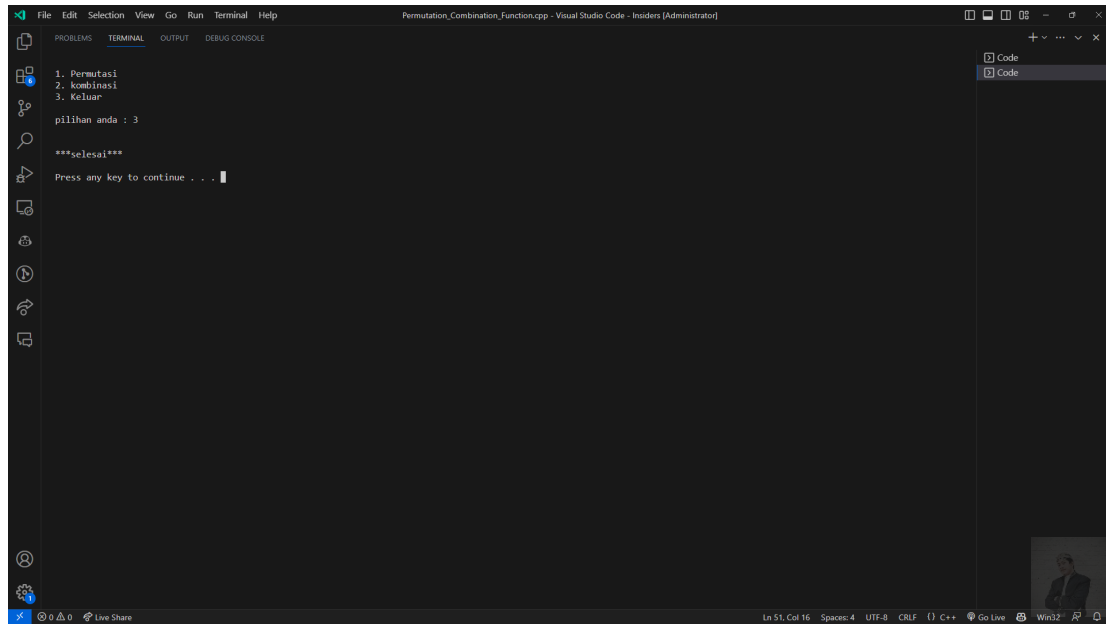
File Edit Selection View Go Run Terminal Help
Permutation_Combination_Function.cpp - Visual Studio Code - Insiders (Administrator)
PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
pilihan anda : 1
Permutasi
Masukkan nilai n: 5
Masukkan nilai r: 2
Maka Hasil permutasi: 20
Press any key to continue . . .

```

```

File Edit Selection View Go Run Terminal Help
Permutation_Combination_Function.cpp - Visual Studio Code - Insiders (Administrator)
PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
1. Permutasi
2. kombinasi
3. Keluar
pilihan anda : 2
Masukkan nilai n: 5
Masukkan nilai r: 2
Kombinasi
Maka hasil kombinasi: 10
Press any key to continue . . .

```



```

File Edit Selection View Go Run Terminal Help
Permutation_Combination_Function.cpp - Visual Studio Code - Insiders (Administrator)

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE

1. Permutasi
2. Kombinasi
3. Keluar

pilihan anda : 3

***selesai***
Press any key to continue . . .
  
```

Gambar 3 Outputnya hampir sama dengan kodingna pertama, tetapi pada kodingan kedua menerapkan clear console. (Sumber: Penulis)

Untuk mengakses kodingan, dapat melihat link Github berikut:

<https://github.com/IRedDragonICY/Matematika-Diskrit>