

# **LAPORAN PRAKTIKUM**

## **“POST TEST 10: GRAF”**

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

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



Disusun Oleh:

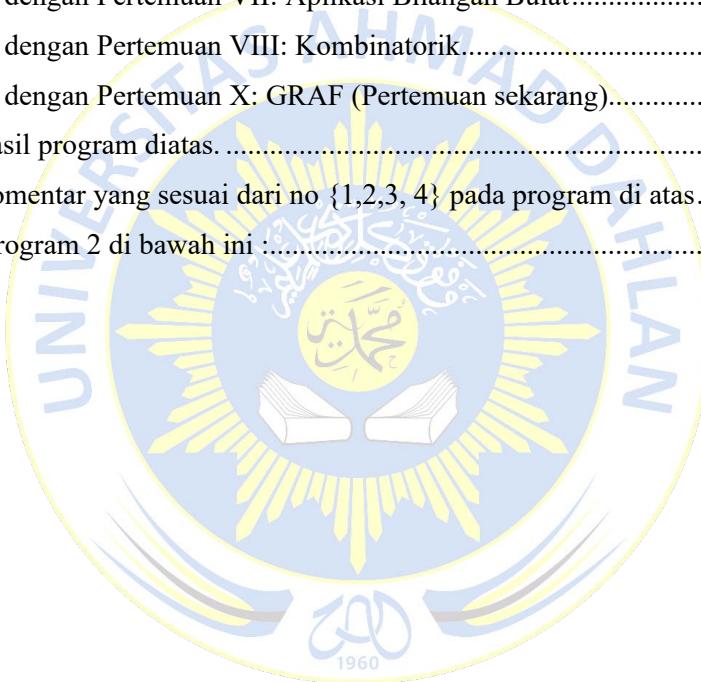
Mohammad Farid Hendianto 2200018401

Selasa 12.00-13.30

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

## Daftar Soal

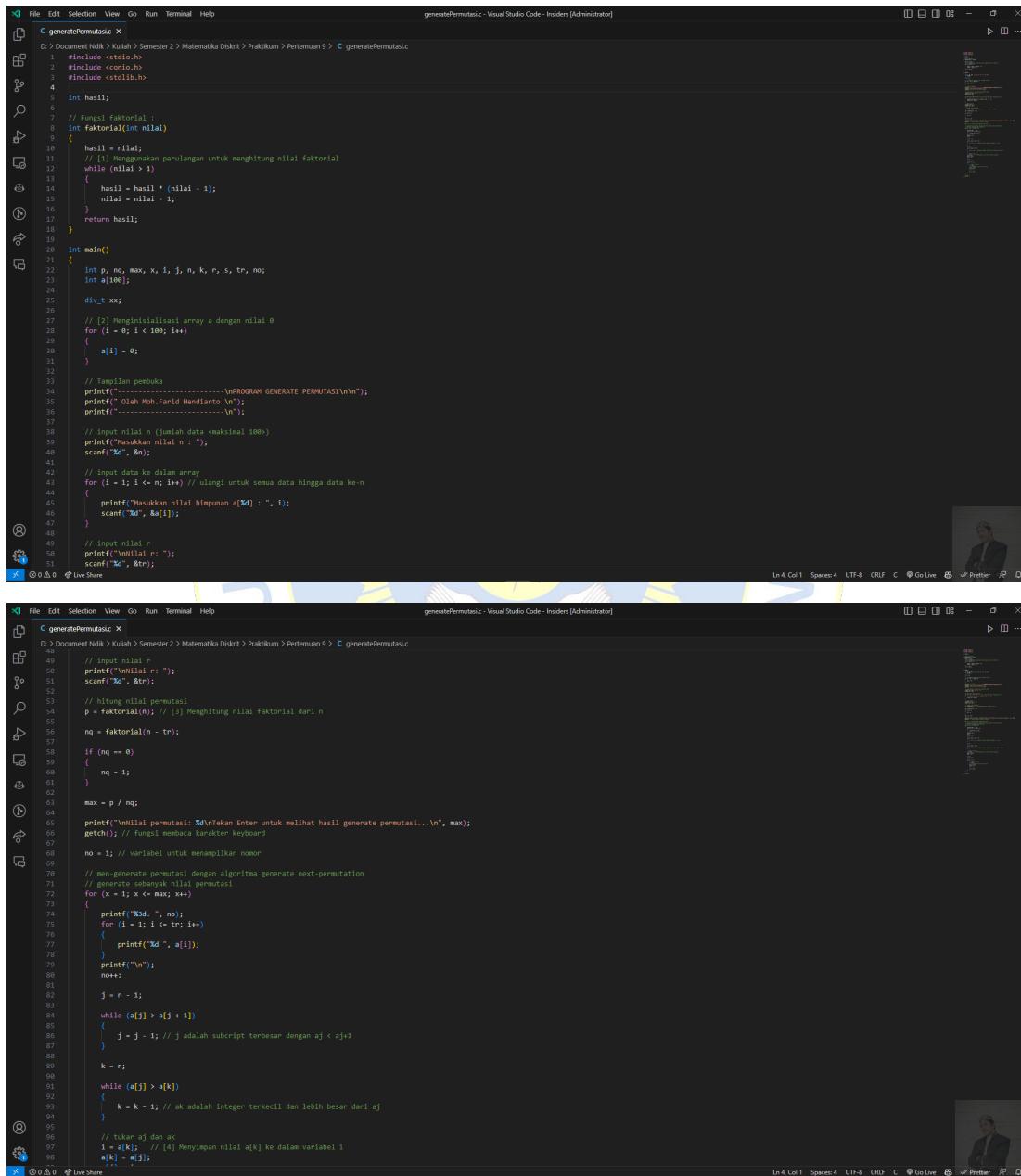
1. Kerjakan program pada halaman 66 kemudian berikan analisis perbedaan dengan program yang ada diawal materi yang ada.....	3
Perbandingan dengan pertemuan I: Himpunan.....	16
Perbandingan dengan Pertemuan II: Relasi .....	19
Perbandingan dengan Pertemuan III: Relasi N-Array .....	21
Perbandingan dengan Pertemuan IV: Fungsi.....	22
Perbandingan dengan Pertemuan V: Fungsi dengan Komposisi Dua Fungsi.....	24
Perbandingan dengan Pertemuan VI: Bilangan Bulat.....	27
Perbandingan dengan Pertemuan VII: Aplikasi Bilangan Bulat.....	29
Perbandingan dengan Pertemuan VIII: Kombinatorik.....	31
Perbandingan dengan Pertemuan X: GRAF (Pertemuan sekarang).....	32
a. Tuliskan hasil program diatas. ....	34
b. Tuliskan komentar yang sesuai dari no {1,2,3, 4} pada program di atas.....	37
c. Ketikkan program 2 di bawah ini !.....	40



1. Kerjakan program pada halaman 66 kemudian berikan analisis perbedaan dengan program yang ada diawal materi yang ada.

Pada halaman 66 merupakan program menghitung permutasi.

Berikut adalah tampilan kodingan di Visual Studio Code.



```

File Edit Selection View Go Run Terminal Help generatePermutasi.c - Visual Studio Code - Insiders [Administrator]
D:\Document Ndk\Kuliah> Semester 2 > Matematika Diskrit > Praktikum > Pertemuan 9 > C generatePermutasi.c

1 //include <stdio.h>
2 //include <comio.h>
3 //include <cslib.h>
4
5 int hasil;
6
7 // Fungsi faktorial :
8 int faktorial(int nilai)
9 {
10     hasil = nilai;
11     for (int i = 1; i < nilai; i++)
12     {
13         hasil = hasil * (nilai - i);
14         nilai = nilai - 1;
15     }
16     return hasil;
17 }
18
19 int main()
20 {
21     int p, nq, max, x, i, j, n, k, r, s, tr, no;
22     int a[100];
23
24     div_t xx;
25
26     // [2] Membuat array a dengan nilai 0
27     for (i = 0; i < 100; i++)
28     {
29         a[i] = 0;
30     }
31
32     // Tampilan pendek
33     printf("-----\nPROGRAM GENERATE PERMUTASI\n-----");
34     printf(" Oleh Moh.Farid Hendianto\n");
35     printf("-----\n");
36
37     // Input nilai n (jumlah data maksimal 100)
38     printf("Masukkan nilai n : ");
39     scanf("%d", &n);
40
41     // Input data ke dalam array
42     for (i = 1; i < n; i++) // ulangi untuk semua data hingga data ke-i
43     {
44         printf("Masukkan nilai himpunan a[%d] : ", i);
45         scanf("%d", &a[i]);
46     }
47
48     // Input nilai r
49     printf("Masukkan nilai r : ");
50     scanf("%d", &r);
51
52     // input nilai nq
53     //-----#
54     //-----#
55     //-----#
56     //-----#
57     //-----#
58     //-----#
59     //-----#
60     //-----#
61     //-----#
62     //-----#
63     //-----#
64     //-----#
65     //-----#
66     //-----#
67     //-----#
68     //-----#
69     //-----#
70     //-----#
71     //-----#
72     //-----#
73     //-----#
74     //-----#
75     //-----#
76     //-----#
77     //-----#
78     //-----#
79     //-----#
80     //-----#
81     //-----#
82     //-----#
83     //-----#
84     //-----#
85     //-----#
86     //-----#
87     //-----#
88     //-----#
89     //-----#
90     //-----#
91     //-----#
92     //-----#
93     //-----#
94     //-----#
95     //-----#
96     //-----#
97     //-----#
98     //-----#

```

```

File Edit Selection View Go Run Terminal Help generatePermutasi.c - Visual Studio Code - Insiders [Administrator]
D:\Document Ndk\Kuliah> Semester 2 > Matematika Diskrit > Praktikum > Pertemuan 9 > C generatePermutasi.c

49 // input nilai r
50 printf("Masukkan nilai r : ");
51 scanf("%d", &r);
52
53 // hitung nilai permutasi
54 p = faktorial(n); // [3] Menghitung nilai faktorial dari n
55
56 nq = faktorial(n - r);
57
58 if (nq == 0)
59 {
60     nq = 1;
61 }
62
63 max = p / nq;
64
65 printf("Nilai permutasi: %d\nTekan Enter untuk melihat hasil generate permutasi...\n", max);
66 getch(); // fungsi membaca karakter keyboard
67
68 no = 1; // variabel untuk menampilkan nomor
69
70 // non-generate permutasi dengan algoritma generate next-permutation
71 // generate selama nilai permutasi
72 for (x = 1; x <= max; x++)
73 {
74     printf("X%d : ", no);
75     for (i = 1; i < r; i++)
76     {
77         printf(" %d ", a[i]);
78     }
79     printf("\n");
80     no++;
81
82     j = n - 1;
83
84     while (a[j] > a[j + 1])
85     {
86         j = j - 1; // j adalah subscript terbesar dengan aj < aj+1
87     }
88
89     k = n;
90
91     while (a[j] > a[k])
92     {
93         k = k - 1; // ak adalah integer terkecil dan lebih besar dari aj
94     }
95
96     // tukan aj dan ak
97     i = a[k]; // [4] Menyimpan nilai a[k] ke dalam variabel i
98     a[k] = a[j];

```



```

File Edit Selection View Go Run Terminal Help
D:\Document Nidik\Xulah> Semester 2> Matematika Diskrit > Praktikum > Pertemuan 9 > C_generatePermutasi.c
C_generatePermutasi.c - Visual Studio Code - Insiders [Administrator]
78 // men-generate permutasi dengan algoritma generate next-permutation
79 // generate sebanyak nilai permutasi
80 for (x = 1; x <= max; x++)
81 {
82     printf("%d", x);
83     for (i = 1; i <= tr; i++)
84     {
85         printf(" %d ", a[i]);
86     }
87     printf("\n");
88     no++;
89 }
90 j = n - 1;
91 while (a[j] > a[j + 1])
92 {
93     j = j - 1; // j adalah subscript terbesar dengan a[j] < a[j+1]
94 }
95 k = n;
96 while (a[j] > a[k])
97 {
98     k = k - 1; // ak adalah integer terkecil dan lebih besar dari aj
99 }
100 a[k] = a[j];
101 a[j] = a[k];
102 r = n;
103 s = j + 1;
104 while (r > s)
105 {
106     // tukar ar dan as
107     i = a[r];
108     // [5] Hemukar nilai a[r] dan a[s]
109     a[r] = a[s];
110     a[s] = i;
111     r = r - 1;
112     s = s + 1;
113 }
114 }
115 getch();
116 }
117 }
118 }

L 4 C 1 Spacers 4 UTF-8 CR/LF C ⌂ Go Live ⌂ Preferences ⌂ D

```

Gambar 1 Tampilan kodingan penerapan kombinatorik di Visual Studio Code. (Sumber: Penulis)

Berikut tampilan source code yang sesuai dengan modul



```
1 #include <stdio.h>
2 #include <conio.h>
3 #include <stdlib.h>
4
5 int hasil;
6
7 // Fungsi faktorial :
8 int faktorial(int nilai)
9 {
10     hasil = nilai;
11     // [1] Menggunakan perulangan untuk menghitung nilai faktorial
12     while (nilai > 1)
13     {
14         hasil = hasil * (nilai - 1);
15         nilai = nilai - 1;
16     }
17     return hasil;
18 }
19
20 int main()
21 {
22     int p, nq, max, x, i, j, n, k, r, s, tr, no;
23     int a[100];
24
25     div_t xx;
26
27     // [2] Menginisialisasi array a dengan nilai 0
28     for (i = 0; i < 100; i++)
29     {
30         a[i] = 0;
31     }
32
33     // Tampilan pembuka
34     printf("-----\nPROGRAM GENERATE PERMUTASI\n\n");
35     printf(" Oleh Moh.Farid Hendianto \n");
36     printf("-----\n");
37
38     // input nilai n (jumlah data <maksimal 100>)
39     printf("Masukkan nilai n : ");
40     scanf("%d", &n);
41
42     // input data ke dalam array
43     for (i = 1; i <= n; i++) // ulangi untuk semua data hingga data ke-n
44     {
45         printf("Masukkan nilai himpunan a[%d] : ", i);
46         scanf("%d", &a[i]);
47     }
48
49     // input nilai r
50     printf("\nNilai r: ");
51     scanf("%d", &r);
52
53     // hitung nilai permutasi
54     p = faktorial(n); // [3] Menghitung nilai faktorial dari n
55
56     nq = faktorial(n - r);
57
58     if (nq == 0)
59     {
60         nq = 1;
61     }
```

```

70      // men-generate permutasi dengan algoritma generate next-permutation
71      // generate sebanyak nilai permutasi
72      for (x = 1; x <= max; x++)
73      {
74          printf("= ", no);
75          for (i = 1; i <= tr; i++)
76          {
77              printf("%d ", a[i]);
78          }
79          printf("\n");
80          no++;
81      }
82      j = n - 1;
83
84      while (a[j] > a[j + 1])
85      {
86          j = j - 1; // j adalah subscript terbesar dengan aj < aj+1
87      }
88
89      k = n;
90
91      while (a[j] > a[k])
92      {
93          k = k - 1; // ak adalah integer terkecil dan lebih besar dari aj
94      }
95
96      // tukar aj dan ak
97      i = a[k]; // [4] Menyimpan nilai a[k] ke dalam variabel i
98      a[k] = a[j];
99      a[j] = i;
100
101     r = n;
102     s = j + 1;
103
104     while (r > s)
105     {
106         // tukar ar dan as
107         i = a[r];
108         // [5] Menukar nilai a[r] dan a[s]
109         a[r] = a[s];
110         a[s] = i;
111
112         r = r - 1;
113         s = s + 1;
114     }
115 }
116
117 getch();
118 }
```

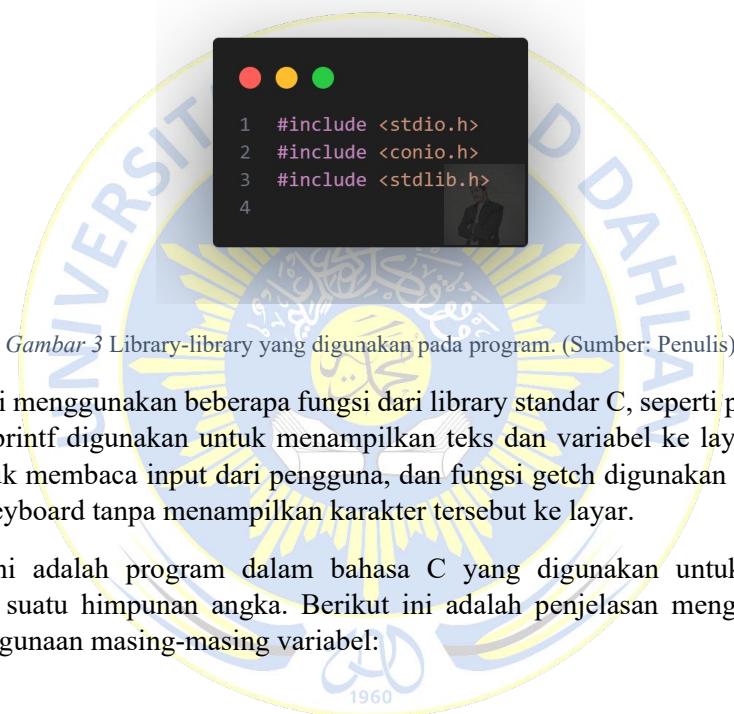
Gambar 2 Kodingan yang sesuai pada modul. (Sumber: Penulis)

Program ini adalah program dalam bahasa C yang digunakan untuk menghasilkan permutasi dari suatu himpunan angka.

Tujuan dari program ini adalah untuk menghasilkan permutasi dari suatu himpunan angka. Fungsi utama dari program ini adalah untuk menghitung nilai permutasi dan menampilkan hasil permutasi tersebut. Program ini menggunakan algoritma generate next-permutation untuk menghasilkan permutasi.

Berikut adalah langkah-langkah yang dilakukan oleh program:

- 1) Menginisialisasi array a dengan nilai 0.
- 2) Menampilkan tampilan pembuka dan meminta input nilai n (jumlah angka yang akan dipermutasi) dan nilai r (jumlah angka yang akan diambil dari himpunan angka).
- 3) Meminta input angka-angka yang akan dipermutasi dan menyimpannya dalam array a.
- 4) Menghitung nilai permutasi dengan menggunakan fungsi faktorial.
- 5) Menampilkan jumlah permutasi yang mungkin dan menunggu input dari pengguna untuk melanjutkan.
- 6) Menghasilkan permutasi dengan menggunakan algoritma generate next-permutation dan menampilkan hasil permutasi tersebut.



Gambar 3 Library-library yang digunakan pada program. (Sumber: Penulis)

Program ini menggunakan beberapa fungsi dari library standar C, seperti printf, scanf, dan getch. Fungsi printf digunakan untuk menampilkan teks dan variabel ke layar, fungsi scanf digunakan untuk membaca input dari pengguna, dan fungsi getch digunakan untuk membaca karakter dari keyboard tanpa menampilkan karakter tersebut ke layar.

Program ini adalah program dalam bahasa C yang digunakan untuk menghasilkan permutasi dari suatu himpunan angka. Berikut ini adalah penjelasan mengenai inisialisasi variabel dan kegunaan masing-masing variabel:

```

20 int main()
21 {
22     int p, nq, max, x, i, j, n, k, r, s, tr, no;
23     int a[100];
24
25     div_t xx;
26

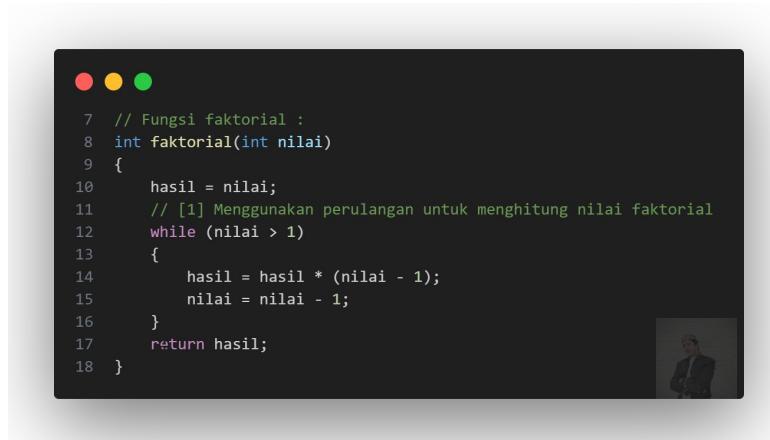
```

Gambar 4 Inisialisasi variabel pada program. (Sumber: Penulis)

- 1) int p, nq, max, x, i, j, n, k, r, s, tr, no; - Inisialisasi variabel-variabel bertipe integer yang akan digunakan dalam program.
  - p: Variabel untuk menyimpan hasil permutasi.
  - nq: Variabel untuk menyimpan hasil faktorial dari ( $n - tr$ ).
  - max: Variabel untuk menyimpan jumlah maksimum permutasi yang akan dihasilkan.
  - x: Variabel untuk mengontrol iterasi dalam loop permutasi.
  - i, j, k, r, s: Variabel indeks dan bantuan untuk loop dan operasi dalam program.
  - n: Variabel untuk menyimpan jumlah elemen dalam himpunan.
  - tr: (target r) Variabel untuk menyimpan nilai r yang merupakan jumlah elemen yang akan diambil dalam permutasi.
  - no: Variabel untuk menampilkan nomor urut permutasi yang dihasilkan.
- 2) int a[100]; - Inisialisasi array a dengan ukuran 100 yang akan digunakan untuk menyimpan elemen-elemen himpunan angka.
- 3) div\_t xx; - Inisialisasi variabel xx dengan tipe data div\_t. Namun, dalam program ini, variabel xx tidak digunakan sama sekali, jadi sebenarnya tidak perlu diinisialisasi.
- 4) int hasil; - Inisialisasi variabel hasil dengan tipe data integer secara global

Program ini akan meminta input jumlah elemen dalam himpunan (n), elemen-elemen himpunan, dan nilai r (jumlah elemen yang akan diambil dalam permutasi). Kemudian, program akan menghitung jumlah permutasi yang mungkin dan menampilkan hasil permutasi tersebut. Dalam program tersebut, masing-masing variabel memiliki peran yang berbeda dalam menghitung dan menampilkan permutasi dari himpunan angka. Variabel-variabel tersebut digunakan dalam berbagai operasi, seperti perhitungan faktorial, perhitungan jumlah permutasi, dan proses permutasi itu sendiri.

Program tersebut menggunakan fungsi faktorial. Berikut adalah penjelasan pada fungsi faktorial.



```

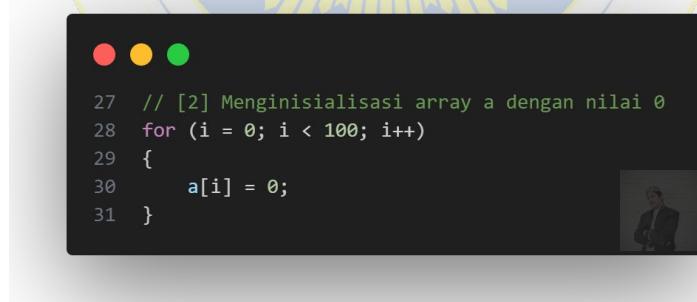
7 // Fungsi faktorial :
8 int faktorial(int nilai)
9 {
10     hasil = nilai;
11     // [1] Menggunakan perulangan untuk menghitung nilai faktorial
12     while (nilai > 1)
13     {
14         hasil = hasil * (nilai - 1);
15         nilai = nilai - 1;
16     }
17     return hasil;
18 }
```

Gambar 5 Fungsi faktorial. (Sumber: Penulis)

Fungsi faktorial adalah sebuah fungsi yang menghitung nilai faktorial dari sebuah bilangan bulat. Faktorial dari sebuah bilangan bulat adalah hasil perkalian dari bilangan tersebut dengan semua bilangan bulat positif yang lebih kecil darinya.

Fungsi faktorial di atas mengambil satu parameter, yaitu nilai, yang merupakan bilangan bulat yang akan dihitung faktorialnya. Fungsi ini menggunakan perulangan while untuk menghitung nilai faktorial dari bilangan tersebut. Setiap iterasi, nilai faktorial dikalikan dengan nilai - 1 dan nilai dikurangi 1. Perulangan berhenti ketika nilai kurang dari atau sama dengan 1. Hasil faktorial kemudian dikembalikan oleh fungsi.

Berikut adalah penjelasan alur program lebih lengkap:

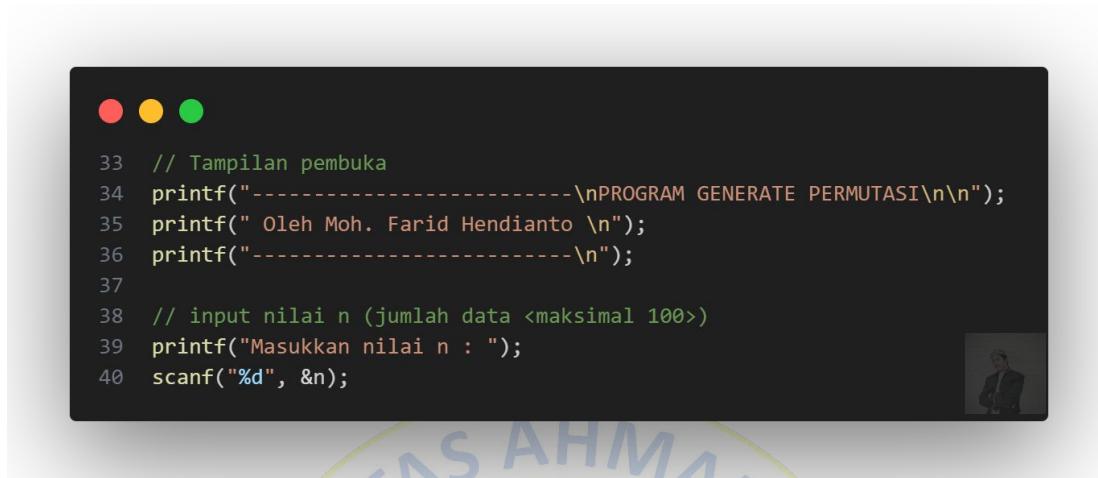


```

27 // [2] Menginisialisasi array a dengan nilai 0
28 for (i = 0; i < 100; i++)
29 {
30     a[i] = 0;
31 }
```

Gambar 6 Inisialisasi array a dengan value 0. (Sumber: Penulis)

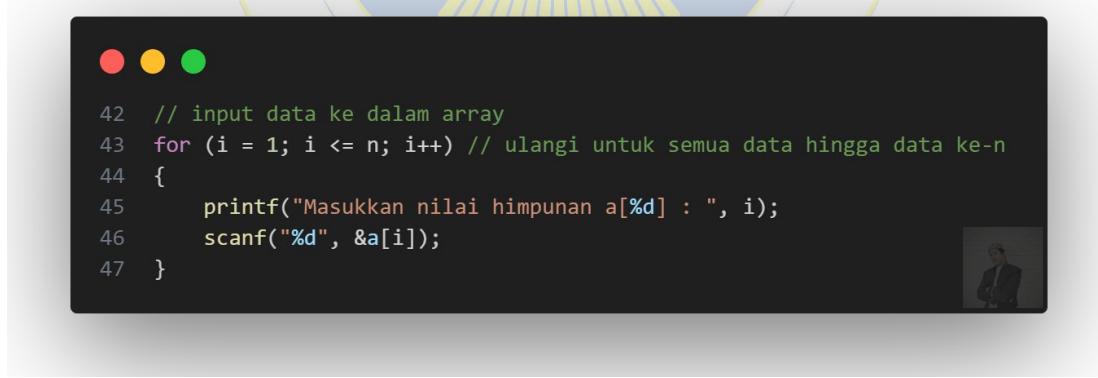
Kodingan tersebut merupakan sebuah loop for yang digunakan untuk menginisialisasi setiap elemen pada array a dengan nilai 0. Loop tersebut akan berjalan sebanyak 100 kali, dimulai dari indeks ke-0 hingga indeks ke-99. Setiap kali loop dijalankan, nilai pada elemen array a pada indeks yang sedang diiterasi akan diubah menjadi 0.



```
33 // Tampilan pembuka
34 printf("-----\nPROGRAM GENERATE PERMUTASI\n\n");
35 printf(" Oleh Moh. Farid Hendianto \n");
36 printf("-----\n");
37
38 // input nilai n (jumlah data <maksimal 100>)
39 printf("Masukkan nilai n : ");
40 scanf("%d", &n);
```

Gambar 7 Output tampilan pertama, dan input nilai n. (Sumber: Penulis)

Kode tersebut merupakan bagian dari program generate permutasi yang ditulis dalam bahasa C. Bagian pertama dari kode tersebut adalah tampilan pembuka program yang menampilkan judul program dan nama pembuat program. Bagian kedua dari kode tersebut adalah input nilai n yang dimasukkan oleh pengguna melalui keyboard. Nilai n tersebut merupakan jumlah data yang akan digenerate permutasinya. Nilai n harus berupa bilangan bulat dan tidak boleh lebih dari 100. Nilai n yang dimasukkan oleh pengguna akan disimpan dalam variabel n.



```
42 // input data ke dalam array
43 for (i = 1; i <= n; i++) // ulangi untuk semua data hingga data ke-n
44 {
45     printf("Masukkan nilai himpunan a[%d] : ", i);
46     scanf("%d", &a[i]);
47 }
```

Gambar 8 Input data ke dalam array,. (Sumber: Penulis)

Kode tersebut merupakan bagian dari program untuk menghasilkan permutasi dari suatu himpunan. Kode tersebut bertujuan untuk meminta input nilai dari pengguna dan menyimpannya ke dalam array a.

Pertama, program akan melakukan perulangan for sebanyak n kali, di mana n adalah jumlah elemen yang akan diisi ke dalam array. Setiap kali melakukan perulangan, program akan menampilkan pesan "Masukkan nilai himpunan a[i] : " di layar, di mana i adalah indeks dari elemen array yang akan diisi. Kemudian, program akan menunggu input dari pengguna menggunakan fungsi scanf(). Input yang diberikan oleh pengguna akan disimpan ke dalam elemen array a[i]. Setelah semua elemen array terisi, program akan melanjutkan ke bagian selanjutnya untuk menghasilkan permutasi dari himpunan tersebut.



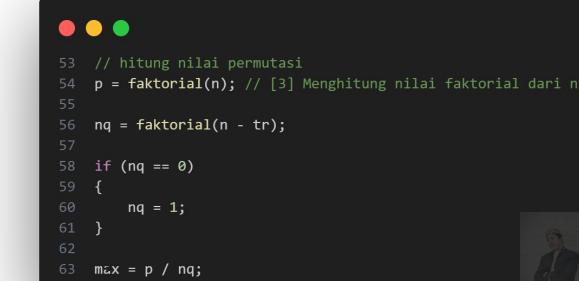
```

49 // input nilai r
50 printf("\nNilai r: ");
51 scanf("%d", &r);

```

Gambar 9 Menginput nilai r. (Sumber: Penulis)

Kode tersebut bertujuan untuk meminta input nilai r dari pengguna dan menyimpannya ke dalam variabel r. Pertama, program akan menampilkan pesan "Nilai r: " di layar menggunakan fungsi printf(). Pesan tersebut bertujuan untuk meminta pengguna untuk memasukkan nilai r. Kemudian, program akan menunggu input dari pengguna menggunakan fungsi scanf(). Input yang diberikan oleh pengguna akan disimpan ke dalam variabel r menggunakan operator & untuk mengambil alamat memori dari variabel tersebut. Setelah nilai r berhasil dimasukkan oleh pengguna dan disimpan ke dalam variabel r, program akan melanjutkan ke bagian selanjutnya untuk menghasilkan permutasi dari himpunan.



```

53 // hitung nilai permutasi
54 p = faktorial(n); // [3] Menghitung nilai faktorial dari n
55
56 nq = faktorial(n - tr);
57
58 if (nq == 0)
59 {
60     nq = 1;
61 }
62
63 max = p / nq;

```

Gambar 10 Perhitungan nilai permutasi. (Sumber: Penulis)

Kode tersebut bertujuan untuk menghitung nilai permutasi dari suatu himpunan dengan menggunakan rumus permutasi. program akan memanggil fungsi faktorial(n) untuk menghitung nilai faktorial dari n. Hasil faktorial dari n akan disimpan ke dalam variabel p.

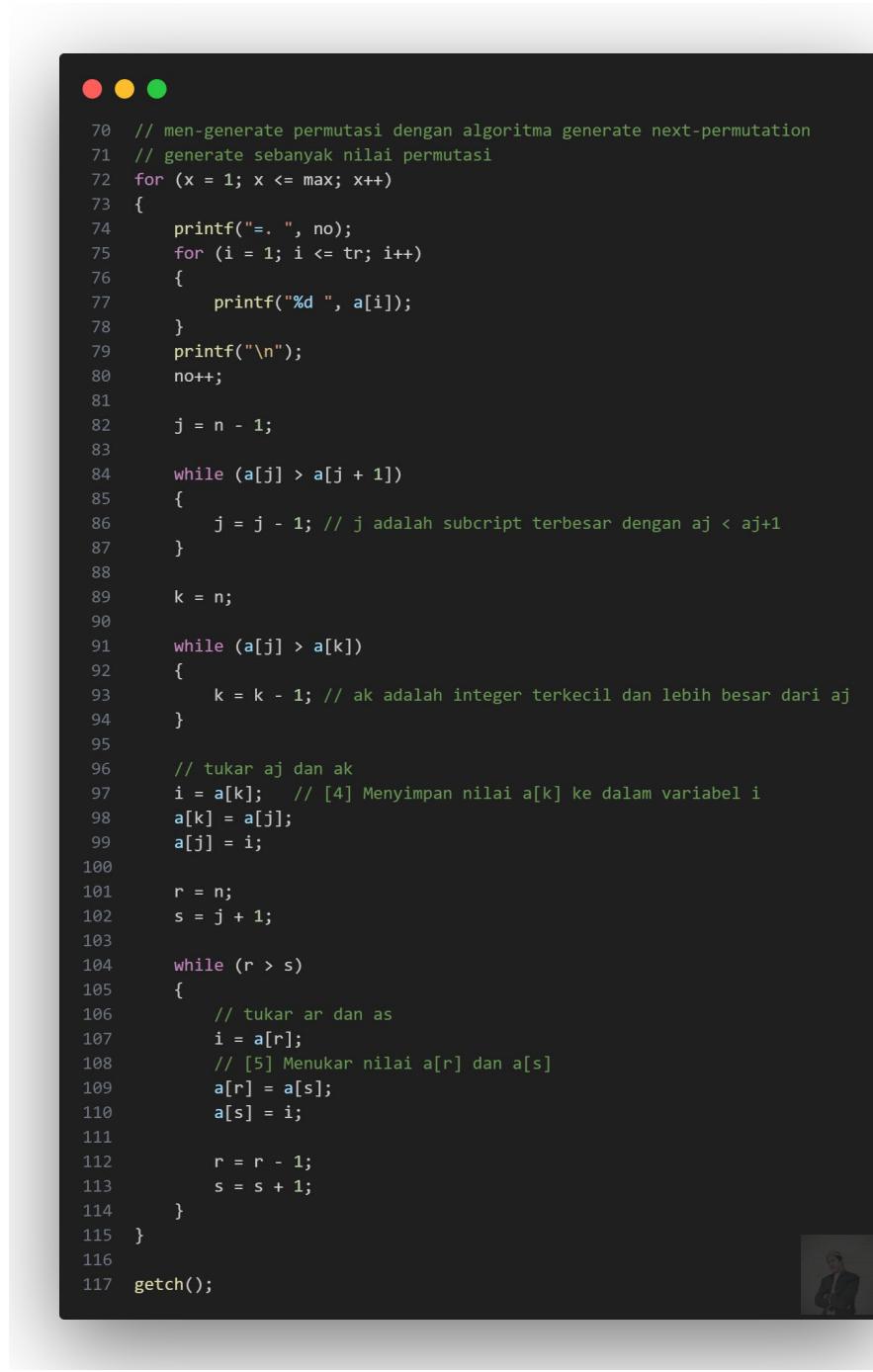
Faktorial dari n digunakan untuk menghitung jumlah permutasi dari himpunan. Selanjutnya, program akan menghitung nilai faktorial dari  $n - r$  menggunakan fungsi faktorial( $n - r$ ). Hasil faktorial dari  $n - r$  akan disimpan ke dalam variabel  $nq$ . Variabel  $nq$  digunakan untuk menghitung jumlah permutasi yang memenuhi syarat nilai  $r$  yang dimasukkan oleh pengguna. Kemudian, program akan melakukan pengecekan apakah nilai  $nq$  sama dengan 0. Jika nilai  $nq$  sama dengan 0, maka variabel  $nq$  akan diubah nilainya menjadi 1. Hal ini dilakukan untuk menghindari terjadinya pembagian dengan nilai 0 pada perhitungan nilai max. Terakhir, program akan menghitung nilai max dengan membagi nilai  $p$  dengan nilai  $nq$ . Nilai max merupakan jumlah permutasi yang memenuhi syarat nilai  $r$  yang dimasukkan oleh pengguna.



```
65 printf("\nNilai permutasi: %d\nTekan Enter untuk melihat hasil generate permutasi...\n", max);
66 getch(); // fungsi membaca karakter keyboard
67
68 no = 1; // variabel untuk menampilkan nomor
69
```

Gambar 11 Menampilkan nilai permutasi. (Sumber: Penulis)

Kode tersebut bertujuan untuk menampilkan nilai permutasi yang telah dihitung sebelumnya dan meminta pengguna untuk menekan tombol Enter untuk melihat hasil generate permutasi. Pertama, program akan menampilkan pesan "Nilai permutasi: " di layar menggunakan fungsi printf(). Pesan tersebut akan menampilkan nilai permutasi yang telah dihitung sebelumnya dan disimpan ke dalam variabel max. Nilai max merupakan jumlah permutasi yang memenuhi syarat nilai  $r$  yang dimasukkan oleh pengguna. Kemudian, program akan menampilkan pesan "Tekan Enter untuk melihat hasil generate permutasi..." di layar. Pesan tersebut bertujuan untuk meminta pengguna untuk menekan tombol Enter untuk melihat hasil generate permutasi. Setelah itu, program akan menunggu pengguna menekan tombol Enter menggunakan fungsi getch(). Fungsi getch() akan membaca karakter keyboard yang ditekan oleh pengguna dan mengembalikan nilai karakter tersebut. Setelah pengguna menekan tombol Enter, program akan melanjutkan ke bagian selanjutnya untuk menghasilkan permutasi dari himpunan. Selanjutnya, program akan menginisialisasi variabel no dengan nilai 1. Variabel no digunakan untuk menampilkan nomor urut dari setiap permutasi yang dihasilkan.



```
70 // men-generate permutasi dengan algoritma generate next-permutation
71 // generate sebanyak nilai permutasi
72 for (x = 1; x <= max; x++)
73 {
74     printf("=. ", no);
75     for (i = 1; i <= tr; i++)
76     {
77         printf("%d ", a[i]);
78     }
79     printf("\n");
80     no++;
81
82     j = n - 1;
83
84     while (a[j] > a[j + 1])
85     {
86         j = j - 1; // j adalah subscript terbesar dengan aj < aj+1
87     }
88
89     k = n;
90
91     while (a[j] > a[k])
92     {
93         k = k - 1; // ak adalah integer terkecil dan lebih besar dari aj
94     }
95
96     // tukar aj dan ak
97     i = a[k]; // [4] Menyimpan nilai a[k] ke dalam variabel i
98     a[k] = a[j];
99     a[j] = i;
100
101    r = n;
102    s = j + 1;
103
104    while (r > s)
105    {
106        // tukar ar dan as
107        i = a[r];
108        // [5] Menukar nilai a[r] dan a[s]
109        a[r] = a[s];
110        a[s] = i;
111
112        r = r - 1;
113        s = s + 1;
114    }
115 }
116 getch();
```

Gambar 12 Menggenerate nilai permutasi. (Sumber: Penulis)

Kode tersebut bertujuan untuk menghasilkan permutasi dari himpunan dengan menggunakan algoritma generate next-permutation.

Pertama, program akan melakukan perulangan for sebanyak max kali, di mana max merupakan jumlah permutasi yang telah dihitung sebelumnya. Setiap kali melakukan perulangan, program akan menampilkan nomor urut dari permutasi menggunakan fungsi printf() dan variabel no. Selanjutnya, program akan melakukan perulangan for sebanyak tr kali, di mana tr merupakan nilai r yang dimasukkan oleh pengguna. Setiap kali melakukan perulangan, program akan menampilkan elemen array a dengan indeks i menggunakan fungsi printf(). Setelah menampilkan elemen array a, program akan menghitung nilai j dengan mengurangi n dengan 1. Selanjutnya, program akan melakukan perulangan while untuk mencari nilai j yang merupakan subscript terbesar dengan  $a[j] < a[j+1]$ . Setelah nilai j ditemukan, program akan menghitung nilai k dengan menginisialisasi k dengan nilai n. Selanjutnya, program akan melakukan perulangan while untuk mencari nilai k yang merupakan integer terkecil dan lebih besar dari  $a[j]$ . Setelah nilai k ditemukan, program akan menukar nilai  $a[j]$  dengan nilai  $a[k]$  menggunakan variabel i sebagai variabel penampung sementara. Selanjutnya, program akan menginisialisasi nilai r dengan n dan nilai s dengan  $j+1$ . Kemudian, program akan melakukan perulangan while untuk menukar nilai  $a[r]$  dengan nilai  $a[s]$  hingga nilai r lebih kecil atau sama dengan nilai s.

Setelah selesai menghasilkan permutasi, program akan menunggu pengguna menekan tombol Enter menggunakan fungsi getch(). Fungsi getch() akan membaca karakter keyboard yang ditekan oleh pengguna.

Berikut adalah outputnya.

*Gambar 13 Output programnya. (Sumber: Penulis)*

Output program tersebut adalah hasil dari program generate permutasi yang telah dijalankan dengan input nilai n dan himpunan a yang dimasukkan oleh pengguna.

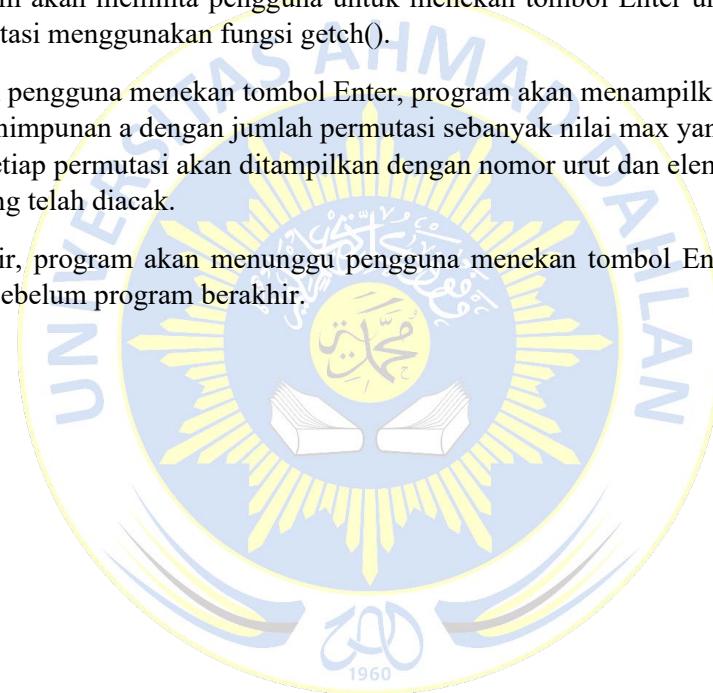
Program akan menampilkan pesan "PROGRAM GENERATE PERMUTASI" dan "Oleh Moh. Farid Hendianto" di layar sebagai header dari program.

Selanjutnya, program akan meminta pengguna untuk memasukkan nilai n dan himpunan a menggunakan fungsi printf() dan scanf(). Setelah nilai n dan himpunan a dimasukkan, program akan meminta pengguna untuk memasukkan nilai r menggunakan fungsi printf() dan scanf().

Setelah nilai r dimasukkan, program akan menghitung nilai permutasi menggunakan rumus permutasi dan menampilkan nilai permutasi tersebut di layar menggunakan fungsi printf(). Program akan meminta pengguna untuk menekan tombol Enter untuk melihat hasil generate permutasi menggunakan fungsi getch().

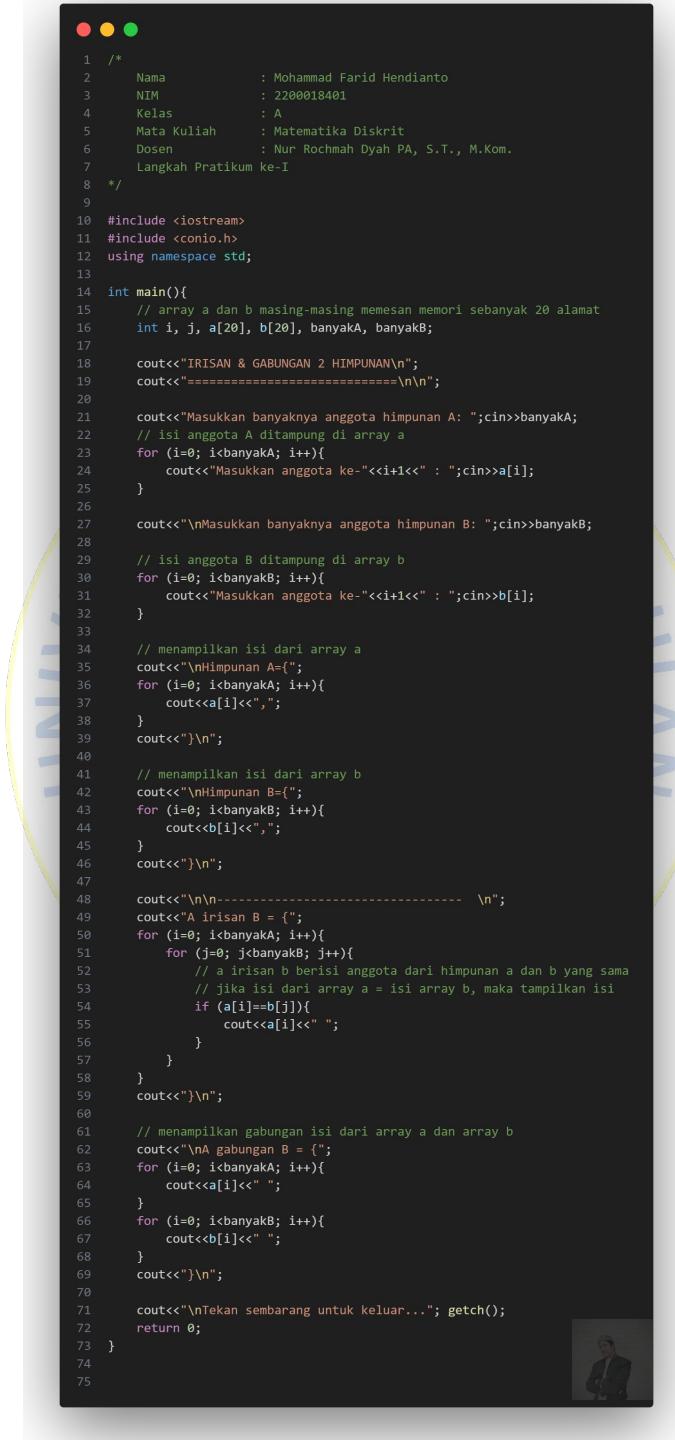
Setelah pengguna menekan tombol Enter, program akan menampilkan hasil generate permutasi dari himpunan a dengan jumlah permutasi sebanyak nilai max yang telah dihitung sebelumnya. Setiap permutasi akan ditampilkan dengan nomor urut dan elemen-elemen himpunan a yang telah diacak.

Terakhir, program akan menunggu pengguna menekan tombol Enter menggunakan fungsi getch() sebelum program berakhir.



Berikut adalah analisis perbedaan dengan program yang ada diawal materi yang ada

## Perbandingan dengan pertemuan I: Himpunan



```

1  /*
2   Nama : Mohammad Farid Hendianto
3   NIM : 2200018401
4   Kelas : A
5   Mata Kuliah : Matematika Diskrit
6   Dosen : Nur Rochmah Dyah PA, S.T., M.Kom.
7   Langkah Praktikum ke-I
8 */
9
10 #include <iostream>
11 #include <conio.h>
12 using namespace std;
13
14 int main(){
15     // array a dan b masing-masing memesan memori sebanyak 20 alamat
16     int i, j, a[20], b[20], banyakA, banyakB;
17
18     cout<<"IRISAN & GABUNGAN 2 HIMPUNAN\n";
19     cout<<"=====\\n\\n";
20
21     cout<<"Masukkan banyaknya anggota himpunan A: "; cin>>banyakA;
22     // isi anggota A ditampung di array a
23     for (i=0; i<banyakA; i++){
24         cout<<"Masukkan anggota ke-<<i+1<< : "; cin>>a[i];
25     }
26
27     cout<<"\\nMasukkan banyaknya anggota himpunan B: "; cin>>banyakB;
28
29     // isi anggota B ditampung di array b
30     for (i=0; i<banyakB; i++){
31         cout<<"Masukkan anggota ke-<<i+1<< : "; cin>>b[i];
32     }
33
34     // menampilkan isi dari array a
35     cout<<"\\nHimpunan A={";
36     for (i=0; i<banyakA; i++){
37         cout<<a[i]<<",";
38     }
39     cout<<}\\n";
40
41     // menampilkan isi dari array b
42     cout<<"\\nHimpunan B={";
43     for (i=0; i<banyakB; i++){
44         cout<<b[i]<<",";
45     }
46     cout<<}\\n ";
47
48     cout<<"\\n\\n----- \\n";
49     cout<<"A irisan B = {";
50     for (i=0; i<banyakA; i++){
51         for (j=0; j<banyakB; j++){
52             // a irisan b berisi anggota dari himpunan a dan b yang sama
53             // jika isi dari array a = isi array b, maka tampilkan isi
54             if (a[i]==b[j]){
55                 cout<<a[i]<<" ";
56             }
57         }
58     }
59     cout<<}\\n";
60
61     // menampilkan gabungan isi dari array a dan array b
62     cout<<"\\nA gabungan B = {";
63     for (i=0; i<banyakA; i++){
64         cout<<a[i]<<" ";
65     }
66     for (i=0; i<banyakB; i++){
67         cout<<b[i]<<" ";
68     }
69     cout<<}\\n";
70
71     cout<<"\\nTekan sembarang untuk keluar..."; getch();
72     return 0;
73 }
74
75

```

Gambar 14 Program Pertemuan I tentang himpunan. (Sumber: Penulis)

Program Himpunan dari langkah praktikum pertama bertujuan untuk menampilkan irisan dan gabungan dari dua buah himpunan A dan B. Pada program ini, pengguna diminta untuk memasukkan banyaknya anggota himpunan A dan B, lalu memasukkan nilai-nilai anggota tersebut. Selanjutnya nilai-nilai tersebut disimpan dalam array a dan b.

Setelah itu, program akan menampilkan isi dari array a dan array b sesuai dengan banyaknya anggota yang diinputkan. Setelah itu, program menampilkan irisan dari himpunan A dan B dengan menggunakan loop for. Pada bagian ini, setiap elemen dari array a akan dibandingkan dengan setiap elemen dari array b. Apabila ditemukan ada nilai yang sama, nilai tersebut akan ditampilkan sebagai bagian dari himpunan irisan.

Selain irisan, program juga menampilkan gabungan dari himpunan A dan B. Pada bagian ini, setiap elemen dari array a akan ditampilkan terlebih dahulu, kemudian diikuti dengan setiap elemen dari array b. Hasil akhir dari program ini adalah tampilan irisan dan gabungan dari dua buah himpunan.

Sedangkan pada program Permutasi dari langkah praktikum kesembilan bertujuan untuk men-generate permutasi dari n buah elemen. Program ini bekerja dengan memanfaatkan algoritma generate next-permutation. Pada program ini, pengguna diminta untuk memasukkan nilai n dan elemen-elemen tersebut akan disimpan dalam array a.

Program juga meminta pengguna untuk memasukkan nilai r yang merupakan banyaknya elemen yang akan digunakan untuk menghasilkan permutasi. Program kemudian akan menghitung nilai permutasi menggunakan fungsi faktorial yang dituliskan pada bagian lain dari program.

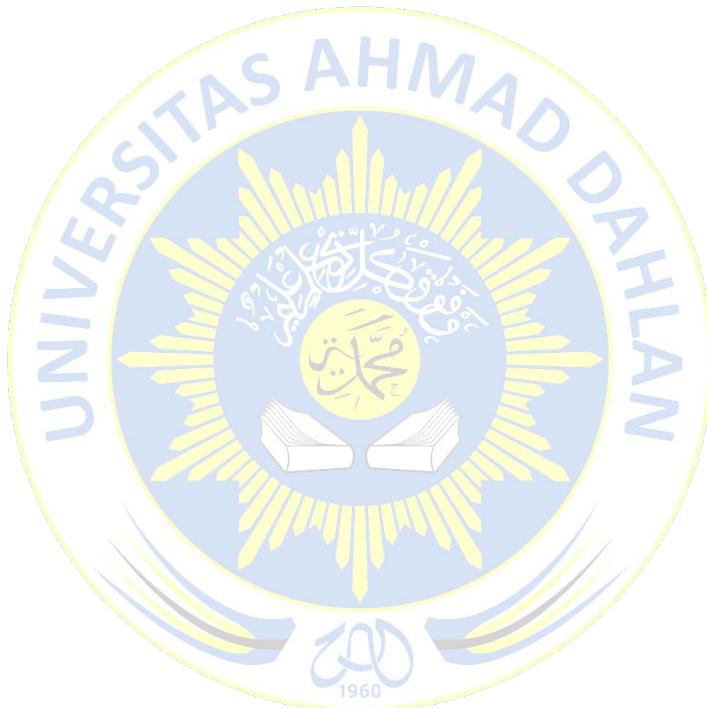
Setelah itu, program akan men-generate setiap permutasi menggunakan loop while. Pada bagian ini, program akan menampilkan nomor urutan permutasi, kemudian menampilkan nilai-nilai elemen dari array a sebanyak r kali. Setelah itu, program akan menukar posisi elemen-elemen pada array a sesuai dengan algoritma next-permutation. Selanjutnya, program akan menampilkan hasil generate permutasi yang dihasilkan.

Perbedaan mendasar antara kedua program ini adalah tujuannya. Program Himpunan bertujuan untuk menampilkan gabungan dan irisan dari dua himpunan A dan B, sedangkan Program Permutasi bertujuan untuk men-generate setiap permutasi dari n buah elemen. Selain itu, cara kerja keduanya juga berbeda. Program Himpunan bekerja dengan memanfaatkan loop for sederhana untuk menampilkan irisan dan gabungan, sedangkan Program Permutasi bekerja dengan menggunakan algoritma next-permutation yang lebih kompleks.

Meskipun begitu, kedua program ini memiliki hubungan karena keduanya menggunakan struktur data array. Pada Program Himpunan, data anggota himpunan A dan B disimpan dalam dua array a dan b, kemudian ditampilkan melalui loop for. Sedangkan pada Program Permutasi, elemen-elemen yang akan digenerate permutasinya juga disimpan dalam sebuah array a.

Selain itu, Program Permutasi juga memanfaatkan fungsi faktorial untuk menghitung nilai permutasi. Fungsi faktorial ini juga digunakan pada Program Himpunan pada saat mengalokasikan memori sebanyak 20 alamat untuk array a dan b.

Dalam hal kesamaan dan perbedaan antara kedua program ini, terdapat beberapa hal yang perlu diperhatikan. Pertama, keduanya menggunakan struktur data array sebagai media penyimpanan data. Kedua, keduanya memanfaatkan loop untuk melakukan operasi pada setiap elemen dalam array. Ketiga, tujuan dari keduanya sangat berbeda, di mana Program Himpunan bertujuan untuk menampilkan gabungan dan irisan dari dua himpunan, sedangkan Program Permutasi bertujuan untuk men-generate setiap permutasi dari n buah elemen.



## Perbandingan dengan Pertemuan II: Relasi

```

1  /*
2   Nama          : Mohammad Farid Hendianto
3   NIM           : 2200018401
4   Kelas         : A
5   Mata Kuliah   : Matematika Diskrit
6   Dosen          : Nur Rochmah Dyah PA, S.T., M.Kom.
7   Langkah Praktikum Pertemuan ke-II
8  */
9
10 #include <iostream>
11 using namespace std;
12
13 int main(){
14     string a[] = {"changmin","Jaejoong"};
15     string b[] = {"f8291","n4810","b0637"};
16
17     int c[] = {2, 3, 4};
18     int d[] = {2, 4, 8, 10, 12};
19
20     cout << "Hasil penggabungan a dan b adalah : \n";
21     for (int i = 0; i < 2;){
22         for (int j = 0; j < 3; j++){
23             cout << "(" << a[i] << "," << b[j] << ")";
24         }
25         i++;
26     }
27     cout << "}" << endl;
28
29     cout << "Hasil himpunan c habis membagi d : \n";
30     for (int k = 0; k < 3;){
31         for (int i = 0; i < 5; i++){
32             if (d[i] % c[k] == 0){
33                 cout << "(" << c[k] << "," << d[i] << "),";
34             }
35         }
36         k++;
37     }
38     cout << "}" << endl;
39     return 0;
40 }

```

Gambar 15 Kodingan langkah praktikum pertemuan kedua. (Sumber: Penulis)

Dalam analisis perbedaan antara program Langkah Praktikum Pertemuan ke-II tentang Relasi dan program Langkah Praktikum Pertemuan ke-9 tentang Permutasi, terdapat beberapa perbedaan yang dapat dianalisis secara kritis dan teliti.

Pertama, perbedaan yang paling mencolok adalah pada jenis topik yang dibahas dalam kedua program tersebut. Program Langkah Praktikum Pertemuan ke-II membahas mengenai matematika diskrit dalam konteks relasi, sedangkan program Langkah Praktikum Pertemuan

ke-9 membahas mengenai permutasi. Perbedaan ini menunjukkan bahwa kedua program tersebut memiliki fokus yang berbeda dalam pembahasan konsep matematika diskrit.

Kedua, dari segi pendekatan pemrograman yang digunakan, terdapat perbedaan yang cukup signifikan. Program Langkah Praktikum Pertemuan ke-II menggunakan bahasa pemrograman C++ dengan pendekatan prosedural, sedangkan program Langkah Praktikum Pertemuan ke-9 menggunakan bahasa pemrograman C dengan pendekatan fungsi. Perbedaan pendekatan ini berpengaruh pada cara penulisan kode program dan pengelolaan data.

Ketiga, dalam hal kompleksitas algoritma, kedua program juga memiliki perbedaan yang mencolok. Program Langkah Praktikum Pertemuan ke-II lebih sederhana dalam hal algoritma yang digunakan, karena hanya menggabungkan dua array dan melakukan operasi pemilihan elemen yang memenuhi suatu syarat tertentu. Sedangkan program Langkah Praktikum Pertemuan ke-9 menggunakan algoritma generate next-permutation yang lebih kompleks untuk menghasilkan permutasi dari suatu himpunan.

Keempat, terdapat perbedaan pada variabel dan tipe data yang digunakan dalam kedua program. Dalam program Langkah Praktikum Pertemuan ke-II, digunakan variabel bertipe string dan int sebagai representasi dari elemen dalam relasi dan array, sedangkan dalam program Langkah Praktikum Pertemuan ke-9, digunakan variabel bertipe int sebagai representasi dari elemen dalam himpunan dan permutasi.

Kelima, cara pengumpulan input dan output dari kedua program juga berbeda. Program Langkah Praktikum Pertemuan ke-II menggunakan fungsi cout untuk menampilkan hasil output pada layar, sedangkan program Langkah Praktikum Pertemuan ke-9 menggunakan fungsi printf untuk menampilkan hasil output pada layar.

Meskipun terdapat perbedaan signifikan antara kedua program tersebut, namun terdapat hubungan antara konsep Relasi dan Permutasi dalam matematika diskrit. Konsep Relasi dapat diaplikasikan dalam pembahasan permutasi dengan mengkaji hubungan antara elemen-elemen dalam himpunan dan permutasi tersebut. Sebagai contoh, jika terdapat dua himpunan A dan B, maka relasi antara elemen-elemen dalam himpunan tersebut dapat dijadikan dasar untuk menghasilkan permutasi dari kedua himpunan tersebut.

Selain itu, kajian tentang permutasi juga dapat membantu dalam mempelajari lebih lanjut tentang relasi dalam matematika diskrit, seperti relasi simetri, antisimetri, transitif, dan sebagainya. Dengan pemahaman yang baik tentang kedua konsep ini, maka dapat memberikan pemahaman yang lebih komprehensif tentang matematika diskrit dan bagaimana mengaplikasikan konsep-konsep tersebut dalam berbagai bidang ilmu.

Dalam kaitannya dengan pengembangan program, perbedaan antara kedua program tersebut dapat dijadikan referensi untuk memilih pendekatan dan algoritma yang tepat dalam menyelesaikan suatu masalah atau tugas yang berkaitan dengan matematika diskrit. Dengan memahami perbedaan dan karakteristik masing-masing topik, maka kita dapat memilih pendekatan yang lebih efektif dan efisien dalam menyelesaikan suatu masalah atau tugas.

### Perbandingan dengan Pertemuan III: Relasi N-Array

Relasi N-Array adalah topik dalam mata kuliah Matematika Diskrit yang berkaitan dengan struktur data relasi dalam bentuk matriks atau array. Sedangkan program permutasi merupakan salah satu contoh penerapan konsep kombinatorial dalam pemrograman, yaitu untuk menghasilkan semua kemungkinan susunan atau urutan dari sekumpulan objek.

Perbedaan utama antara kedua program ini terletak pada konsep matematika yang mendasarinya. Relasi N-Array lebih berkaitan dengan analisis struktur data yang kompleks, sedangkan program permutasi lebih fokus pada konsep kombinatorial dan algoritma generate next-permutation. Namun, kedua program ini memiliki hubungan erat dalam penggunaannya dalam konteks ilmu komputer dan pemrograman.

Dalam praktikum Relasi N-Array, kita mempelajari cara membuat dan mengelola relasi dalam bentuk matriks atau array menggunakan bahasa SQL. Prosesnya melibatkan pemodelan struktur data dan hubungan antar data dengan cara yang efisien dan mudah dipahami. Di sisi lain, program permutasi menggunakan metode generate next-permutation untuk menghasilkan semua susunan atau urutan dari sebuah himpunan objek, dengan tujuan untuk mencari solusi terbaik dalam masalah optimasi atau pencarian.

Namun, keduanya tidak berdiri sendiri dan saling berkaitan dalam beberapa aspek. Misalnya, dalam program permutasi, kita dapat menggunakan teknik seperti brute-force search dan backtracking untuk menemukan solusi optimal. Hal ini membutuhkan pemahaman yang baik tentang struktur data dan relasi antar data dalam suatu masalah.

Di sisi lain, dalam praktikum Relasi N-Array, kita dapat menggunakan teknik seperti join dan subquery untuk menggabungkan beberapa tabel atau matriks menjadi satu kesatuan yang lebih besar. Teknik ini juga dapat membantu dalam pemecahan masalah yang kompleks dalam pengolahan data dan analisis.

Dalam hal ini, kedua program tersebut memiliki hubungan erat dan saling melengkapi dalam penggunaannya sebagai alat bantu dalam pemrograman dan pengolahan data. Dalam pemrograman, penting bagi para pengembang untuk memahami konsep matematika dasar seperti kombinatorial dan struktur data relasi agar dapat membuat program yang efektif dan efisien.

## Perbandingan dengan Pertemuan IV: Fungsi

```

1  /*
2   Nama      : Mohammad Farid Hendianto
3   NIM       : 2200018401
4   Kelas     : A
5   Mata Kuliah : Matematika Diskrit
6   Dosen     : Nur Rochmah Dyah PA, S.T., M.Kom.
7   Langkah Praktikum ke-IV
8 */
9
10 #include <iostream>
11 using namespace std;
12
13 class hitung{
14 public:
15     int proses();
16     void input();
17
18 private:
19     int n;
20     float rumus,jumlah,total;
21 };
22
23 void hitung::input(){
24     cin>>n;
25     cout<<endl;
26 }
27
28 int hitung::proses(){
29     jumlah=0;total=0;
30     rumus=-1;
31     for(int j=1;j<=n;j++){
32         rumus=rumus*(-1);total=rumus/j;
33         jumlah+=total;
34         if (j==1)
35             cout<<"("<<total<<")";
36         if (j>1)
37             cout<<"+("<<total<<")";
38     }
39     cout<<endl<<endl<<"hasil penjumlahan deret = "<<jumlah;
40     return jumlah;
41 }
42
43 int main(){
44     cout<<"program sederhana menghitung jumlah dari rumus 1-(1/2)+(1/3)-(1/4)+...+(1/n)"<<endl<<endl;
45     cout<<"tentukan nilai n : ";hitung deret;deret.input();
46     deret.proses();return 0;
47 }
48

```

Gambar 16 Kodingan pada pertemuan keempat. (Sumber: Penulis)

Kedua program tersebut memiliki perbedaan pada fungsi dan tujuannya. Program pada langkah praktikum pertemuan keempat berfungsi untuk menghitung jumlah dari rumus  $1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \dots + \frac{1}{n}$  sedangkan program pada langkah praktikum pertemuan kesembilan berfungsi untuk membuat daftar permutasi dari suatu himpunan.

Pada program langkah praktikum keempat, terdapat sebuah kelas bernama `hitung` dengan dua belas fungsi, yaitu `input()` dan `proses()`. Fungsi `input()` digunakan untuk meminta inputan variabel `n`, sedangkan fungsi `proses()` digunakan untuk melakukan penghitungan jumlah dari rumus yang diinginkan. Selain itu, pada program ini juga terdapat sebuah loop `for` untuk melakukan perulangan sebanyak `n` kali. Setiap perulangan, nilai rumus akan diubah

menjadi negatif (dikalikan -1) kemudian dibagi dengan variabel j. Selanjutnya, nilai total akan ditambahkan dengan hasil perhitungan rumus, dan nilai jumlah akan diakumulasikan oleh nilai total. Lalu, jika nilai j sama dengan 1 atau lebih besar dari 1, maka akan dicetak output "(nilai total)" atau "+(nilai total)". Terakhir, hasil penjumlahan deret akan dikembalikan oleh fungsi proses().

Sementara itu, program pada langkah praktikum kesembilan menggunakan algoritma generate next-permutation untuk menghasilkan daftar permutasi dari suatu himpunan. Pertama-tama, program akan meminta inputan nilai n, yaitu jumlah data yang ingin dimasukkan ke dalam himpunan. Kemudian, program akan meminta inputan nilai untuk setiap elemen pada himpunan a. Setelah itu, program akan meminta inputan nilai r yang merupakan ukuran dari permutasi yang ingin dibuat. Selanjutnya, program akan menghitung nilai permutasi dengan menggunakan fungsi faktorial(). Kemudian, program akan menampilkan output nilai permutasi dan meminta user untuk menekan tombol enter untuk melihat hasil generate permutasi. Program ini juga menggunakan loop for untuk melakukan generate next-permutation sebanyak nilai permutasi. Pada setiap iterasi, program akan mencetak nomor dan daftar permutasi yang dihasilkan oleh algoritma generate next-permutation.

Hubungan antara kedua program tersebut adalah keduanya memiliki fungsi yang berbeda-beda namun keduanya sama-sama menggunakan beberapa metode matematika diskrit. Pada program langkah praktikum keempat, terdapat sebuah rumus yang digunakan untuk menghitung jumlah suatu deret, sedangkan pada program langkah praktikum kesembilan, digunakan algoritma generate next-permutation untuk membuat daftar permutasi dari suatu himpunan. Kedua program tersebut juga menggunakan perulangan untuk mengulang tugas yang sama secara berkali-kali untuk mendapatkan hasil yang diinginkan. Selain itu, keduanya juga menggunakan fungsi faktorial() untuk menghitung nilai faktorial suatu bilangan, dimana faktorial() adalah salah satu topik yang dipelajari dalam matematika diskrit.

Kesimpulannya, kedua program tersebut memiliki tujuan yang berbeda-beda namun sama-sama menggunakan beberapa metode matematika diskrit. Program pada langkah praktikum keempat berfungsi untuk menghitung jumlah suatu deret, sedangkan program pada langkah praktikum kesembilan berfungsi untuk membuat daftar permutasi dari suatu himpunan. Keduanya juga menggunakan loop for untuk melakukan perulangan dan fungsi faktorial() untuk menghitung nilai faktorial suatu bilangan. Namun, kedua program tersebut memiliki cara implementasi yang berbeda dalam pemrogramannya.

## Perbandingan dengan Pertemuan V: Fungsi dengan Komposisi Dua Fungsi

```

File Edit Selection View Go Run Terminal Help
absoluteNumber.cpp - Visual Studio Code - Insiders
absoluteNumber.cpp factorialNumber.cpp compositeFunction.cpp

D:\> Document Ndik > Kuliah > Semester 2 > Matematika Diskrit > Praktikum > Pertemuan 5 > absoluteNumber.cpp > Absolut(double)

1 #include <iostream>
2 using namespace std;
3
4 double Absolut(double X);
5
6 int main(){
7     float Nilai;
8     Nilai=-123.45;
9     cout<<Nilai<<" nilai mutlaknya adalah "<<Absolut(Nilai)<<endl;
10 }
11
12 /* ... Fungsi untuk memutalkan nilai negatif ... */
13 double Absolut(double X){
14     if(X<0)
15         X=-X;
16     return X;
17 }
18 }

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5> cd "d:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5\" ; if ($?) { g++ absoluteNumber.cpp -o a
absoluteNumber } ; if ($?) { ./absoluteNumber
-123.45 nilai mutlaknya adalah 123.45
PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5> [REDACTED]

```

Gambar 17 Program pertama pada praktikum pertemuan kelima. (Sumber: Penulis)

```

File Edit Selection View Go Run Terminal Help
factorialNumber.cpp - Visual Studio Code - Insiders
factorialNumber.cpp absoluteNumber.cpp compositeFunction.cpp

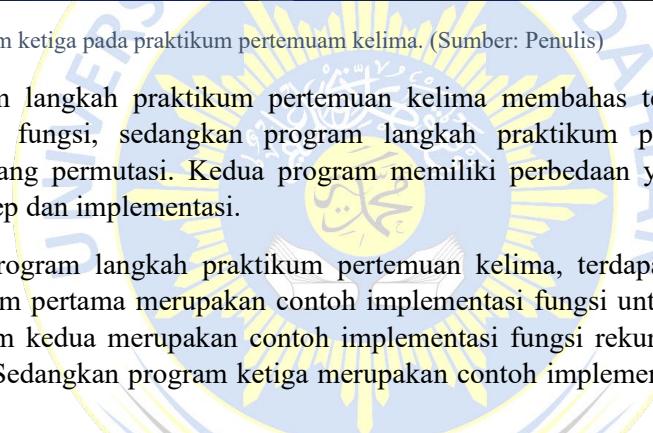
D:\> Document Ndik > Kuliah > Semester 2 > Matematika Diskrit > Praktikum > Pertemuan 5 > factorialNumber.cpp > Fak_Rekursif(int)

1 #include <stdio.h>
2 long int Fak_Rekursif (int N);
3
4 main(){
5     int N=N-5;
6     printf("%d faktorial = %ld\n",N,Fak_Rekursif(N));
7 }
8
9 long int Fak_Rekursif (int N){
10     if (N<=1)
11         return 1;
12     else
13         return N*Fak_Rekursif(N-1);
14 }
15

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5> cd "d:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5\" ; if ($?) { g++ factorialNumber.cpp -o
factorialNumber } ; if ($?) { ./factorialNumber
5 faktorial = 120
PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5> [REDACTED]

```

Gambar 18 Program kedua pada praktikum pertemuan kelima. (Sumber: Penulis)



```

File Edit Selection View Go Run Terminal Help
absoluteNumber.cpp factorialNumber.cpp compositeFunction.cpp factorialNumber.cpp - Visual Studio Code - Insiders
D:\Document Ndik\Kuliah> Semester 2> Matematika Diskrit > Praktikum > Pertemuan 5 > factorialNumber.cpp > Fak_Rekursif(int)
1 #include <stdio.h>
2 long int Fak_Rekursif (int N);
3
4 main(){
5     int N;N=5;
6     printf("%d faktorial = %ld\n",N,Fak_Rekursif(N));
7 }
8
9 long int Fak_Rekursif (int N){[cursor]
10    if (N<=1)
11        return 1;
12    else
13        return N*Fak_Rekursif(N-1);
14 }
15

```

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE

PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5> cd "D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5\"
PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5> g++ factorialNumber.cpp -o FactorialNumber ; if (\$?) { ./FactorialNumber
\$ Factorial = 120
PS D:\Document Ndik\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 5>

Ln 15, Col 2 Spaces: 4 UTF-8 CRLF C++ Go Live Win32

Gambar 19 Program ketiga pada praktikum pertemuan kelima. (Sumber: Penulis)

Program langkah praktikum pertemuan kelima membahas tentang fungsi dengan komposisi dua fungsi, sedangkan program langkah praktikum pertemuan kesembilan membahas tentang permutasi. Kedua program memiliki perbedaan yang cukup signifikan dalam hal konsep dan implementasi.

Pada program langkah praktikum pertemuan kelima, terdapat tiga program yang dibahas. Program pertama merupakan contoh implementasi fungsi untuk memutlakkan nilai negatif. Program kedua merupakan contoh implementasi fungsi rekursif untuk menghitung nilai faktorial. Sedangkan program ketiga merupakan contoh implementasi fungsi komposisi dua fungsi.

Sedangkan pada program langkah praktikum pertemuan kesembilan, terdapat satu program yang membahas tentang generate permutasi. Program ini digunakan untuk menghasilkan permutasi dari data yang diinputkan oleh pengguna.

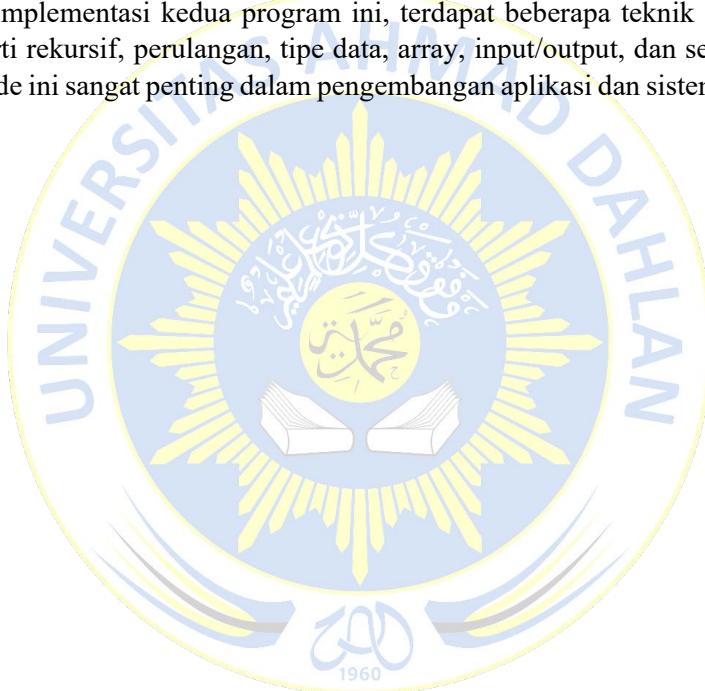
Perbedaan mendasar antara kedua program ini adalah pada konsep yang digunakan. Pada program langkah praktikum pertemuan kelima, konsep yang digunakan adalah fungsi dan komposisi fungsi. Sedangkan pada program langkah praktikum pertemuan kesembilan, konsep yang digunakan adalah permutasi.

Selain itu, implementasi kedua program juga berbeda. Pada program langkah praktikum pertemuan kelima, terdapat beberapa contoh program yang diimplementasikan menggunakan bahasa pemrograman C++ dan C. Sedangkan pada program langkah praktikum pertemuan kesembilan, hanya terdapat satu program yang diimplementasikan menggunakan bahasa pemrograman C.

Selain perbedaan mendasar tersebut, terdapat juga perbedaan lainnya antara kedua program ini. Pada program langkah praktikum pertemuan kelima, lebih banyak membahas tentang konsep matematika diskrit yang berkaitan dengan fungsi dan komposisi fungsi. Sedangkan pada program langkah praktikum pertemuan kesembilan, lebih banyak membahas tentang implementasi algoritma generate permutasi dan penggunaannya dalam pemrograman.

Meskipun demikian, kedua program ini memiliki beberapa hubungan yang saling terkait. Konsep fungsi dan komposisi fungsi pada program langkah praktikum pertemuan kelima dapat digunakan untuk mempermudah implementasi algoritma generate permutasi pada program langkah praktikum pertemuan kesembilan. Selain itu, konsep permutasi juga dapat diterapkan dalam penghitungan kombinasi yang merupakan salah satu bagian dari konsep matematika diskrit yang dibahas pada program langkah praktikum pertemuan kelima.

Dalam implementasi kedua program ini, terdapat beberapa teknik dan metode yang digunakan seperti rekursif, perulangan, tipe data, array, input/output, dan sebagainya. Semua teknik dan metode ini sangat penting dalam pengembangan aplikasi dan sistem yang kompleks.



## Perbandingan dengan Pertemuan VI: Bilangan Bulat

```

1 #include <iostream>
2
3 using namespace std;
4
5 class Math {
6 public:
7     // Algoritma Brute Force
8     int getPK(int num1, int num2) {
9         if (1 % num1 == 0 && 1 % num2 == 0) {
10            return 1;
11        }
12        return getPK(num1, num2);
13    }
14
15    // int getFPR(int num1, int num2) {
16    //     if (num1 == 0) {
17    //         return num1;
18    //     } else {
19    //         return getFPR(num2, num1 % num2);
20    //     }
21    //}
22    // Algoritma Euclidean
23    int getPK(int num1, int num2) {
24        int fpr = getFPR(num1, num2);
25        return (num1 * num2) / fpr;
26    }
27
28    int getFPR(int num1, int num2) {
29        while (num2 != 0) {
30            int temp = num2;
31            num2 = num1 % num2;
32            num1 = temp;
33        }
34        return num1;
35    }
36};
37
38 class Menu {
39 public:
40     int displayMenu() {
41         int choice;
42         cout << "Masukkan pilihan anda:" << endl
43             << "1. Menentukan KPK" << endl
44             << "2. Menentukan FPR" << endl
45             << "3. Keluar" << endl
46             << ">";
47         cin >> choice;
48         return choice;
49     }
50
51     void inputNum(int& num1, int& num2) {
52         cout << "Masukkan bilangan pertama: ";
53         cin >> num1;
54         cout << "Masukkan bilangan kedua: ";
55         cin >> num2;
56     }
57     void pressEnterToContinue() {
58         cout << "Tekan Enter untuk melanjutkan...";
59         cin.ignore();
60         cin.get();
61     }
62
63 class Program {
64 public:
65     void run() {
66         Math math;
67         Menu menu;
68         int choice;
69         int num1, num2;
70
71         while (true) {
72             system("cls");
73             choice = menu.displayMenu();
74
75             system("cls");
76             switch (choice) {
77                 case 1:
78                     cout << "Menentukan KPK" << endl;
79                     menu.inputNum(num1, num2);
80
81                     cout << "Bilangan pertama: " << num1 << endl;
82                     cout << "Bilangan kedua : " << num2 << endl;
83                     cout << "KPK: " << math.getPK(num1, num2) << endl;
84                     break;
85
86                 case 2:
87                     cout << "Menentukan FPR" << endl;
88                     menu.inputNum(num1, num2);
89
90                     cout << "Bilangan pertama: " << num1 << endl;
91                     cout << "Bilangan kedua : " << num2 << endl;
92                     cout << "FPR: " << math.getFPR(num1, num2) << endl;
93                     break;
94
95                 case 3:
96                     cout << "Keluar dari program..." << endl;
97                     return;
98
99                 default:
100                     cout << "Pilihan tidak valid!" << endl;
101                 }
102
103             menu.pressEnterToContinue();
104         }
105     }
106 };
107
108 int main() {
109     Program program;
110     program.run();
111     return 0;
112 }

```

Gambar 20 Program pertemuan keenam. (Sumber: Penulis)

Dalam menganalisis perbedaan dari program Langkah Praktikum Pertemuan 6 dan 9, terdapat beberapa hal yang dapat diperhatikan. Dalam tugas ini, kita akan membahas secara mendetail tentang perbedaan antara kedua program tersebut.

Pada program Langkah Praktikum Pertemuan 6, program tersebut digunakan untuk menentukan KPK dan FPB dari dua bilangan bulat. Program tersebut menggunakan dua algoritma, yaitu Algoritma Brute Force dan Algoritma Euclidean. Pada Algoritma Brute Force, program akan mencoba setiap angka yang lebih besar dari kedua bilangan tersebut sampai mendapatkan kelipatan persekutuan keduanya atau KPK-nya. Sedangkan pada Algoritma Euclidean, program akan menghitung FPB (Faktor Persekutuan Terbesar) dengan cara membagi bilangan yang lebih besar dengan yang lebih kecil dan menyimpan sisa hasil bagi pada variabel. Kemudian, bilangan yang lebih kecil akan menjadi pembilang baru dan sisa hasil bagi tersebut akan menjadi penyebut baru. Proses ini diulangi terus menerus sampai sisa hasil bagi tersebut sama dengan nol. Setelah itu, bilangan yang lebih kecil adalah FPB dari kedua bilangan tersebut.

Sedangkan pada Program Langkah Praktikum Pertemuan 9, program tersebut digunakan untuk menghasilkan permutasi dari himpunan data dengan nilai r tertentu. Dalam program ini, fungsi faktorial digunakan untuk menghitung nilai faktorial dari n yang kemudian digunakan untuk menghitung jumlah permutasi dari data dengan nilai r tertentu. Program ini juga menggunakan algoritma generate next-permutation untuk menghasilkan permutasi sebanyak nilai permutasi yang telah dihitung sebelumnya.

Dari penjelasan singkat di atas, perbedaan antara kedua program tersebut dapat dilihat dari tujuan dan metode yang digunakan. Pada Program Langkah Praktikum Pertemuan 6, tujuannya adalah untuk menentukan KPK dan FPB dari dua bilangan bulat dengan menggunakan Algoritma Brute Force atau Algoritma Euclidean. Sedangkan pada Program Langkah Praktikum Pertemuan 9, tujuannya adalah untuk menghasilkan permutasi dari himpunan data dengan nilai r tertentu dengan menggunakan algoritma generate next-permutation.

Selain itu, terdapat juga perbedaan dalam hal bahasa pemrograman yang digunakan. Program Langkah Praktikum Pertemuan 6 dibuat dengan menggunakan bahasa pemrograman C++, sementara Program Langkah Praktikum Pertemuan 9 dibuat dengan menggunakan bahasa pemrograman C.

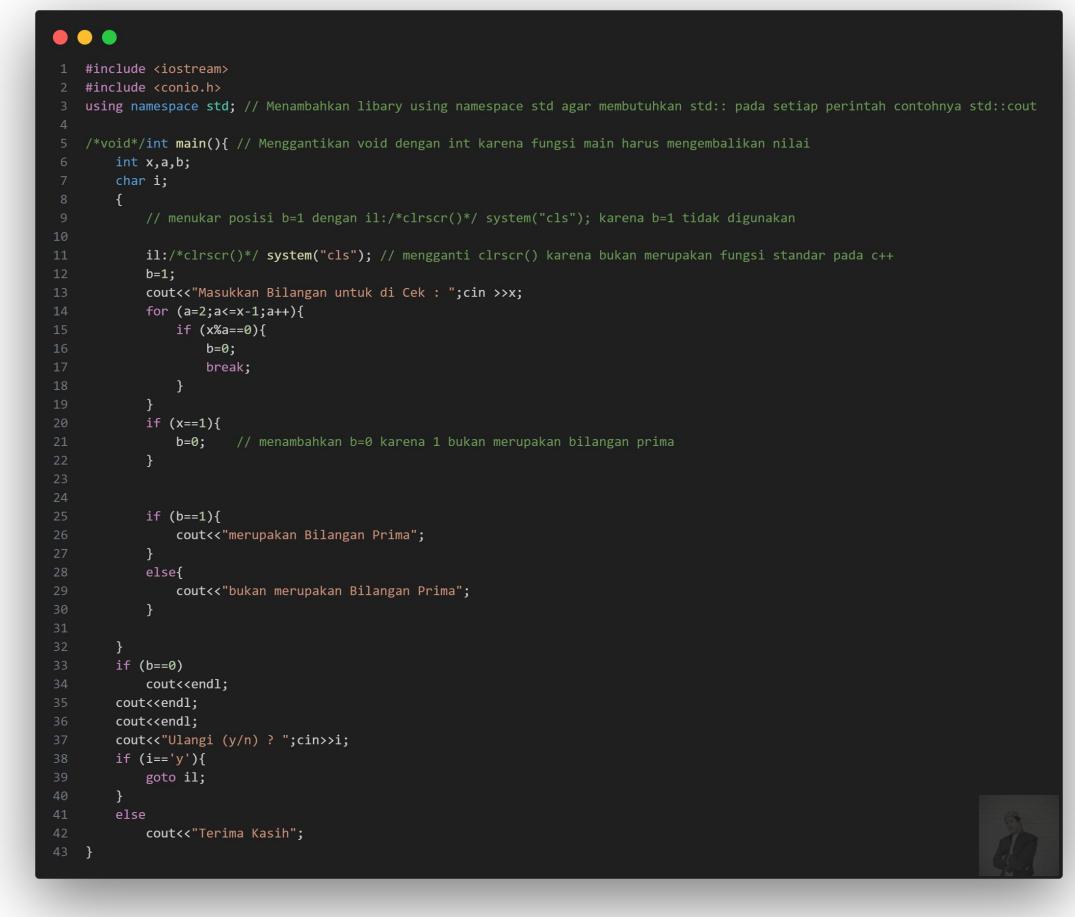
Dalam hal hubungan antara kedua program tersebut, keduanya tidak memiliki hubungan yang langsung satu sama lain karena memiliki tujuan dan metode yang berbeda. Namun, keduanya memiliki kesamaan dalam penggunaan fungsi faktorial, sehingga dapat dikatakan bahwa Program Langkah Praktikum Pertemuan 9 meneruskan penggunaan fungsi faktorial yang dimulai pada Program Langkah Praktikum Pertemuan 6.

Dalam konteks matematika diskrit, keduanya juga memiliki perbedaan yang signifikan. Program Langkah Praktikum Pertemuan 6 lebih berkaitan dengan topik aritmatika bilangan bulat, khususnya dalam menentukan KPK dan FPB dari dua bilangan bulat.

Sedangkan Program Langkah Praktikum Pertemuan 9 lebih berkaitan dengan topik kombinatorika, khususnya dalam menghasilkan permutasi dari himpunan data.

Jadi, Program Langkah Praktikum Pertemuan 6 dan 9 memiliki tujuan dan metode yang berbeda serta dibuat dengan menggunakan bahasa pemrograman yang berbeda pula. Namun, keduanya memiliki kesamaan dalam penggunaan fungsi faktorial. Dalam konteks matematika diskrit, Program Langkah Praktikum Pertemuan 6 lebih berkaitan dengan aritmatika bilangan bulat, sedangkan Program Langkah Praktikum Pertemuan 9 lebih berkaitan dengan kombinatorika.

### Perbandingan dengan Pertemuan VII: Aplikasi Bilangan Bulat



```

1 #include <iostream>
2 #include <conio.h>
3 using namespace std; // Menambahkan library using namespace std agar membutuhkan std:: pada setiap perintah contohnya std::cout
4
5 /*void*/int main(){ // Menggantikan void dengan int karena fungsi main harus mengembalikan nilai
6     int x,a,b;
7     char i;
8     {
9         // menukar posisi b=1 dengan il://clrscr()/* system("cls"); karena b=1 tidak digunakan
10        il://clrscr()/* system("cls"); // mengganti clrscr() karena bukan merupakan fungsi standar pada c++
11        b=1;
12        cout<<"Masukkan Bilangan untuk di Cek : ";cin >>x;
13        for (a=2;a<x-1;a++){
14            if (x%a==0){
15                b=0;
16                break;
17            }
18        }
19        if (x==1){
20            b=0; // menambahkan b=0 karena 1 bukan merupakan bilangan prima
21        }
22
23
24        if (b==1){
25            cout<<"merupakan Bilangan Prima";
26        }
27        else{
28            cout<<"bukan merupakan Bilangan Prima";
29        }
30    }
31
32    if (b==0)
33        cout<<endl;
34    cout<<endl;
35    cout<<endl;
36    cout<<"Ulangi (y/n) ? ";cin>>i;
37    if (i=='y'){
38        goto il;
39    }
40    else
41        cout<<"Terima Kasih";
42    }
43 }

```

Gambar 21 Program pertama pada Pertemuan ketujuh. (Sumber: Penulis)

```
● ○ ●
1 #include <cstdlib>
2 #include <iostream>
3 #include <string.h>
4 #define maks 500
5
6 using namespace std;
7
8 class Enkripsi {
9 public:
10     Enkripsi();
11     void enkripsi();
12     void deskripsi();
13     void output();
14
15 private:
16     char chiper[maks];
17     int key;
18     char plain[maks];
19 };
20
21 Enkripsi::Enkripsi() {
22     cout << "Masukkan kata: ";
23     cin.getline(chiper, sizeof(chiper));
24     cout << "Masukkan key: ";
25     cin >> key;
26     cout << endl;
27 }
28
29 void Enkripsi::enkripsi() {
30     for (int i = 0; i < strlen(chiper); i++) {
31         cout << chiper[i] << "(" << int(chiper[i]) << ")";
32         chiper[i] = (chiper[i] + key) % 128;
33     }
34 }
35
36 void Enkripsi::deskripsi() {
37     for (int i = 0; i < strlen(chiper); i++) {
38         plain[i] = (chiper[i] - key) % 128;
39         chiper[i] = plain[i];
40     }
41 }
42
43 void Enkripsi::output() {
44     for (int i = 0; i < strlen(chiper); i++) {
45         cout << chiper[i];
46     }
47 }
48
49 int main(int argc, char *argv[]) {
50     Enkripsi Deskripsi;
51     Deskripsi.enkripsi();
52     cout << "\n\nSetelah diEnkripsi: ";
53     Deskripsi.output();
54     Deskripsi.deskripsi();
55     cout << "\n\nKembali diDeskripsi: ";
56     Deskripsi.output();
57     cout << endl << endl;
58     system("PAUSE");
59     return EXIT_SUCCESS;
60 }
```

Gambar 22 Program kedua pada Pertemuan ketujuh. (Sumber: Penulis)

### Perbandingan dengan Pertemuan VIII: Kombinatorik

Program pada langkah praktikum pertemuan ke-7 bertujuan untuk mengecek apakah suatu bilangan merupakan bilangan prima atau bukan. Program ini menggunakan perulangan for untuk mengecek apakah ada bilangan selain 1 dan bilangan itu sendiri yang dapat membagi habis bilangan tersebut. Jika tidak ada, maka bilangan tersebut merupakan bilangan prima. Program ini juga dilengkapi dengan fitur pengulangan agar pengguna dapat mengulangi proses pengecekan bilangan.

Sementara itu, program pada langkah praktikum pertemuan ke-9 bertujuan untuk membuat program generate permutasi. Program ini dimulai dengan meminta input jumlah data dan nilai r, kemudian meminta input data pada array. Selanjutnya, program akan menghitung nilai permutasi dengan menggunakan fungsi faktorial dan men-generate permutasi menggunakan algoritma generate next-permutation. Program ini juga dilengkapi dengan fitur untuk menampilkan hasil generate permutasi.

Perbedaan mendasar antara kedua program tersebut terletak pada tujuan program dan cara kerjanya. Program pada langkah praktikum pertemuan ke-7 berfokus pada pengecekan bilangan prima dengan menggunakan perulangan for, sementara program pada langkah praktikum pertemuan ke-9 berfokus pada pembuatan program generate permutasi dengan menggunakan algoritma generate next-permutation dan fungsi faktorial. Selain itu, program pada langkah praktikum pertemuan ke-7 dilengkapi dengan fitur pengulangan, sedangkan program pada langkah praktikum pertemuan ke-9 dilengkapi dengan fitur untuk menampilkan hasil generate permutasi.

Meski berbeda dalam tujuan dan cara kerjanya, kedua program tersebut memiliki hubungan dengan mata kuliah Matematika Diskrit. Program pada langkah praktikum pertemuan ke-7 menggunakan konsep bilangan prima yang merupakan salah satu topik dalam matematika diskrit. Sedangkan program pada langkah praktikum pertemuan ke-9 menggunakan konsep permutasi yang juga merupakan salah satu topik dalam matematika diskrit.

Selain itu, terdapat perbedaan lainnya antara kedua program tersebut. Program pada langkah praktikum pertemuan ke-7 ditulis menggunakan bahasa C++, sedangkan program pada langkah praktikum pertemuan ke-9 ditulis menggunakan bahasa C. Selain itu, program pada langkah praktikum pertemuan ke-7 menggunakan library `<iostream>` dan `<conio.h>`, sementara program pada langkah praktikum pertemuan ke-9 menggunakan library `<stdio.h>`, `<stdlib.h>`, dan `<string.h>`. Perbedaan library yang digunakan dapat mempengaruhi cara penulisan kode program dan fitur-fitur yang tersedia dalam program tersebut.

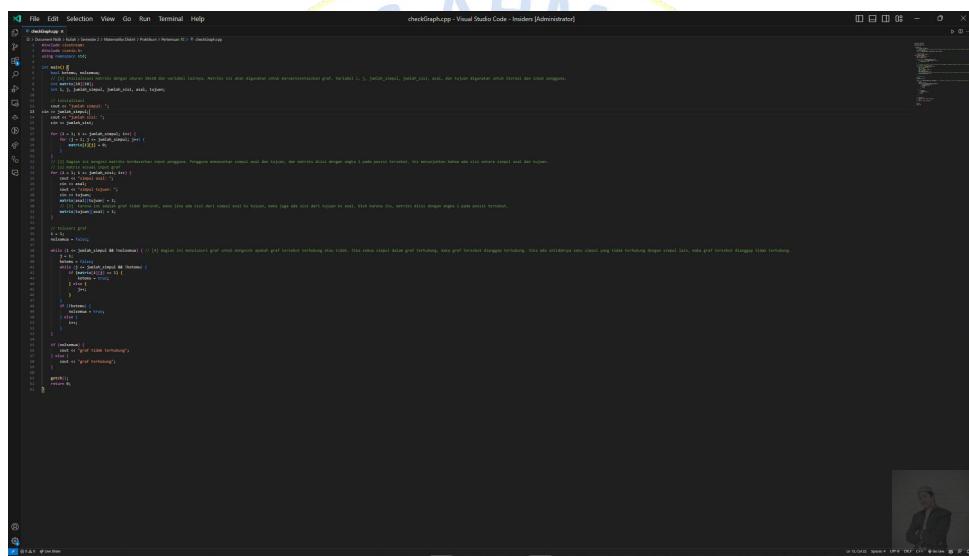
Dalam hal struktur kode program, terdapat perbedaan dalam penamaan variabel dan fungsi antara kedua program tersebut. Program pada langkah praktikum pertemuan ke-7 menggunakan nama variabel yang lebih umum seperti x, a, b, dan i, sementara program pada langkah praktikum pertemuan ke-9 menggunakan nama variabel yang lebih spesifik seperti

chiper, key, plain, p, nq, max, x, i, j, n, k, r, s, tr, dan no. Hal ini bertujuan untuk memudahkan pembacaan dan pemahaman kode program.

Selain itu, terdapat perbedaan dalam cara penulisan komentar pada kedua program tersebut. Program pada langkah praktikum pertemuan ke-7 menggunakan komentar yang ditandai dengan // atau /\* /, sedangkan program pada langkah praktikum pertemuan ke-9 menggunakan komentar yang ditandai dengan // dan /\*/. Hal ini dapat mempengaruhi cara pembacaan dan pemahaman kode program oleh pengguna.

## Perbandingan dengan Pertemuan X: GRAF (Pertemuan sekarang)

Setelah terlanjur saya membandingkan semua pertemuan (I hingga X), ternyata asisten praktikum meminta menganalisis program halaman 65 dengan 73.



```

checkGrafo.cpp - Visual Studio Code - Irudin [Administrator]

File Edit Selection View Go Run Terminal Help
D:\Documents\HDI\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Kesepuluh\10\checkGrafo.cpp

// checkGrafo.h
#include <iostream>
#include <vector>
#include <algorithm>
#include <assert.h>
using namespace std;

int main() {
    int n, m;
    cout << "Masukkan jumlah vertex: ";
    cin >> n;
    cout << "Masukkan jumlah sisi: ";
    cin >> m;
    vector<vector<int>> adj(n);
    for (int i = 0; i < m; i++) {
        int u, v;
        cout << "Masukkan sisi " << i + 1 << ": ";
        cin >> u >> v;
        adj[u].push_back(v);
        adj[v].push_back(u);
    }
    cout << endl;
    cout << "Hasil: " << endl;
    if (isBipartite(adj)) {
        cout << "Bipartite" << endl;
    } else {
        cout << "Tidak Bipartite" << endl;
    }
}

bool isBipartite(vector<vector<int>> adj) {
    int n = adj.size();
    vector<int> color(n, -1);
    queue<int> q;
    q.push(0);
    color[0] = 0;
    while (!q.empty()) {
        int u = q.front();
        q.pop();
        for (int v : adj[u]) {
            if (color[v] == -1) {
                color[v] = 1 - color[u];
                q.push(v);
            } else if (color[v] == color[u]) {
                return false;
            }
        }
    }
    return true;
}

```

Gambar 23 Membandingkan dengan program pertemuan kesepuluh. (Sumber: Penulis)

Program Langkah Praktikum Pertemuan Kesepuluh (Graf) dan Program Langkah Praktikum Pertemuan Kesembilan (Permutasi) memiliki perbedaan dan persamaan yang cukup signifikan. Berikut adalah analisis perbandingan antara kedua program tersebut:

### Tujuan Program:

- Program Graf bertujuan untuk membuat dan mengecek apakah graf yang dibuat oleh pengguna terhubung atau tidak.
- Program Permutasi bertujuan untuk menghasilkan permutasi dari himpunan angka yang dimasukkan oleh pengguna.

### Struktur Data:

```

6  boolean ketemu, nosemua;
7  // [1] Inisialisasi matriks dengan ukuran 10x10 dan variabel lainnya. Matriks ini akan digunakan untuk merepresentasikan graf. Variabel i, j, jumlah_simpul, jumlah_sisi, asal, dan tujuan digunakan untuk iterasi dan input pengguna.
8  int matrix[10][10];
9  int i, j, jumlah_simpul, jumlah_sisi, asal, tujuan;

```

Gambar 25 Program GRAF menggunakan matriks 2D

```

22  int p, nq, max, x, i, j, n, k, r, s, tr, no;
23  int a[100];
24
25  div_t xx;

```

Gambar 24 Program Permutasi menggunakan array 1D. (Sumber: Penulis)

- Program Graf menggunakan matriks 2D untuk merepresentasikan graf.
- Program Permutasi menggunakan array 1D untuk menyimpan himpunan angka.

### Fungsi dan Proses:

```

7  // Fungsi faktorial :
8  int faktorial(int nilai)
9  {
10    hasil = nilai;
11    // [1] Menggunakan perulangan untuk menghitung nilai faktorial
12    while (nilai > 1)
13    {
14      hasil = hasil * (nilai - 1);
15      nilai = nilai - 1;
16    }
17    return hasil;
18 }
19

```

Gambar 26 Fungsi tambahan pada program permutasi. (Sumber: Penulis)

- Program Graf menggunakan fungsi main saja dengan proses iterasi untuk mengisi matriks dan mengecek keterhubungan graf.
- Program Permutasi menggunakan fungsi tambahan faktorial untuk menghitung nilai faktorial, dan proses iterasi dalam fungsi main untuk menghasilkan permutasi.

### Input dan Output:

- Program Graf meminta pengguna memasukkan jumlah simpul dan sisi, serta simpul asal dan tujuan untuk setiap sisi. Outputnya adalah informasi apakah graf terhubung atau tidak.

- Program Permutasi meminta pengguna memasukkan jumlah data dan data itu sendiri, serta nilai r untuk permutasi. Outputnya adalah semua permutasi yang mungkin dari data tersebut.

Berikut adalah berbentuk tabelnya secara singkatnya

Aspek	Program Graf	Program Permutasi
Tujuan Program	Membuat dan mengecek keterhubungan graf	Menghasilkan permutasi dari himpunan angka
Struktur Data	Matriks 2D	Array 1D
Fungsi dan Proses	Fungsi main dengan proses iterasi	Fungsi main dan fungsi faktorial dengan proses iterasi
Input dan Output	Jumlah simpul dan sisi, simpul asal dan tujuan untuk setiap sisi. Output: informasi keterhubungan graf	Jumlah data, data itu sendiri, nilai r untuk permutasi. Output: semua permutasi yang mungkin

Hubungan antara kedua program ini terletak pada penggunaan struktur data dan proses iterasi. Keduanya menggunakan struktur data (matriks dan array) untuk menyimpan data, dan menggunakan proses iterasi untuk melakukan operasi pada data tersebut. Namun, tujuan dan cara mereka melakukan operasi tersebut sangat berbeda, sesuai dengan tujuan masing-masing program.

- a. Tuliskan hasil program diatas.

Berikut adalah hasil program pada pertemuan kesepuluh.

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

D:\Document Ndk3\Kuliah> Semester 2> Matematika Diskrit > Praktikum > Pertemuan 10 > checkGraph.cpp

1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4
5 int main() {
6     bool ketemu, nosemua;
7     // [1] Inisialisasi matriks dengan ukuran 10x10 dan variabel lainnya. Matriks ini akan digunakan untuk merepresentasikan graf. Variabel i, j, jumlah_simpul, jumlah_sisi, asal
8     int matrix[10][10];
9     int i, j, jumlah_simpul, jumlah_sisi, asal, tujuan;
10
11    // inisialisasi
12    cout << "jumlah simpul: ";
13    cin >> jumlah_simpul;
14    cout << "jumlah sisi: ";
15    cin >> jumlah_sisi;
16
17    for (i = 1; i < jumlah_simpul; i++) {
18        for (j = 1; j < jumlah_simpul; j++) {
19            matrix[i][j] = 0;
20        }
21    }
22    // [2] Bagian ini mengisi matriks berdasarkan input pengguna. Pengguna memasukkan simpul asal dan tujuan, dan matriks diisi dengan angka 1 pada posisi tersebut. Ini menunjukkan
23    // bahwa ada sisi antara simpul asal dan tujuan, dan matriks sesuai input graf
24    for (i = 1; i < jumlah_sisi; i++) {
25        cout << "simpul asal: ";
26        cin >> asal;
27
28        cout << "simpul tujuan: ";
29        cin >> tujuan;
30
31        matrix[asal][tujuan] = 1;
32    }
33
34    // Cek siap graf
35    if (jumlah_simpul == jumlah_sisi) {
36        nosemua = true;
37
38        while (i < jumlah_simpul && !nosemua) { // jika ada sisi dari simpul asal ke tujuan, maka logo add iiii dari tujuan ke asal. Dik karena itu, matriks diisi dengan angka 1 pada posisi tersebut.
39            ketemu = false;
40
41            for (j = 1; j < jumlah_simpul && !ketemu) {
42                if (matrix[i][j] == 1) {
43                    ketemu = true;
44                }
45                j++;
46            }
47
48            if (ketemu) {
49                nosemua = true;
50            } else {
51                i++;
52            }
53        }
54
55        if (nosemua) {
56            cout << "graf tidak terhubung";
57        } else {
58            cout << "graf terhubung";
59        }
60    }
61    getch();
62    return 0;
63 }

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PS D:\Document Ndk3\Kuliah> Semester 2> Matematika Diskrit>Praktikum>Pertemuan 10> cd "d:\Document Ndk3\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 10"; if ($!) { g++ -c "checkGraph.cpp" } & if ($!) { & ".\tempCodeRunnerFile.exe" }
jumlah_simpul: 4
jumlah_sisi: 3
simpul asal: 1
simpul tujuan: 2
simpul asal: 3
simpul tujuan: 3
simpul asal: 3
simpul tujuan: 4
graf terhubung

Ln 51, Col 17 Spaces:4 UTF-8 CRLF C++ Go Live

File Edit Selection View Go Run Terminal Help checkGraph.cpp - Visual Studio Code - Insiders (Administrator)
D:\Document Ndk3\Kuliah> Semester 2> Matematika Diskrit > Praktikum > Pertemuan 10 > checkGraph.cpp
23 // [1] Inisialisasi matriks berdasarkan input pengguna. Pengguna memasukkan simpul asal dan tujuan, dan matriks diisi dengan angka 1 pada posisi tersebut. Ini menunjukkan bahwa ada sisi antara simpul asal dan tujuan.
24 for (i = 1; i < jumlah_sisi; i++) {
25     cout << "simpul asal: ";
26     cin >> asal;
27
28     cout << "simpul tujuan: ";
29     cin >> tujuan;
30
31     matrix[asal][tujuan] = 1;
32 }
33
34 // Cek siap graf
35 if (jumlah_simpul == jumlah_sisi) {
36     nosemua = true;
37
38     while (i < jumlah_simpul && !nosemua) { // jika ada sisi dari simpul asal ke tujuan, maka logo add iiii dari tujuan ke asal. Dik karena itu, matriks diisi dengan angka 1 pada posisi tersebut.
39         ketemu = false;
40
41         for (j = 1; j < jumlah_simpul && !ketemu) {
42             if (matrix[i][j] == 1) {
43                 ketemu = true;
44             }
45             j++;
46         }
47
48         if (ketemu) {
49             nosemua = true;
50         } else {
51             i++;
52         }
53     }
54
55     if (nosemua) {
56         cout << "graf tidak terhubung";
57     } else {
58         cout << "graf terhubung";
59     }
60 }
61 getch();
62 return 0;
63 }

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PS D:\Document Ndk3\Kuliah> Semester 2> Matematika Diskrit>Praktikum>Pertemuan 10> cd "d:\Document Ndk3\Kuliah\Semester 2\Matematika Diskrit\Praktikum\Pertemuan 10"; if ($!) { g++ -c "tempCodeRunnerFile.cpp" & if ($!) { & ".\tempCodeRunnerFile.exe" }
jumlah_simpul: 3
jumlah_sisi: 3
simpul asal: 1
simpul tujuan: 2
simpul asal: 3
simpul tujuan: 3
simpul asal: 5
simpul tujuan: 3
graf terhubung

PS D:\Document Ndk3\Kuliah> Semester 2> Matematika Diskrit>Praktikum>Pertemuan 10>
```

```

File Edit Selection View Go Run Terminal Help
checkGraph.cpp - Visual Studio Code - Insiders [Administrator]
D:\Document Ndk> Kullah > Semester 2> Matematika Diskrit> Praktikum> Pertemuan 10 > & checkGraph.cpp
23 // isi matrix sesuai input graf
24 for (int i = 0; i < jumlah_simpul; i++) {
25     cout << "simpul[" << i << "]";
26     cin >> asal;
27     cout << "simpul[" << i << "] = ";
28     cin >> tujuan;
29     matrix[asal][tujuan] = 1;
30 }
31 cout << endl;
32 }

33 // telusuri graf
34 int i = 1;
35 noisemua = false;
36
37 while (i <= jumlah_simpul && !noisemua) { // [4] Bagian ini menelusuri graf untuk mengecek apakah graf tersebut terhubung atau tidak. Jika semua simpul dalam graf terhubung, maka graf tersebut dianggap terhubung. Jika ada setidaknya satu simpul yang tidak terhubung, maka graf tersebut dianggap tidak terhubung.
38     i++;
39     ketemu = false;
40     while (jumlah_simpul > i && !ketemu) {
41         if (matrix[i][i] == 1) {
42             ketemu = true;
43         } else {
44             i++;
45         }
46     }
47     if (!ketemu) {
48         noisemua = true;
49     } else {
50         i++;
51     }
52 }
53
54 if (noisemua) {
55     cout << "graf tidak terhubung";
56 } else {
57     cout << "graf terhubung";
58 }
59
60 getch();
61 return 0;
62 }

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PS D:\Document Ndk\Kullah> cd "D:\Document Ndk\Kullah\Semester 2>Matematika Diskrit>Praktikum>Pertemuan 10"> & checkGraph.cpp
jumlah_simpul: 5
simpul asal: 0
simpul tujuan: 5
simpul asal: 1
simpul tujuan: 2
simpul asal: 2
simpul tujuan: 4
simpul tujuan: 3
graf terhubung

File Edit Selection View Go Run Terminal Help
checkGraph.cpp - Visual Studio Code - Insiders [Administrator]
D:\Document Ndk> Kullah > Semester 2> Matematika Diskrit> Praktikum > Pertemuan 10 > & checkGraph.cpp
22 // [2] Bagian ini mengisi matriks berdasarkan input pengguna. Pengguna memasukkan simpul asal dan tujuan, dan matriks dilihi dengan angka 1 pada posisi tersebut. Ini menunjukkan bahwa ada sisi antara simpul asal dan tujuan.
23 // isi matrix sesuai input graf
24 for (int i = 0; i < jumlah_simpul; i++) {
25     cout << "simpul[" << i << "]";
26     cin >> asal;
27     cout << "simpul[" << i << "] = ";
28     cin >> tujuan;
29     matrix[asal][tujuan] = 1;
30 }
31 cout << endl;
32 }

33 // telusuri graf
34 int i = 1;
35 noisemua = false;
36
37 while (i <= jumlah_simpul && !noisemua) { // [4] Bagian ini menelusuri graf untuk mengecek apakah graf tersebut terhubung atau tidak. Jika semua simpul dalam graf terhubung, maka graf tersebut dianggap terhubung.
38     i++;
39     ketemu = false;
40     while (i < jumlah_simpul && !ketemu) {
41         if (matrix[i][i] == 1) {
42             ketemu = true;
43         } else {
44             i++;
45         }
46     }
47     if (!ketemu) {
48         noisemua = true;
49     } else {
50         i++;
51     }
52 }
53
54 if (noisemua) {
55     cout << "graf tidak terhubung";
56 } else {
57     cout << "graf terhubung";
58 }
59
60 getch();
61 return 0;
62 }

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE
PS C:\Users\Ireddragon\cyg> cd "D:\Document Ndk\Kullah\Semester 2>Matematika Diskrit>Praktikum>Pertemuan 10"> & checkGraph.cpp
jumlah_simpul: 5
simpul asal: 1
simpul tujuan: 1
simpul asal: 2
simpul tujuan: 2
simpul tujuan: 3
graf tidak terhubung

```

```

File Edit Selection View Go Run Terminal Help
D:\Document Ndk\Kuliah> Semester 2 > Matematika Diskrit > Praktikum > Pertemuan 10 > checkGraph.cpp
22 // [2] Bagian ini mengisi matriks berdasarkan input pengguna. Pengguna menasukan simpul asal dan tujuan, dan matriks diisi dengan angka 1 pada posisi tersebut. Ini menunjukkan bahwa ada sisi antara simpul asal da
23 // [3] Bagian ini menulis matriks graf
24 for (i = 0; i < jumlah_simpul; i++) {
25     cout << "simpul asal: ";
26     cin >> asal;
27     cout >> "simpul tujuan: ";
28     cin >> tujuan;
29     matrix[asal][tujuan] = 1;
30     // [3] Karena ini adalah graf tidak berarah, maka jika ada sisi dari simpul asal ke tujuan, maka juga ada sisi dari tujuan ke asal. Oleh karena itu, matriks diisi dengan angka 1 pada posisi tersebut.
31     matrix[tujuan][asal] = 1;
32 }
33
34 // telusuri graf
35 i = 1;
36 nosemua = false;
37
38 while (i < jumlah_simpul && !nosemua) { // [4] Bagian ini mensusuri graf untuk mengecek apakah graf tersebut terhubung atau tidak. Jika semua simpul dalam graf terhubung, maka graf tersebut dianggap terhubung.
39     j = 1;
40     ketemu = false;
41     while (j < jumlah_simpul && !ketemu) {
42         if (matrix[i][j] == 1) {
43             ketemu = true;
44         } else {
45             j++;
46         }
47     }
48     if (ketemu) {
49         nosemua = true;
50     } else {
51         i++;
52     }
53 }
54
55 if (nosemua) {
56     cout << "graf tidak terhubung";
57 }

```

Gambar 27 Beberapa contoh output program. (Sumber: Penulis)

b. Tuliskan komentar yang sesuai dari no {1,2,3, 4} pada program di atas



Gambar 28 Komentar pertama. (Sumber: Penulis)

[1] Inisialisasi matriks dengan ukuran 10x10 dan variabel lainnya. Matriks ini akan digunakan untuk merepresentasikan graf. Variabel i, j, jumlah\_simpul, jumlah\_sisi, asal, dan tujuan digunakan untuk iterasi dan input pengguna.

Komentar [1] tersebut menjelaskan bahwa pada bagian tersebut dilakukan inisialisasi matriks dengan ukuran 10x10 dan beberapa variabel lainnya. Matriks tersebut akan digunakan untuk merepresentasikan graf. Variabel i dan j dengan tipe data integer digunakan untuk iterasi pada matriks, sedangkan variabel jumlah\_simpul dan jumlah\_sisi dengan tipe data integer digunakan untuk menyimpan jumlah simpul dan sisi pada graf. Variabel asal dan tujuan dengan tipe data integer digunakan untuk menyimpan simpul asal dan simpul tujuan pada graf.

Selain itu, pada bagian tersebut juga dideklarasikan variabel matrix dengan tipe data integer dan ukuran 10x10. Variabel matrix ini akan digunakan untuk merepresentasikan graf dalam bentuk matriks. Dengan demikian, pada bagian tersebut dilakukan inisialisasi variabel-variabel yang akan digunakan pada program untuk merepresentasikan graf dalam bentuk matriks dan juga untuk melakukan iterasi dan input pengguna.

```

21 // [2] Bagian ini mengisi matriks berdasarkan input pengguna. Pengguna memasukkan simpul asal dan tujuan, dan matriks diisi dengan angka 1 pada posisi tersebut. Ini menunjukkan bahwa ada sisi antara simpul asal dan tujuan.
22 // isi matriks sebagaimana input graf
23 for (int i = 0; i < jumlah_sisi; i++) {
24     cout << "simpul asal: ";
25     cin >> asal;
26     cout << "simpul tujuan: ";
27     cin >> tujuan;
28     matriks[asal][tujuan] = 1;
29     // [3] Karena ini adalah graf tidak berarah, maka jika ada sisi dari simpul asal ke tujuan, maka juga ada sisi dari tujuan ke asal. Oleh karena itu, matriks diisi dengan angka 1 pada posisi tersebut.
30     matriks[tujuan][asal] = 1;
31 }
32
33

```

Gambar 29 Komentar kedua. (Sumber: Penulis)

[2] Bagian ini mengisi matriks berdasarkan input pengguna. Pengguna memasukkan simpul asal dan tujuan, dan matriks diisi dengan angka 1 pada posisi tersebut. Ini menunjukkan bahwa ada sisi antara simpul asal dan tujuan.

Komentar [2] tersebut menjelaskan bahwa pada bagian tersebut dilakukan pengisian matriks berdasarkan input pengguna. Pengguna diminta untuk memasukkan simpul asal dan tujuan, dan matriks akan diisi dengan angka 1 pada posisi tersebut. Hal ini menunjukkan bahwa ada sisi antara simpul asal dan tujuan pada graf.

Selain itu, pada bagian tersebut juga terdapat perulangan for yang digunakan untuk meminta input pengguna sebanyak jumlah\_sisi kali. Variabel asal dan tujuan akan diisi dengan input pengguna, dan matriks akan diisi dengan angka 1 pada posisi yang sesuai dengan simpul asal dan tujuan yang dimasukkan.

Dengan demikian, pada bagian tersebut dilakukan pengisian matriks berdasarkan input pengguna untuk merepresentasikan sisi-sisi pada graf.

```

30 // [3] Karena ini adalah graf tidak berarah, maka jika ada sisi dari simpul asal ke tujuan, maka juga ada sisi dari tujuan ke asal. Oleh karena itu, matriks diisi dengan angka 1 pada posisi tersebut.
31 matriks[tujuan][asal] = 1;
32

```

Gambar 30 Komentar ketiga. (Sumber: Penulis)

[3] Karena ini adalah graf tidak berarah, maka jika ada sisi dari simpul asal ke tujuan, maka juga ada sisi dari tujuan ke asal. Oleh karena itu, matriks diisi dengan angka 1 pada posisi tersebut.

Komentar [3] tersebut menjelaskan bahwa karena graf yang direpresentasikan adalah graf tidak berarah, maka jika ada sisi dari simpul asal ke tujuan, maka juga ada sisi dari tujuan ke asal. Oleh karena itu, pada bagian tersebut matriks diisi dengan angka 1 pada posisi yang sesuai dengan simpul asal dan tujuan yang dimasukkan, baik pada posisi [asal][tujuan] maupun [tujuan][asal].

Dengan demikian, pada bagian tersebut dilakukan pengisian matriks dengan angka 1 pada posisi yang sesuai dengan simpul asal dan tujuan yang dimasukkan, baik pada posisi [asal][tujuan] maupun [tujuan][asal], untuk merepresentasikan sisi-sisi pada graf yang tidak berarah.

```

while (i < jumlah_simpul && !nolsemua) { // [4] Bagian ini menelusuri graf untuk mengecek apakah graf tersebut terhubung atau tidak. Jika semua simpul dalam graf terhubung, maka graf tersebut dianggap terhubung. Jika ada setidaknya satu simpul yang tidak terhubung dengan simpul lain, maka graf tersebut dianggap tidak terhubung
    ketemu = false;
    while (ketemu != true) {
        if (outdegree[i][j] == 1) {
            ketemu = true;
        } else {
            j++;
        }
    }
    if (ketemu) {
        nolsemua = true;
    } else {
        j++;
    }
}
if (nolsemua) {
    cout << "graf tidak terhubung";
} else {
    cout << "graf terhubung";
}

```

Gambar 31 Komentar keempat. (Sumber: Penulis)

[4] Bagian ini menelusuri graf untuk mengecek apakah graf tersebut terhubung atau tidak. Jika semua simpul dalam graf terhubung, maka graf tersebut dianggap terhubung. Jika ada setidaknya satu simpul yang tidak terhubung dengan simpul lain, maka graf tersebut dianggap tidak terhubung.

Komentar [4] tersebut menjelaskan bahwa pada bagian tersebut dilakukan penelusuran graf untuk mengecek apakah graf tersebut terhubung atau tidak. Jika semua simpul dalam graf terhubung, maka graf tersebut dianggap terhubung. Namun, jika ada setidaknya satu simpul yang tidak terhubung dengan simpul lain, maka graf tersebut dianggap tidak terhubung.

Pada bagian tersebut terdapat perulangan while yang digunakan untuk menelusuri setiap simpul pada graf. Variabel i digunakan untuk menyimpan nomor simpul yang sedang diperiksa, sedangkan variabel j digunakan untuk menelusuri setiap simpul yang terhubung dengan simpul i. Variabel ketemu digunakan untuk menyimpan informasi apakah simpul i terhubung dengan simpul lain atau tidak. Variabel nolsemua digunakan untuk menyimpan informasi apakah semua simpul dalam graf terhubung atau tidak.

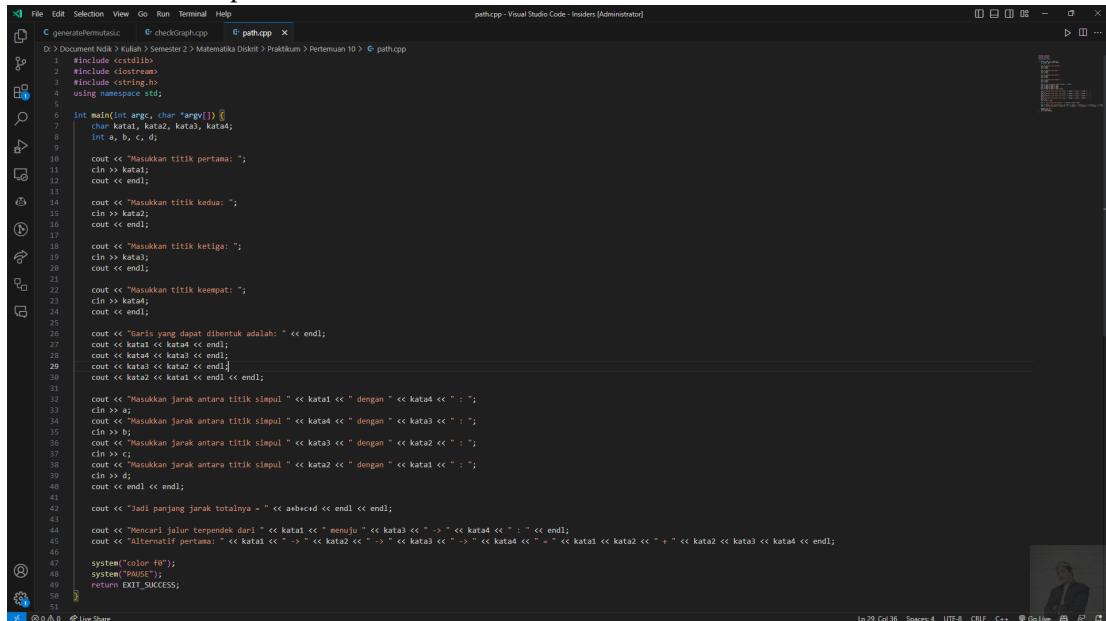
Pada setiap iterasi perulangan while pertama, variabel j diinisialisasi dengan nilai 1 dan variabel ketemu diinisialisasi dengan nilai false. Kemudian, dilakukan perulangan while kedua untuk menelusuri setiap simpul yang terhubung dengan simpul i. Jika ditemukan simpul yang terhubung dengan simpul i, maka variabel ketemu diubah menjadi true dan perulangan while kedua dihentikan. Namun, jika tidak ditemukan simpul yang terhubung dengan simpul i, maka variabel ketemu tetap false dan perulangan while kedua akan terus dilakukan hingga semua simpul yang terhubung dengan simpul i telah diperiksa.

Setelah perulangan while kedua selesai dilakukan, dilakukan pengecekan apakah simpul i terhubung dengan simpul lain atau tidak. Jika simpul i tidak terhubung dengan simpul lain, maka variabel nolsemua diubah menjadi true. Namun, jika simpul i terhubung dengan simpul lain, maka variabel i diubah menjadi nilai selanjutnya dan perulangan while pertama akan dilanjutkan untuk menelusuri simpul selanjutnya.

Dengan demikian, pada bagian tersebut dilakukan penelusuran graf untuk mengecek apakah graf tersebut terhubung atau tidak. Variabel i dan j digunakan untuk menelusuri setiap simpul pada graf dan mengecek apakah simpul tersebut terhubung dengan simpul lain atau tidak. Variabel ketemu digunakan untuk menyimpan informasi apakah simpul i terhubung dengan simpul lain atau tidak, sedangkan variabel nolsemua digunakan untuk menyimpan informasi apakah semua simpul dalam graf terhubung atau tidak.

c. Ketikkan program 2 di bawah ini :

Berikut adalah tampilan sourcecode di visual studio code



```

1 #include <iostream>
2 #include <string.h>
3 using namespace std;
4
5 int main(int argc, char *argv[])
6 {
7     char kata1, kata2, kata3, kata4;
8     int a, b, c, d;
9
10    cout << "Masukkan titik pertama: ";
11    cin >> kata1;
12    cout << endl;
13
14    cout << "Masukkan titik kedua: ";
15    cin >> kata2;
16    cout << endl;
17
18    cout << "Masukkan titik ketiga: ";
19    cin >> kata3;
20    cout << endl;
21
22    cout << "Masukkan titik keempat: ";
23    cin >> kata4;
24    cout << endl;
25
26    cout << "Garis yang dapat dibentuk adalah: " << endl;
27    cout << kata1 << kata4 << endl;
28    cout << kata1 << kata3 << endl;
29    cout << kata3 << kata2 << endl;
30    cout << kata2 << kata1 << endl << endl;
31
32    cout << "Masukkan jarak antara titik simpul " << kata1 << " dengan " << kata4 << " : ";
33    cin >> a;
34    cout << "Masukkan jarak antara titik simpul " << kata4 << " dengan " << kata3 << " : ";
35    cin >> b;
36    cout << "Masukkan jarak antara titik simpul " << kata3 << " dengan " << kata2 << " : ";
37    cin >> c;
38    cout << "Masukkan jarak antara titik simpul " << kata2 << " dengan " << kata1 << " : ";
39    cin >> d;
40    cout << endl << endl;
41
42    cout << "Jadi panjang jarak totalnya = " << abcd << endl << endl;
43
44    cout << "Mencari jalur terpendek dari " << kata1 << " menuju " << kata3 << " -> " << kata4 << " : " << endl;
45    cout << "Alternatif pertama: " << kata1 << " -> " << kata2 << " -> " << kata3 << " -> " << kata4 << " = " << kata1 << kata2 << " + " << kata2 << kata3 << kata4 << endl;
46
47    system("color 70");
48    system("PAUSE");
49    return EXIT_SUCCESS;
50 }

```

Gambar 32 Postest program. (Sumber: Penulis)

Program C++ ini adalah program interaktif yang meminta pengguna untuk memasukkan empat titik atau simpul dan jarak antara setiap titik tersebut. Program ini kemudian menghitung dan mencetak panjang total dari jarak tersebut dan mencoba mencari jalur terpendek dari titik pertama ke titik ketiga dan keempat.

Berikut adalah penjelasan lebih rinci:

- 1) Program ini meminta pengguna untuk memasukkan empat titik atau simpul, yang disimpan dalam variabel kata1, kata2, kata3, dan kata4.
- 2) Program ini kemudian mencetak garis yang dapat dibentuk dari titik-titik tersebut.
- 3) Selanjutnya, program meminta pengguna untuk memasukkan jarak antara setiap titik atau simpul. Jarak ini disimpan dalam variabel a, b, c, dan d.
- 4) Program kemudian menghitung dan mencetak panjang total dari jarak tersebut, yang merupakan penjumlahan dari a, b, c, dan d.
- 5) Program mencoba mencari jalur terpendek dari titik pertama (kata1) menuju titik ketiga (kata3) dan keempat (kata4). Namun, perlu dicatat bahwa program ini tidak benar-benar melakukan perhitungan untuk menemukan jalur terpendek. Sebaliknya, ia hanya mencetak alternatif pertama yang melibatkan perjalanan dari kata1 ke kata2, kemudian ke kata3, dan akhirnya ke kata4.

- 6) Akhirnya, program mengubah warna teks dan latar belakang konsol dengan perintah `system("color f0")`, dan kemudian mem-pause program dengan perintah `system("PAUSE")` sebelum akhirnya keluar dengan status sukses.

Berikut adalah outputnya:

The screenshot shows a Visual Studio Code window with the following details:

- File Explorer:** Shows files like `generatorpermuatuc`, `checkGrph.cpp`, and `path.cpp`.
- Code Editor:** Displays the `path.cpp` file content:

```
1 #include <cstdlib>
2 #include <iostream>
3 #include <sstream.h>
4 using namespace std;
5
6 int main(int argc, char *argv[])
7 {
8     char kata1, kata2, kata3, kata4;
9     int a, b, c, d;
10
11     cout << "Masukkan titik pertama: ";
12     cin >> kata1;
13     cout << endl;
14
15     cout << "Masukkan titik kedua: ";
16     cin >> kata2;
17     cout << endl;
18
19     cout << "Masukkan titik ketiga: ";
20     cin >> kata3;
21     cout << endl;
22 }
```
- Terminal:** Shows the command to build and run the program, followed by user input and the program's output.

```
matika-Disk11\PRAktikum\Perfumeman 10>; if ($?) { g++ -O3 "path.cpp" -o "path.exe" }; if ($?) { & ".\path.exe" }
Masukkan titik pertama: 2
Masukkan titik kedua: 2
Masukkan titik ketiga: 3
Masukkan titik keempat: 4
Jenis yang dapat dibentuk adalah:
14
43
32
21

Masukkan jarak antara titik simpul 1 dengan 4 : 2
Masukkan jarak antara titik simpul 4 dengan 3 : 3
Masukkan jarak antara titik simpul 3 dengan 2 : 4
Masukkan jarak antara titik simpul 2 dengan 1 : 5

Jadi panjang jarak totalnya = 14
Mencari jalur terpendek dari 1 menuju 3 -> 4 :
alternatif pertama: 1 -> 2 -> 3 -> 4 = 12 + 234
Press any key to continue . . .
```
- Status Bar:** Shows the current file is `path.cpp - Visual Studio Code - Insiders (Administrator)`, and the bottom right corner shows "In 15 Col 18 Spaces: 4 UTM-B CR LF C++ Go Live".

Untuk mengakses kodingan, dapat melihat link github berikut.

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