**LAPORAN PRAKTIKUM**

**ALGORITMA DAN STRUKTUR DATA**

**HALAMAN JUDUL**

****

**Disusun Oleh :**

**NAMA : ROYHAN ABDURROHIM**

**NIM : 32602200018**

**PRODI TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS ISLAM SULTAN AGUNG**

**SEMARANG**

**2022**

**HALAMAN PENGESAHAN**

**Laporan Praktikum**

**Algoritma dan Struktur Data**

Disusun Oleh :

Royhan Abdurrohim

(32602200018)

Telah di setujui sebagai syarat untuk memenuhi mata kuliah Praktikum Algoritma dan Struktur Data. Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang.

Hari :

Tanggal :

Asisten Praktikum

1. Bagas Afza Joko Ariyanto 1...............

|  |  |
| --- | --- |
| Mengetahui, | |
| Laboran | Dosen Praktikum |
|  |  |
| Werdha Wilubertha H, S.Kom | Badieah, S.T., M.Kom |

**KATA PENGANTAR**

Puji syukur penulis panjatkan kepada ALLAH SWT, yang telah memberikan rahmat, taufik serta hidayah-Nya, sehingga laporan Algoritma dan Struktur Data dapat terselesaikan.

Tanpa lupa penulis mengucapkan terima kasih kepada :

1. Rektor UNISSULA Bapak Prof. Dr. H. Gunarto, S.H., M.H yang mengijinkan penulis menimba ilmu di kampus ini
2. Dekan Fakultas Teknologi Industri Ibu Dr. Ir. Hj.Novi Marlyana, S.T., M.T
3. Dosen pengampu penulis Bapak/Ibu Badieah, S.T., M.Kom yang telah memberi ilmu tentang Algoritma dan Struktur Data
4. Asisten dosen yang telah membantu menjelasan menegenai Praktikum Algoritma dan Struktur Data
5. Orang tua penulis yang telah mengijinkan untuk menyelesaikan laporan ini,
6. Dan kepada semua pihak yang tidak dapat saya satu persatu.

Penulis menyadari bahwa dalam penyusunan laporan ini masih terdapat banyak kekuranggan, untuk itu penulis mengharap kritik dan saran dari pembaca untuk sempurnanya laporan ini. Semoga dengan ditulisnya laporan ini dapat menjadi sumber ilmu bagi setiap pembaca.

|  |
| --- |
| Semarang, <TGL TTD PENULIS> |
|  |
| Royhan Abdurrohim |

**DAFTAR ISI**

**DAFTAR GAMBAR**

**DAFTAR TABEL**

# BAB I

# STRUKTUR DASAR ALGORITMA

1. Tujuan

Adapun tujuan dari diadakannya praktikum ini:

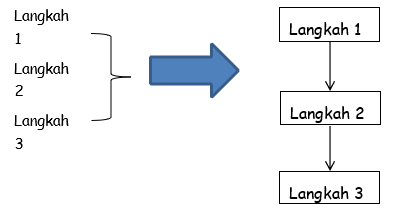
1. Mahasiswa dapat mengetahui dan memahami macam struktur dasar algoritma
2. Mahasiswa dapat mengetahui dan memahami struktur sekuensial
3. Mahasiswa dapat mengetahui dan memahami struktur seleksi
4. Mahasiswa dapat mengetahui dan memahami struktur perulangan
5. Dasar Teori

### Sekuensial

Suatu pemrograman tidaklah terlepas dari algoritma sebagai acuan untuk membuat program. Pemrograman sekuensial merupakan runtunan (sekuensial / *sequence*) satu atau lebih instruksi, yang berarti bahwa:

* Tiap instruksi dikerjakan satu per satu
* Tiap instruksi dilaksanakan tepat satu kali; tidak ada instruksi yang di ulang
* Urutan instruksi yang dilaksanakan pemroses sama dengan urutan instruksi sebagaimana yang tertulis di dalam teks algoritmanya.
* Akhir dari instruksi terakhir merupakan akhir algoritma.

Pada struktur sekuensial, langkah-langkah yang dilakukan dalam algoritma diproses secara berurutan, sebagaima diperlihatkan pada gambar berikut:



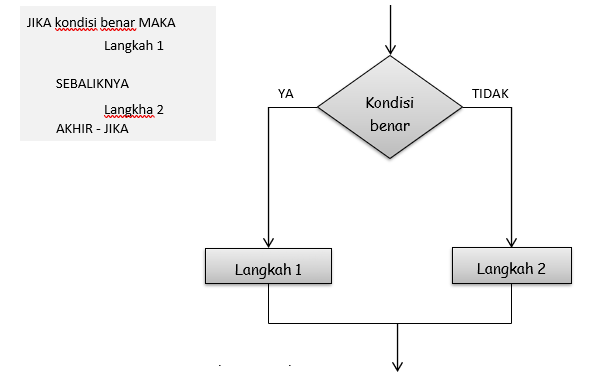
Gambar 1.1 Struktur Sekuensial

Pada contoh tersebut, langkah 1 merupakan langkah yang akan dijalankan pertama kali. Setelah itu, langkah 2 dikerjakan dan diikuti dengan langkah 3.

### Seleksi

Seleksi digunakan unuk mengarahkan pencabangan aliran eksekusi program. Eksekusi program diarahkan sesuai kondisi yang mengendalikannya. Aliran seleksi dalam C++ dinyatakan dalam bentuk if dan switch. Aliran ini digunakan untuk menentukan alur jalannya eksekusi program. Alur eksekusi program ditentukan dari nilai kebenaran kondisi yang di tentukan.

Struktur seleksi menyatakan pemilihan langkah yang didasarkan oleh suatu kondisi (pengambilan keputusan). Gambar 1.2 memperlihatkan diagram alir struktur seleksi yang melibatkan dua alternatif. Dalam hal ini, simbol belah ketupat digunakan untuk mewakili langkah pengambilan keputusan.



Gambar 1.2 Struktur Seleksi

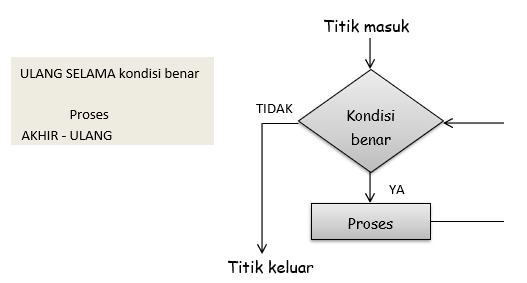
Pada struktur tersebut, langkah 1 hanya akan dijalankan jika kondisi bernilai benar, sedangkan langkah 2 hanya akan dijalankan jika kondisi bernilai salah.

### Pengulangan

Pengulangan adalah suatu tindakan melakukan hal yang sama berulang-ulang kali. Penting untuk dilakukan suatu pengulangan atau perulangan dalam pembuatan suatu program dengan tujuan untuk mengefisienkan kerja program ataupun mengefisienkan listing dari coding dari program tersebut, karena programer tidak perlu mengulang proses yang sama. Untuk perulangan pada c++ dapat dilakukan dengan menggunakan pernyataan for, while dan do-while.

Pengulangan menyatakan suatu tindakan atau langkah yang dijalankan beberapa kali. Struktur pengulangan menyatakan perwujudan keadaan seperti itu. Sebagai contoh, jika anda ingin menampilkan 10 tulisan “SELAMAT BELAJAR”, Anda bisa menuliskannya dengan menggunakan struktur sekuensial. Hal itu berarti Anda memberikan 10 instruksi untuk menuliskan kesepuluh tulisan tersebut. Cara seperti itu memang praktis untuk jumlah pengulangan yang sedikit (misalnya 2 atau 3 pengulangan), tetapi tidak cocok untuk jumlah yang besar. Agar lebih praktis, Anda bisa menggunakan struktur pengulangan.

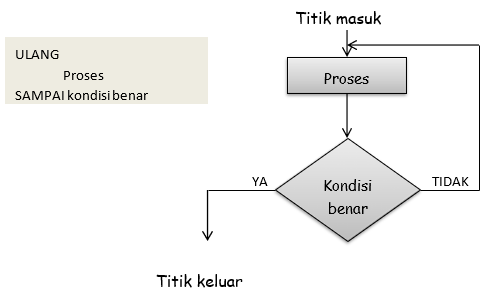
**Struktur pengulangan pertama**



Gambar 1.3 Perulangan Bentuk Pertama

Pada struktur tersebut, proses dapat berupa satu atau beberapa langkah. Pada bentuk ini, ada kemungkinan proses tidak dijalankan sama sekali sekiranya di awal kondisi bernilai salah. Diagram alir pada Gambar 1.3 menunjukkan bahwa sebelum proses dijalankan pertama kali, kondisi diuji terlebih dahulu. Sekiranya kondisi bernilai benar maka proses dijalankan, kemudian kondisi diuji lagi. Sepanjang kondisi masih bernilai benar proses akan tetap dijalankan. Namun, begitu kondisi bernilai salah maka pengulangan berakhir.

**Struktur pengulangan kedua**



Gambar 1.4 Perulangan Bentuk Kedua

Pada bentuk kedua, proses paling tidak dijalankan sekali.

### Flowchart

Flowchart adalah urutan proses kegiatan yang digambarkan dalam bentuk simbol. *Flowchart* juga didefinisikan sebagai diagram yang menyatakan aliran proses dengan menggunakan anotasianotasi.

Beberapa simbol yang digunakan didalam *flowchart* adalah sebagai berikut:

1. **(Terminator**) menyatakan titik awal atau titik akhir diagram alir.
2. (**Proses)** menyatakan sebarang proses, misal untuk menyatakan suatu operasi aritmatika.
3. (**Input/output** atau kadang disebut data) digunakan untuk menyatakan operasi pemasukan data atau penampilann data.
4. **(Keputusan**) dilakukan untuk melakukan pengambilan keputusan. Dalam hal ini, yang ada dalam simbol ini berupa suatu pertanyaan yang jawabannya berupa dua kemungkinan, yaitu YA atau TIDAK.
5. (**Konektor**) digunakan untuk menghubungkan ke berbagai bagian dalam diagram alir. (Kadir, 2010)
6. Latihan
7. Contoh seleksi
8. Kode Program

#include <iostream>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

char jenis;

int panjang, lebar, luas, jari;

float luaslingkar;

cout << "Pilih Persegi panjang<P> atau Lingkaran<L>: ";

cin >> jenis;

if (jenis == 'P' || jenis == 'p')

{

cout << "panjang = ";

cin >> panjang;

cout << "lebar = ";

cin >> lebar;

luas = panjang \* lebar;

cout << "Luas = " << luas << endl;

}

else if (jenis == 'L' || jenis == 'l')

{

cout << "jari-jari = ";

cin >> jari;

luaslingkar = 3.14 \* jari \* jari;

cout << "Luas = " << luaslingkar << endl;

}

else

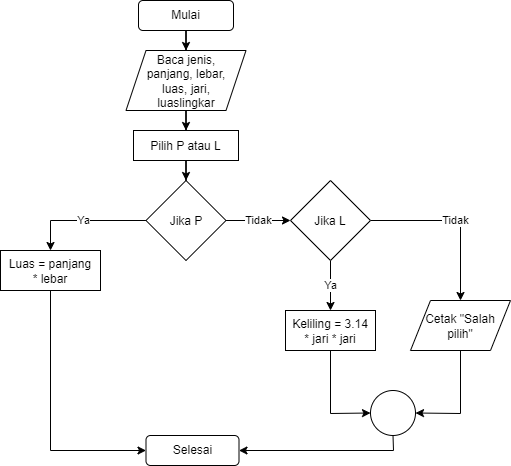
}

cout << "salah pilih" << endl;

}

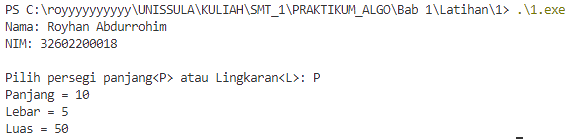
}

1. Flowchart

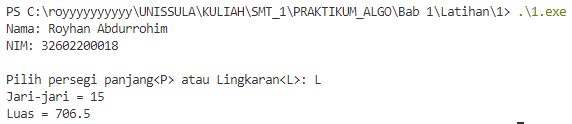


Gambar 1.5 Flowchart Latihan 1

1. Output



Gambar 1.6 Output Latihan 1 Persegi Panjang



Gambar 1.7 Output Latihan 1 Lingkaran

1. Penjelasan

Gambar 1.6 dan gambar 1.7 menunjukkan contoh percabangan

branching (if) antara pilihan P untuk perhitungan luas persegi dan L untuk perhitungan luas lingkaran. Dalam kode perhitungan tersebut menggunakan deklarasi variable dengan tipe data integer, float dan char.

Variabel dengan tipe data char merupakan variabel yang akan menampung pilihan dari masukan pengguna. Jika kita ingin menghitung luas persegi kita masukkan pilihan P. Sehingga output yang akan keluar adalah perhitungan persegi dengan integer panjang dan lebar.

Namun, apabila kita ingin menghitung luas lingkaran, kita masukkan pilihan L. Sehingga output yang keluar adalah perhitungan luas lingkaran dengan integer jari-jari.

1. Contoh perulangan
2. Kode Program

#include <iostream>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int l=0;

for (int j = 0; j <= 10; j++)

{

for (int k = j; k <=l; k++)

{

cout<<k<<' ';

}

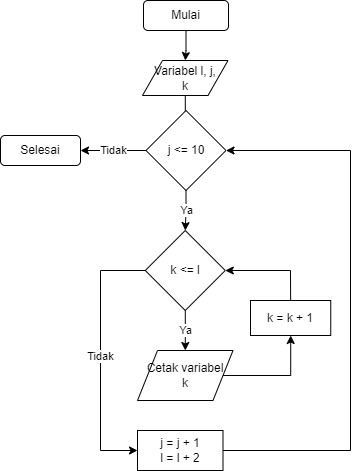
l+=2;

cout<<"\n";

}

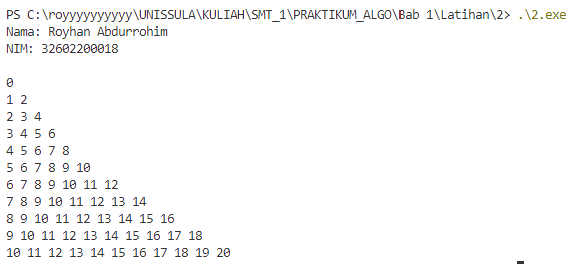
}

1. Flowchart



Gambar 1.8 Flowchart Latihan 2

1. Output



Gambar 1.9 Output Latihan 2

1. Penjelasan

Pada gambar 1.9 merupakan hasil dari perulangan atau *looping.* Untuk melakukan *looping*, kita bisa menggunakan *for* dan *while.* Diatas merupakan contoh *looping* dengan menggunakan for. Bisa kita lihat disini dalam membentuk perulangan for berlaku komposisi for(inisialisasi (nilai awal); condition (syarat perulangan); pengubah nilai pecacahan).

Inisialisasi adalah bagian untuk memberikan nlai awal untul variabel-variabel tertentu. Contohnya ketika kita ingin memulai perhitungan dari angka 9 maka pada bagian inisialisasi dimasukan angka 9 (variabel=9).

Kemudian ada syarat pengulangan. Syarat pengulangan adalah sampai mana kita ingin melakukan *looping* atau bisa di artikan angka terakhir yang kita inginkan itu apa. Contoh ketika kita ingin melakukan *looping* sampai angka 10 kita masukan tanda lebih dari sama dengan <=10 (variabel<=10).

Dan yang terakhir adalah pengubah nilai pecacahan. Pengubah nilai pecacahan adalah mengatur hasil *looping* itu naik atau turun. Contoh kita ingin melakukan *looping* dari terkecil ke terbesar kita masukan variabel++ misal kita ingin kelipatan 2 maka masukan variable +=2. Sedangkan jika ingin dari terbesar ke terkecil masukan variable — jika kita ingin berkelipatan 3 maka masukan variabel -=3.

1. Tugas
2. Modifikasilah program perulangan pada contoh 2 dengan struktur while menggunakan bahasa C.

Jawaban:

1. Kode Program

#include <stdio.h>

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

int l = 0, j = 0, k = 0;

while (j <= 10)

{

k = j;

while (k <= l)

{

printf("%i ", k);

k++;

}

printf("\n");

l += 2;

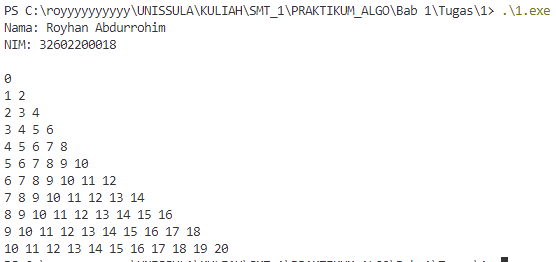
j++;

}

return 0;

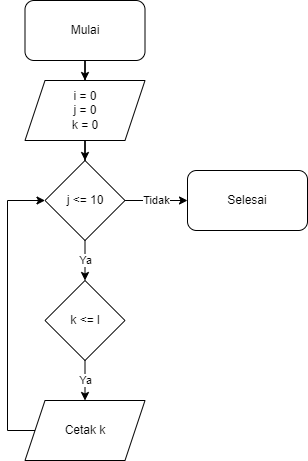
}

1. Output



Gambar 1.10 Output Tugas 1

1. Flowchart



Gambar 1.11 Flowchart Tugas 1

1. Penjelasan

Untuk output dari program ini sama seperti pada contoh nomor 2. Namun perbedaannya ada di Bahasa pemrograman dan struktur perulangan yang digunakan. Pada program ini, saya menggunakan Bahasa pemrograman C dan struktur perulangan while.

Secara umum, syntax pada Bahasa pemrograman C dan C++ sebenarnya cukup mirip. Jika di C++ kita menggunakan header iostream untuk melakukan input dan output, di C kita menggunakan header stdio.h. Kemudian jika di C++ kita menggunakan fungsi cout untuk mengeluarkan ouput, di C kita menggunakan fungsi printf.

1. Buatlah program yang dapat menghitung luas, keliling, dan diagonal persegipanjang. Diagonal dihitung dengan rumus:

diagonal = akar(panjang2 + lebar2). akar dapat dikerjakan dengan fungsi sqrt().

Masukan berupa panjang dan lebar. Program berupa pilihan untuk luas, keliling, dan diagonal.

Jawaban:

1. Kode Program

#include <iostream>

#include <cmath>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

char jenis;

int panjang, lebar;

cout << "Fungsi persegi panjang" << endl;

cout << "Luas (L)" << endl;

cout << "Keliling (K)" << endl;

cout << "Diagonal (D)" << endl;

cout << "Pilih fungsi persegi panjang: ";

cin >> jenis;

jenis = tolower(jenis);

if (jenis != 'l' && jenis != 'k' && jenis != 'd')

{

cout << "Fungsi yang anda masukkan salah";

return 0;

}

cout << "Masukkan panjang: ";

cin >> panjang;

cout << "Masukkan lebar: ";

cin >> lebar;

if (jenis == 'l')

{

int luas = panjang \* lebar;

cout << "Luas = " << luas;

}

else if (jenis == 'k')

{

int keliling = (2 \* panjang) + (2 \* lebar);

cout << "Keliling = " << keliling;

}

else if (jenis == 'd')

{

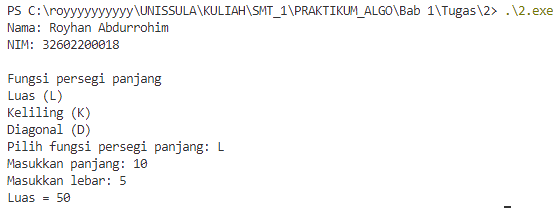
float diagonal = sqrt((panjang \* panjang) + (lebar \* lebar));

cout << "Diagonal = " << diagonal;

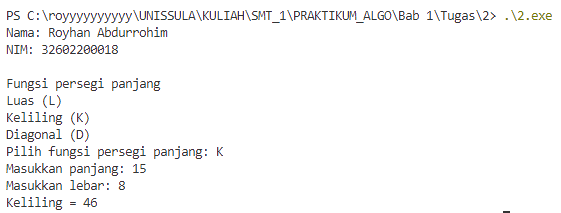
}

}

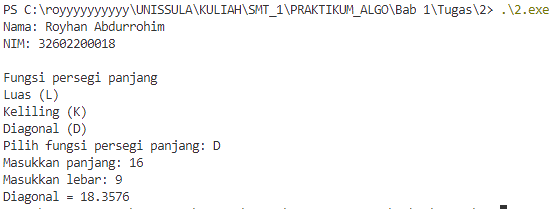
1. Output



Gambar 1.12 Output Tugas 2 Luas

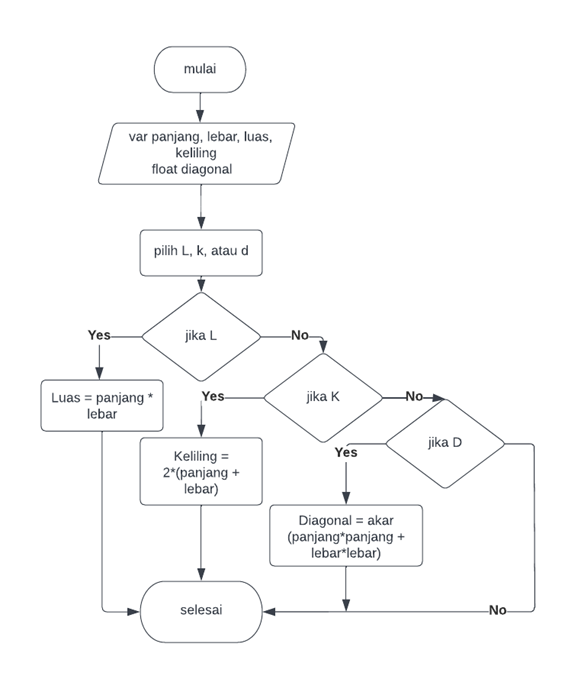


Gambar 1.13 Output Tugas 2 Keliling



Gambar 1.14 Output Tugas 2 Diagonal

1. Flowchart



Gambar 1.15 Flowchart Tugas 2

1. Penjelasan

Program ini dimulai dengan memasukkan header, header yang saya gunakan adalah iostream dan cmath. Header cmath digunakan untuk melakukan perhitungan matematika yang lebih kompleks. Pada program ini sendiri, nantinya akan digunakan untuk menghitung diagonal persegi panjang.

Saya menggunakan beberapa tipe data di program ini, yaitu: char untuk input tipe fungsi persegi panjang, int untuk input panjang dan lebar; hasil perhitungan luas; hasil perhitungan keliling, float untuk hasil perhitungan diagonal.

Struktur seleksi digunakan pada program ini untuk menyeleksi fungsi persegi panjang yang akan digunakan sesuai dengan input dari pengguna.

1. Modifikasilah program nomor 2 menjadi bahasa C

Jawaban:

1. Kode Program

#include <stdio.h>

#include <ctype.h>

#include <math.h>

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

char jenis;

int panjang, lebar;

printf("Fungsi persegi panjang\n");

printf("Luas (L)\n");

printf("Keliling (K)\n");

printf("Diagonal (D)\n");

printf("Pilih fungsi persegi panjang: ");

scanf("%s", &jenis);

jenis = tolower(jenis);

if (jenis != 'l' && jenis != 'k' && jenis != 'd')

{

printf("Fungsi yang anda masukkan salah");

return 0;

}

printf("Masukkan panjang: ");

scanf("%d", &panjang);

printf("Masukkan lebar: ");

scanf("%d", &lebar);

if (jenis == 'l')

{

int luas = panjang \* lebar;

printf("Luas = %d", luas);

}

else if (jenis == 'k')

{

int keliling = (2 \* panjang) + (2 \* lebar);

printf("Keliling = %d", keliling);

}

else if (jenis == 'd')

{

float diagonal = sqrt((panjang \* panjang) + (lebar \* lebar));

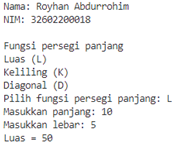
printf("Diagonal = %f", diagonal);

}

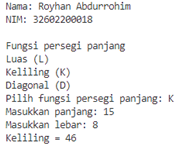
return 0;

}

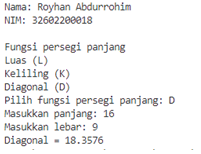
1. Output



Gambar 1,16 Output Tugas 3 Luas

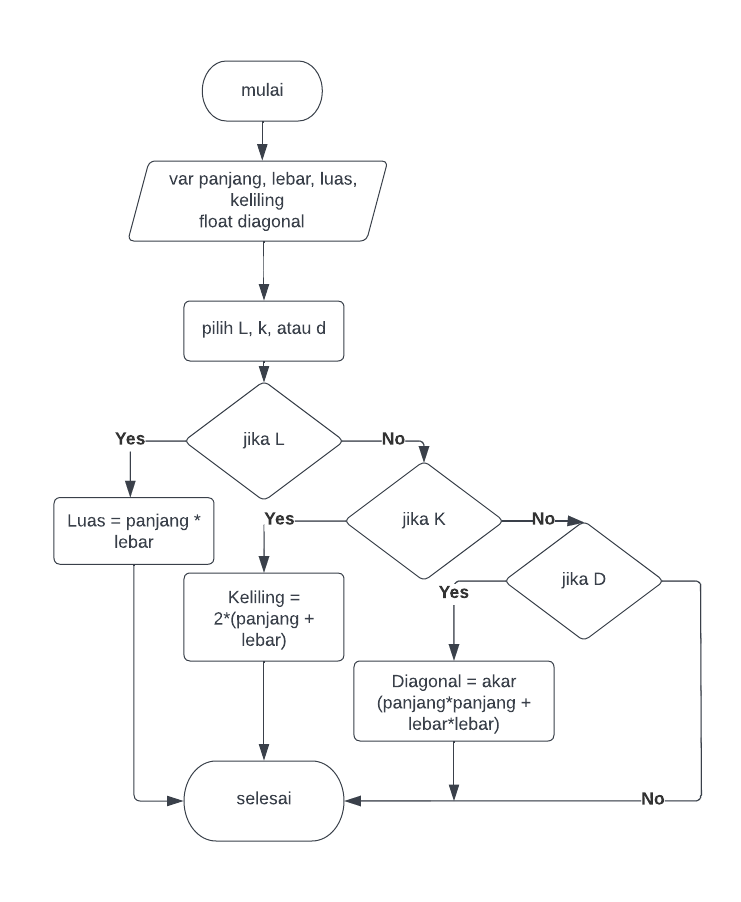


Gambar 1.17 Output Tugas 3 Keliling



Gambar 1.18 Output Tugas 3 Diagonal

1. Flowchart



Gambar 1.19 Flowchart Tugas 3

1. Penjelasan

Program ini memiliki fungsi yang sama seperti pada program nomor 2, perbedaannya adalah program ini ditulis dengan Bahasa C. Jika di Bahasa C++ kita menggunakan header cmath.h untuk dapat menggunakan fungsi sqrt, disini kita menggunakan header math.h. Disini kita juga menggunakan header ctype untuk dapat menggunakan fungsi tolower.

1. Buatlah program agar hasil outputnya menjadi persegi seperti gambar di bawah ini dalam bahasa C, dengan ketentuan panjang sisinya di inputkan.



Jawaban:

1. Kode Program

#include <stdio.h>

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

int panjang;

printf("Masukkan panjang sisi: ");

scanf("%d", &panjang);

for (int i = 0; i < panjang; i++)

{

for (int j = 0; j < panjang; j++)

{

if (i == 0 || i == panjang - 1 || j == 0 || j == panjang - 1)

{

printf("\* ");

}

else

{

printf(" ");

}

}

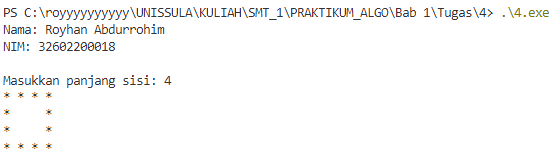
printf("\n");

}

return 0;

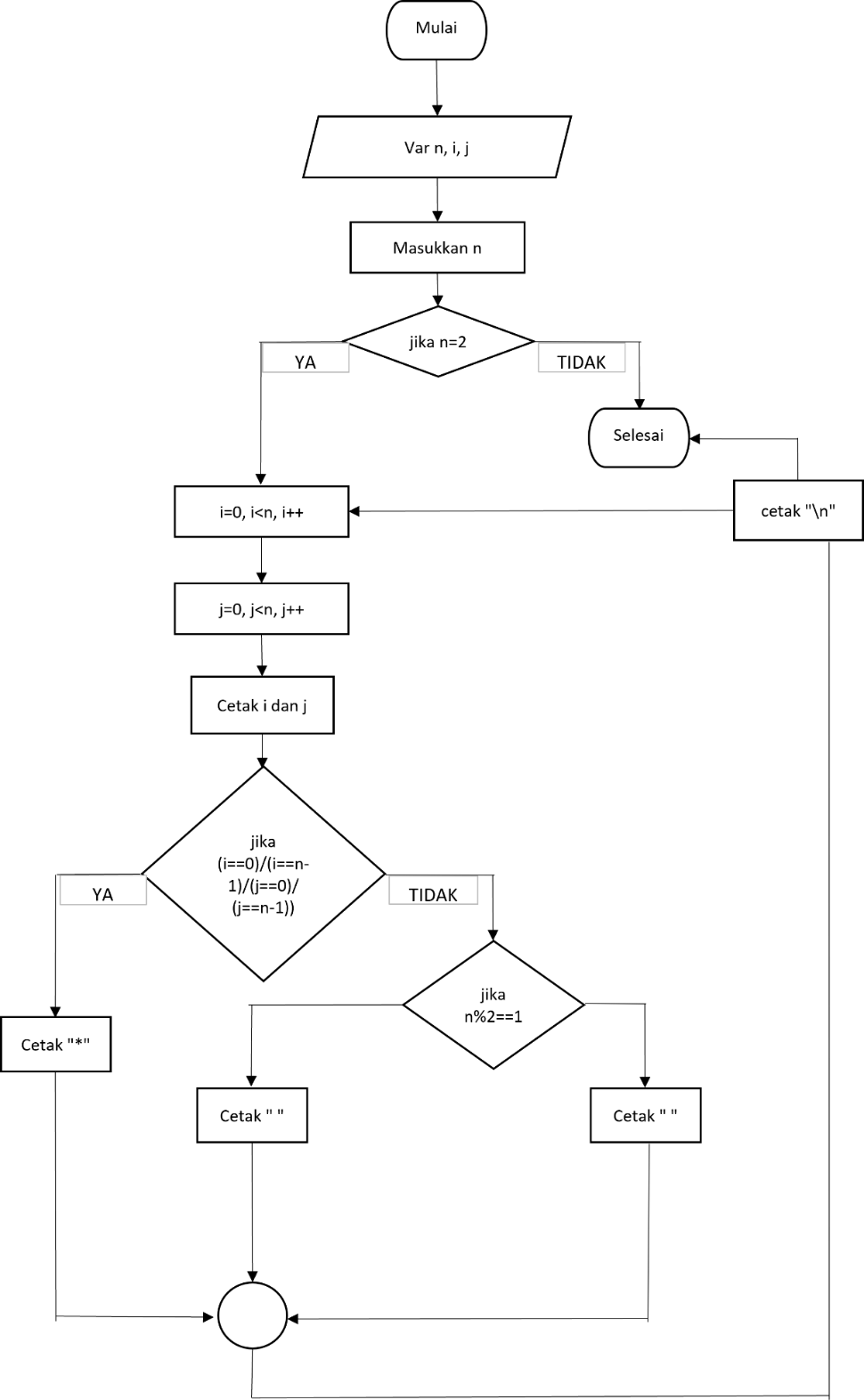
}

1. Output



Gambar 1.20 Output Tugas 4

1. Flowchart



Gambar 1.21 Flowchart Tugas 4

1. Penjelasan

Program ini menggunakan struktur seleksi dan percabangan di dalamnya. Disini saya menggunakan dua perulangan for, perulangan pertama digunakan sebagai indikator pada sisi vertikal, dan perulangan kedua digunakan sebagai indikator pada sisi horizontal.

1. Buatlah simulasi menu program dengan tampilan di bawah ini menggunakan WHILE.

MENU PILIHAN

* 1. Dangdut
  2. Pop
  3. Rock
  4. Exit

Pilihan Anda (1/2/3/4) ? ...

Apabila dipilih menu no 1, maka akan tampil teks “Anda memilih menu 1”. Demikian pula untuk menu 2 dan 3. Kemudian setelah itu muncul teks “Tekan ENTER untuk kembali ke menu utama”. Artinya begitu kita tekan ENTER menu pilihan akan muncul kembali, dst. Akan tetapi bila yang dipilih menu 4 (EXIT), program langsung berhenti.

Jawaban:

1. Kode Program

#include <iostream>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int pilihan;

do

{

cout << "MENU PILIHAN" << endl;

cout << "1. Dangdut" << endl;

cout << "2. Pop" << endl;

cout << "3. Rock" << endl;

cout << "4. Exit" << endl;

cout << "Pilihan Anda (1/2/3/4) ? ";

cin >> pilihan;

cin.ignore();

if (pilihan != 4)

{

cout << "Anda memilih menu " << pilihan << endl;

cout << "Tekan ENTER untuk kembali ke menu utama" << endl;

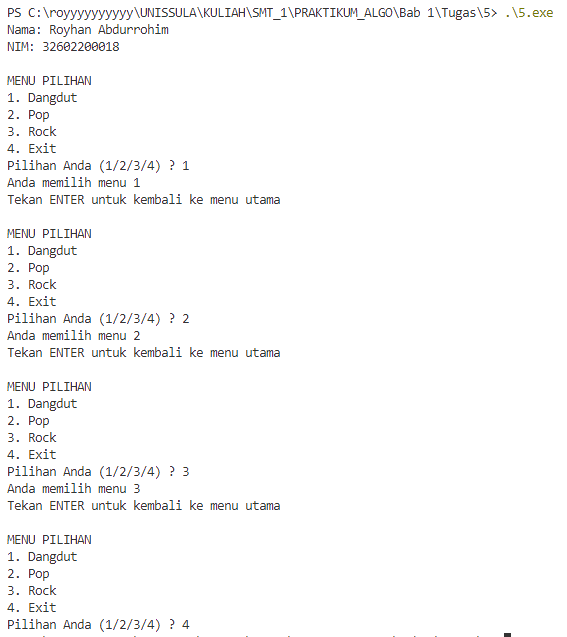
cin.ignore();

}

} while (pilihan != 4);

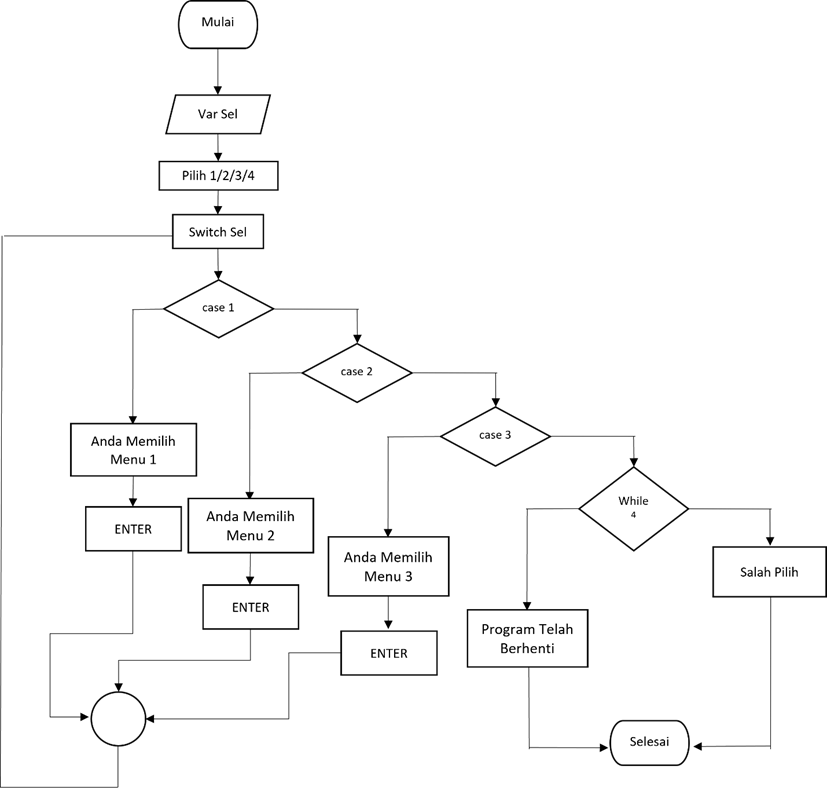
}

1. Output



Gambar 1.22 Output Tugas 5

1. Flowchart



Gambar 1.23 Flowchart Tugas 5

1. Penjelasan

Pada program ini, saya menggunakan perulangan do-while. Pada perulangan do-while, kode yang berada di dalam blok do akan dijalankan setidaknya satu kali walaupun kondisi di dalam while tidak terpenuhi. Itulah sebabnya menu pada program ini bisa muncul pada saat pertama kali dijalankan, walaupun nilai pada variabel pilihan tidak memenuhi kondisi while.

1. Kesimpulan

Kesimpulan pada bab kali ini adalah, dalam membuat program, kita harus familiar dengan sintaks pada bahasa pemrograman yang akan kita gunakan terlebih dahulu. Ini akan membantu kita dalam membuat kode yang efektif dan meminimalisir kemungkinan adanya bug pada program yang kita buat.

Walaupun setiap bahasa pemrograman memiliki sintaks dan ciri khasnya masing-masing, pada dasarnya mereka semua sama dan tidak akan jauh berbeda satu sama lainnya.

Disini kita juga telah mempelajari dan mengaplikasikan beberapa macam struktur dalam pemrograman , yaitu: sekuensial, seleksi dan perulangan.

# BAB II

# ARRAY 2D DAN 3D

## Tujuan

Adapun tujuan dari praktikum ini:

* + 1. Mengenal *array* 2 dimensi dan *array* 3 dimensi
    2. Mengerti dan memahami penggunaan array 2 dimensi dan 3 dimensi
  1. Dasar Teori

### Pengertian Array

*Array* adalah sebuah variabel yang menyimpan sekumpulan data yang memiliki tipe yang sama. Setiap data tersebut menempati lokasi atau alamat memori yang berbeda- beda dan selanjutnya disebut dengan elemen *array.* Elemen *array* itu kemudian dapat kita akses melalui indeks yang terdapat di dalamnya.

Untuk mendeklarasikan sebuah *array* dalam C++, kita harus menggunakan tanda [] (*bracket*).

### Array Multidimensi

*Array* Multidimensi yaitu *array* yang terdiri dari beberapa subskrip *array*. Sebagai contoh, *array* 2 dimensi adalah *array* yang mempunyai 2 subskrip *array*, 3 dimensi mempunyai 3 subskrip *array* dan seterusnya. *Array* seperti ini sering digunakan untuk pemrosesan matriks.

* + - 1. *Array* Dua Dimensi

*Array* dua dimensi adalah *array* yang memilik dua buah elemen bertipe *array*. Dengan kata lain, *array* dua dimensi memiliki dua buah subskrip, yang biasanya dipresentasikan dengan baris dan kolom.

Contoh sederhana adalah data yang tertulis dalam tabel berikut ini :

Tabel 2. 1 Array 2 Dimensi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Merk HP | 1992 | 1993 | 1994 | 1995 |
| 1. | Nokia | 35 | 45 | 80 | 120 |
| 2. | Samsung | 100 | 110 | 70 | 101 |
| 3. | Sony | 10 | 15 | 20 | 17 |

Jika dibuat programnya adalah sebagai berikut :

Int data\_hp [3] [4];

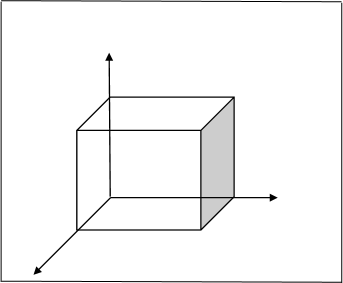
Bentuk umum pendeklarasian sebuah array dua dimensi adalah sebagai berikut :

tipe\_data nama\_array[jumlah\_elemen\_baris][jumlah\_elemen\_kolom] ;

* + - 1. *Array Tiga Dimensi*

Array tiga dimensi merupakan array yang memiliki tiga buah subskrip dan lebih kompleks apabila dibandingkan dengan array dua dimensi. Di sini, subskrip yang ada akan dipresentasikan dengan sumbu x, y, z atau panjang, lebar, dan tinggi seperti yang ditunjukkan oleh gambar berikut.

Bentuk umum pendeklarasian sebuah *array* 3 dimensi adalah sebagai berikut :



Gambar 2. 1 Array 3 Dimensi

tipe\_data nama\_array [jumlah\_elemen\_x] [jumlah\_elemen\_y] [jumlah\_elemen\_z] ;

### Array yang Bersifat Konstant

Nilai dalam elemen *array* dapat dibuat tetap, yaitu dengan mendefinisikannya sebagai konstanta. Caranya sama seperti pada saat kita membuat konstanta dari *identifier*, yaitu dengan menggunakan kata kunci const didepan nama *array* yang didefinisikan.

### Array sebagai tipe data bentukan

Dalam C++, *array* juga dapat digunakan sebagai tipe data bentukan seperti halnya struktur dan enumerasi*.* Untuk melakukan hal ini kita harus menggunakan kata kunci typedef, dimana berfungsi untuk memberikan nama lain dari *array* yang dideklarasikan. Berikut ini bentuk umum untuk membuat *array* sebagai tipe data bentukan.

typedef tipe\_data nama\_array[jumlah\_elemen];

### Array dari Karakter

Dalam C++, kumpulan karakter disebut dengan *string* (teks). Dan untuk mendeklarasikan *array* dari tipe karakter kita tentu akan menuliskannya ke dalam bentuk umum seperti dibawah ini :

char nama\_array[jumlah\_elemen];

Dengan demikian, apabila kita ingin melakukan deklarasi variabel *string* (misalnya dengan nama TEKS) yang terdiri dari 5 buah karakter, maka sintaksnya adalah sebagai berikut :

char TEKS[5] = {“B‟, “U‟, “D‟, “I‟, “\0‟};

Karakter terakhir, “\0” disebut dengan karakter *null*, yaitu karakter yang digunakan sebagai terminator dari sebuah *string* didalam C++. Namun, karena bentuk tersebut susah untuk ditulis dan riskan terhadap terjadinya sebuah kesalahan, maka C++ memperbolehkan kita untuk dapat menuliskan

karakter-karakter tersebut dalam sebuah string yaitu dengan cara seperti dibawah ini :

char TEKS[5] = “BUDI”;

* 1. Latihan
     1. Latihan 1
        1. Kode Program

#include <iostream>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int i, j;

int data[2][2];

cout << "Input Data" << endl;

for (i = 0; i <= 1; i++)

{

for (j = 0; j <= 1; j++)

{

cout << "data[" << i + 1 << "][" << j + 1 << "]=";

cin >> data[i][j];

}

}

cout << "Matrik yang diinputkan : " << endl;

for (i = 0; i <= 1; i++)

{

for (j = 0; j <= 1; j++)

{

cout << data[i][j] << " ";

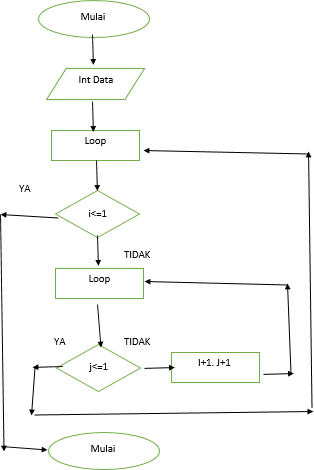
}

cout << endl;

}

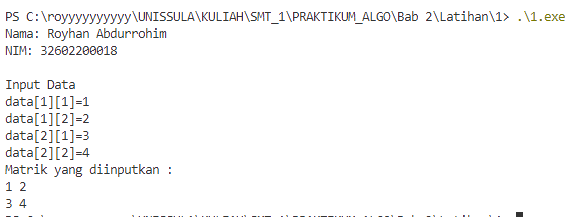
}

* + - 1. *Flowchart*



Gambar 2. 2 *Flowchart* Latihan 1

* + - 1. Output



Gambar 2. 3 Output Latihan 1

* + - 1. Penjelasan

Program pada latihan 1 ini menampilkan program *array* yang berupa matrik 2x2 (2 baris dan 2 kolom). *Header* yang digunakan adalah *iostream* untuk menampilkan *cin, cout, endl*, dan *endl*. Lalu menggunakan tipe data *integer* karena menunjukkan sebuah variabel untuk menyimpan nilai yang dimasukkan, dalam pemasukan nilai harus menggunakan tanda [] (*bracket)* sebagai pendeklarasian bentuk *array.* Pertama kita menginput nilai data yang diinginkan, menggunakan fungsi perulangan *(for*) untuk mengulang suatu proses yang telah diketahui banyaknya. Lalu membuat *output* data matriks yang telah diinputkan.

* + 1. Latihan 2
       1. Kode Program

#include <iostream>

using namespace std;

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

float nilai[2][3];

int baris, kolom;

for (baris = 0; baris < 2; baris++)

{

for (kolom = 0; kolom < 3; kolom++)

{

cout << "Nilai [baris][kolom] ke-[" << baris + 1 << "][" << kolom + 1 << "] = ";

cin >> nilai[baris][kolom];

}

}

cout << endl;

cout << "Matriks Nilai yang dimasukkan : " << endl

<< endl;

for (baris = 0; baris < 2; baris++)

{

for (kolom = 0; kolom < 3; kolom++)

{

cout << nilai[baris][kolom] << " ";

}

cout << endl;

}

cout << endl;

float jml\_nilai1, jml\_nilai2;

float rerata1, rerata2;

jml\_nilai1 = 0;

jml\_nilai2 = 0;

float jml\_nilaicol1, jml\_nilaicol2, jml\_nilaicol3;

float reratacol1, reratacol2, reratacol3;

jml\_nilaicol1 = 0;

jml\_nilaicol2 = 0;

jml\_nilaicol3 = 0;

for (baris = 0; baris < 2; baris++)

{

if (baris == 0)

{

for (kolom = 0; kolom < 3; kolom++)

jml\_nilai1 = jml\_nilai1 + nilai[baris][kolom];

}

else

{

for (kolom = 0; kolom < 3; kolom++)

jml\_nilai2 = jml\_nilai2 + nilai[baris][kolom];

}

}

rerata1 = jml\_nilai1 / 3;

rerata2 = jml\_nilai2 / 3;

cout << "Rerata baris ke 1 = " << rerata1 << endl;

cout << "Rerata baris ke 2 = " << rerata2 << endl;

// per kolom

cout << endl;

for (kolom = 0; kolom < 3; kolom++)

{

if (kolom == 0)

{

for (baris = 0; baris < 2; baris++)

jml\_nilaicol1 = jml\_nilaicol1 + nilai[baris][kolom];

}

else if (kolom == 1)

{

for (baris = 0; baris < 2; baris++)

jml\_nilaicol2 = jml\_nilaicol2 + nilai[baris][kolom];

}

else

{

for (baris = 0; baris < 2; baris++)

jml\_nilaicol3 = jml\_nilaicol3 + nilai[baris][kolom];

}

}

reratacol1 = jml\_nilaicol1 / 2;

reratacol2 = jml\_nilaicol2 / 2;

reratacol3 = jml\_nilaicol3 / 2;

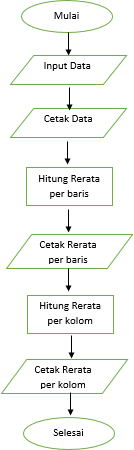
cout << "Rerata kolom ke 1 = " << reratacol1 << endl;

cout << "Rerata kolom ke 2 = " << reratacol2 << endl;

cout << "Rerata kolom ke 3 = " << reratacol3 << endl;

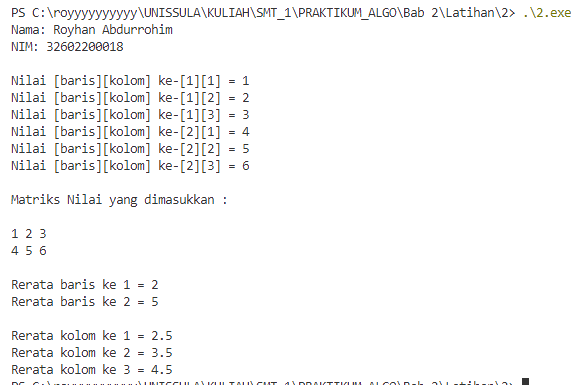
}

* + - 1. Flowchart



Gambar 2. 4 Flowchart Latihan 2

* + - 1. Output



Gambar 2. 5 Output Latihan 2

* + - 1. Penjelasan

Program ini menampilkan array matriks 2x3 (2 baris dan 3 kolom) beserta rerata baris dan kolomnya. *Header* yang digunakan adalah *iostream* yang merupakan *header* pada program C++ untuk menampilkan *cin, cout, endl*, dan *ends.* Tipe data yang digunakan adalah *float* sebagai deklarasi nilai desimal. Menggunakan fungsi perulangan (*for*) untuk mengulang suatu proses yang telah diketahui banyaknya. Menggunakan fungsi (*if*) untuk menyeleksi kondisi, yaitu jika pernyataan pertama tidak sesuai maka diseleksi dipernyataan kedua dan seterusnya. Maka disertakan pula fungsi (*else*) yang merupakan lanjutan dari fungsi (*if*) dimana jika kondisi benar maka statement pertama yang akan dilaksanakan, tetapi jika kondisi salah maka statement yang ada pada blok *else* yang dilaksanakan. Lalu memasukkan inputan baris dan kolom serta nilai matriks yang diinginkan. Memasukkan input rerata baris dan kolom. Jika telah memasukkan input dengan benar maka akan menghasilkan output matriks baris dan kolom yaitu 2 baris dan 3 kolom serta menghasilkan output rerata matriks sesuai dengan baris dan kolomnya.

* + 1. Latihan 3
       1. Kode Program

#include <iostream>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int i, j, k, x = 1;

int matrik[3][4][4];

cout << " 3 dimensi array " << endl

<< endl;

for (i = 0; i < 3; i++)

{

for (j = 0; j < 4; j++)

{

for (k = 0; k < 5; k++)

{

matrik[i][j][k] = x;

cout << " " << matrik[i][j][k];

x++;

}

cout << endl;

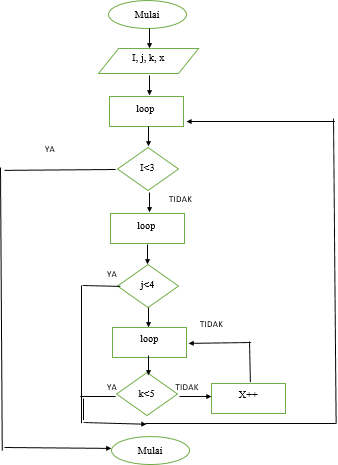
}

cout << endl;

}

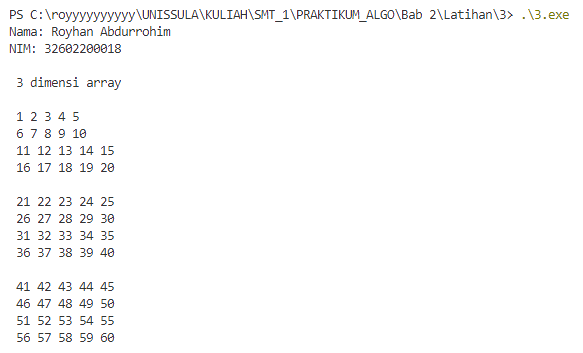
}

* + - 1. Flowchart



Gambar 2. 6 Flowchart Latihan 3

* + - 1. Output



Gambar 2. 7 Output Latihan 3

* + - 1. Penjelasan

Program ini menampilkan *array* 3 dimensi. *Header* yang digunakan adalah *iostream* karena menggunakan bahasa pemrograman C++ yaitu untuk menampilkan *cin, cout, endl,* dan *ends*. Tipe data yang digunakan adalah *integer* yang menampilkan variabel untuk menyimpan nilai yang dimasukkan. Matriks *array* 3 dimensi dapat digambarkan sebagai benda ruang. Program diatas di-*input*-kan 3 dimensi dimana angka-angkanya selalu berselisihan. Dan program tersebut menggunakan fungsi (*for*) untuk mengulang suatu proses yang telah diketahui banyaknya, lalu ada juga fungsi matriks yang berfungsi sebagai kumpulan bilangan, simbol, atau ekspresi.

* 1. Tugas
     1. Ubahlah Latihan 2 kedalam Bahasa C!

Jawaban:

* + - 1. Kode Program

#include <stdio.h>

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

float nilai[2][3];

int baris, kolom;

for (baris = 0; baris < 2; baris++)

{

for (kolom = 0; kolom < 3; kolom++)

{

printf("Nilai [baris][kolom] ke-[%d][%d] = ", baris + 1, kolom + 1);

scanf("%f", &nilai[baris][kolom]);

}

}

printf("\nMatriks Nilai yang dimasukkan : \n\n");

for (baris = 0; baris < 2; baris++)

{

for (kolom = 0; kolom < 3; kolom++)

{

printf("%g ", nilai[baris][kolom]);

}

printf("\n");

}

printf("\n");

float jml\_nilai1, jml\_nilai2;

float rerata1, rerata2;

jml\_nilai1 = 0;

jml\_nilai2 = 0;

float jml\_nilaicol1, jml\_nilaicol2, jml\_nilaicol3;

float reratacol1, reratacol2, reratacol3;

jml\_nilaicol1 = 0;

jml\_nilaicol2 = 0;

jml\_nilaicol3 = 0;

for (baris = 0; baris < 2; baris++)

{

if (baris == 0)

{

for (kolom = 0; kolom < 3; kolom++)

{

jml\_nilai1 = jml\_nilai1 + nilai[baris][kolom];

}

}

else

{

for (kolom = 0; kolom < 3; kolom++)

{

jml\_nilai2 = jml\_nilai2 + nilai[baris][kolom];

}

}

}

rerata1 = jml\_nilai1 / 3;

rerata2 = jml\_nilai2 / 3;

printf("Rerata baris ke 1 = %g\n", rerata1);

printf("Rerata baris ke 2 = %g\n", rerata2);

printf("\n");

for (kolom = 0; kolom < 3; kolom++)

{

if (kolom == 0)

{

for (baris = 0; baris < 2; baris++)

{

jml\_nilaicol1 = jml\_nilaicol1 + nilai[baris][kolom];

}

}

else if (kolom == 1)

{

for (baris = 0; baris < 2; baris++)

{

jml\_nilaicol2 = jml\_nilaicol2 + nilai[baris][kolom];

}

}

else

{

for (baris = 0; baris < 2; baris++)

{

jml\_nilaicol3 = jml\_nilaicol3 + nilai[baris][kolom];

}

}

}

reratacol1 = jml\_nilaicol1 / 2;

reratacol2 = jml\_nilaicol2 / 2;

reratacol3 = jml\_nilaicol3 / 2;

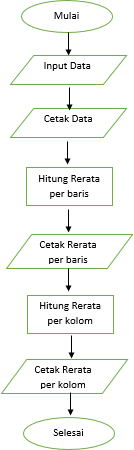
printf("Rerata kolom ke 1 = %g\n", reratacol1);

printf("Rerata kolom ke 2 = %g\n", reratacol2);

printf("Rerata kolom ke 3 = %g\n", reratacol3);

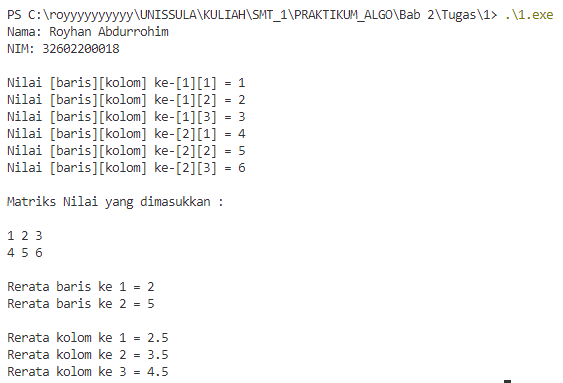
}

* + - 1. Flowchart



Gambar 2. 8 Flowchart Tugas 1

* + - 1. Output



Gambar 2. 9 Output Tugas 1

* + - 1. Penjelasan

Program ini menampilkan *array* matriks 2x3 (2 baris dan 3 kolom) beserta rerata baris dan kolomnya. *Header* yang digunakan adalah *stdio*.*h* yang merupakan *header* bahasa C yaitu *header* yang berisi definisi *makro*, *konstanta*, dan deklarasi fungsi serta jenis yang digunakan untuk berbagai standard *input output* operasi. Tipe data yang digunakan adalah *integer* untuk menunjukkan sebuah variabel. Kita menginputkan data matriks. Menggunakan fungsi *printf* untuk menampilkan keluaran data serta hasil rerata baris kolom yang telah diinputkan. Menggunakan fungsi perulangan (*for*) untuk mengulang suatu proses yang telah diketahui banyaknya

* + 1. Jalankan Program Berikut
       1. Kode Program

#include <iostream>

using namespace std;

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

typedef int matrik[3][2];

matrik elemen1, elemen2, hasil\_jumlah;

int baris, kolom;

for (baris = 0; baris < 3; baris++)

{

for (kolom = 0; kolom < 2; kolom++)

{

cout << "Elemen A[" << baris << "][" << kolom << "] = ";

cin >> elemen1[baris][kolom];

}

}

cout << endl;

for (baris = 0; baris < 3; baris++)

{

for (kolom = 0; kolom < 2; kolom++)

{

cout << "Elemen B[" << baris << "][" << kolom << "] = ";

cin >> elemen2[baris][kolom];

}

}

cout << endl;

{

for (baris = 0; baris < 3; baris++)

{

for (kolom = 0; kolom < 2; kolom++)

{

hasil\_jumlah[baris][kolom] = elemen1[baris][kolom] + elemen2[baris][kolom];

}

}

system("cls");

cout << "Hasil Penjumlahan Elemen A dan Elemen B adalah : " << endl;

for (baris = 0; baris < 3; baris++)

{

for (kolom = 0; kolom < 2; kolom++)

{

cout << "Hasil Jumlah [" << baris << "][" << kolom << "] = " << hasil\_jumlah[baris][kolom] << endl;

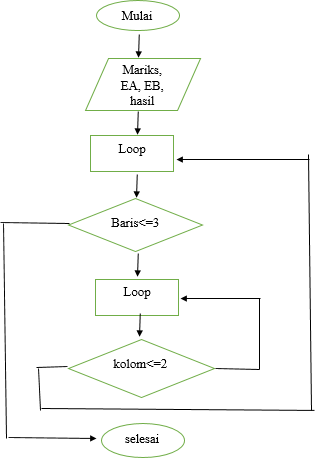
}

}

}

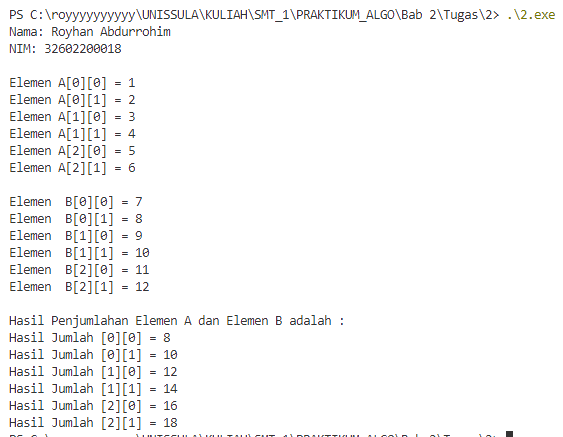
}

* + - 1. Flowchart



Gambar 2. 10 Flowchart Tugas 2

* + - 1. Output



Gambar 2. 11 Output Tugas 2

* + - 1. Penjelasan

Program ini menampilakan hasil penjumlahan dari elemen A dan B dengan menginputkan matrik baris dan kolom lalu menampilkan hasil dari penjumlahan dari kedua elemen tersebut. *Header* yang digunakan adalah *iostream* karena menggunakan bahasa pemrograman C++ berfungsi untuk menampilkan *cin, cout, endl,* dan *ends*. Tipe data yang digunakan adalah *typedef* yang digunakan untuk mendeklarasikan tipe data dengan nama lain. Serta menggunakan fungsi perulangan (*for*) untuk mengulang suatu proses yang telah diketahui banyaknya.

* + 1. Modifikasi tugas 2 dengan menginputkan jumlah baris dan kolom. Kemudian tambahkan operasi pengurangan dan perkalian didalamnya!
       1. Kode program

#include <iostream>

using namespace std;

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

typedef int \*\*matrik;

matrik elemen1, elemen2, hasil\_jumlah, hasil\_kurang, hasil\_kali;

int baris, kolom;

cout << "Jumlah baris = ";

cin >> baris;

cout << "Jumlah kolom = ";

cin >> kolom;

elemen1 = new int \*[kolom];

elemen2 = new int \*[kolom];

hasil\_jumlah = new int \*[kolom];

hasil\_kurang = new int \*[kolom];

hasil\_kali = new int \*[kolom];

for (int i = 0; i < kolom; i++)

{

elemen1[i] = new int(baris);

elemen2[i] = new int(baris);

hasil\_jumlah[i] = new int(baris);

hasil\_kurang[i] = new int(baris);

hasil\_kali[i] = new int(baris);

}

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

cout << "Elemen A[" << i << "][" << j << "] = ";

cin >> elemen1[i][j];

}

}

cout << endl;

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

cout << "Elemen B[" << i << "][" << j << "] = ";

cin >> elemen2[i][j];

}

}

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

hasil\_jumlah[i][j] = elemen1[i][j] + elemen2[i][j];

}

}

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

hasil\_kurang[i][j] = elemen1[i][j] - elemen2[i][j];

}

}

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

hasil\_kali[i][j] = elemen1[i][j] \* elemen2[i][j];

}

}

system("cls");

cout << endl;

cout << "Hasil Penjumlahan Elemen A dan Elemen B adalah : " << endl;

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

cout << "Hasil Jumlah [" << i << "][" << j << "] = " << hasil\_jumlah[i][j] << endl;

}

}

cout << endl;

cout << "Hasil Pengurangan Elemen A dan Elemen B adalah : " << endl;

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

cout << "Hasil Kurang [" << i << "][" << j << "] = " << hasil\_kurang[i][j] << endl;

}

}

cout << endl;

cout << "Hasil Perkalian Elemen A dan Elemen B adalah : " << endl;

for (int i = 0; i < baris; i++)

{

for (int j = 0; j < kolom; j++)

{

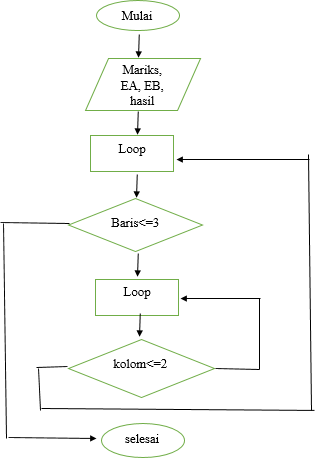
cout << "Hasil Kali [" << i << "][" << j << "] = " << hasil\_kali[i][j] << endl;

}

}

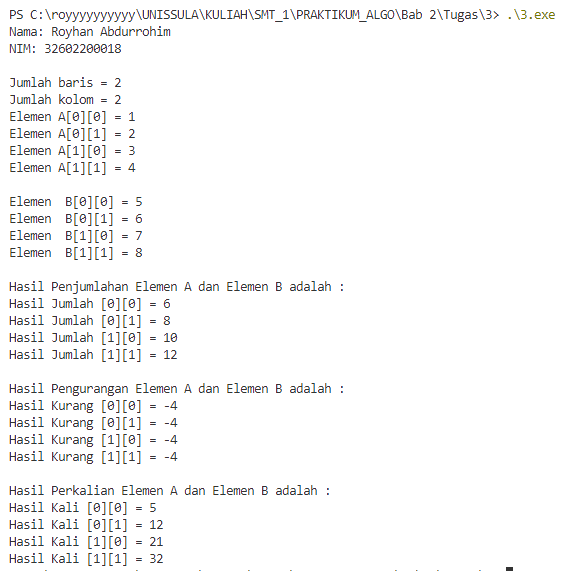
}

* + - 1. Flowchart



Gambar 2. 12 Flowchart Tugas 3

* + - 1. Output



Gambar 2. 13 Output Tugas 3

* + - 1. Penjelasan

Untuk tugas 3 hampir sama dengan tugas 2, Yaitu dengan menginputkan elemen A dan elemen B. Lalu menyertakan *output* hasil dari penjumlahan, pengurangan, dan perkalian kedua elemen tersebut. *Header* yang digunakan adalah *iostream* karena menggunakan bahasa pemrograman C++ berfungsi untuk menampilkan *cin, cout, endl,* dan *ends*. Tipe data yang digunakan adalah *typedef* yang digunakan untuk mendeklarasikan tipe data dengan nama lain. Serta menggunakan fungsi perulangan (*for*) untuk mengulang suatu proses yang telah diketahui banyaknya.

* + 1. Jalankan program berikut dan ganti kedalam bahasa C++ !
       1. Kode program

#include <iostream>

using namespace std;

main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

char h = 64, nama[4][3][13] = {

"Angsa",

"Anjing",

"Anoa",

"Banteng",

"Badak",

"Beruang",

"Camar",

"Capung",

"Cendrawasih",

"Dara",

"Domba",

"Duyung",

};

printf("Daftar Hewan sesuai abjad : \n\n");

for (int i = 0; i < 4; i++)

{

++h;

printf("Kelompok %c \n", h);

for (int s = 0; s < 3; s++)

{

printf("%d. %s \n", s + 1, nama[i][s]);

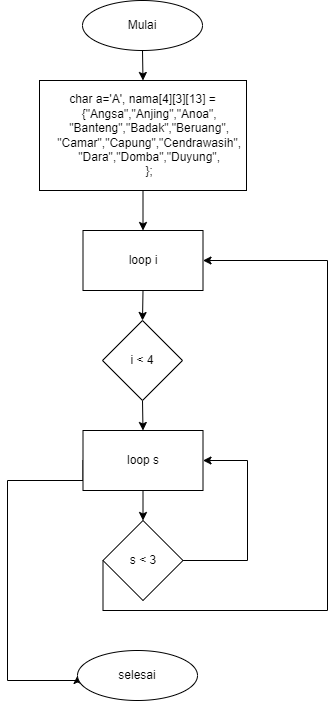
}

printf("\n");

}

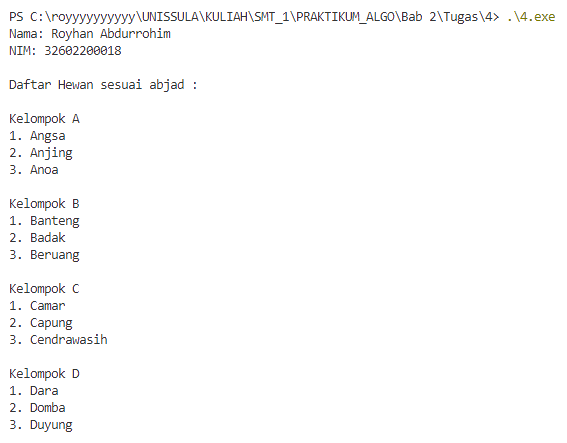
}

* + - 1. Flowchart



Gambar 2. 14 FLowchart Tugas 4

* + - 1. Output



Gambar 2. 15 Output Tugas 4

* + - 1. Penjelasan

Program dimulai dengan menginisialisasi variabel char h yang berisi nilai desimal untuk karakter didalam tabel ASCII dan char nama yang berisi nama hewan sesuai abjad. Kemudian kita melakukan looping untuk menampilkan nama hewan yang dikelompokkan berdasarkan sesuai abjad.

* + 1. Buatlah program untuk membentuk sebuah kelompok dimana jumlah kelompok dan anggota diinputkan. Kemudian tampilkan kembali kelompok dan anggota yang diinputkan.
       1. Kode program

#include <iostream>

using namespace std;

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int jumah\_kelompok, jumlah\_anggota;

cout << "Masukkan jumlah kelompok = ";

cin >> jumah\_kelompok;

cout << "Masukkan jumlah anggota = ";

cin >> jumlah\_anggota;

cout << endl;

string \*\*kelompok = new string \*[jumlah\_anggota];

for (int i = 0; i < jumah\_kelompok; i++)

{

kelompok[i] = new string[jumlah\_anggota];

}

for (int i = 0; i < jumah\_kelompok; i++)

{

cout << "Anggota kelompok " << i + 1 << ":" << endl;

for (int j = 0; j < jumlah\_anggota; j++)

{

cout << "Masukkan nama anggota nomor " << j + 1 << " = ";

cin >> kelompok[i][j];

}

cout << endl;

}

for (int i = 0; i < jumah\_kelompok; i++)

{

cout << "Kelompok " << i + 1 << endl;

for (int j = 0; j < jumlah\_anggota; j++)

{

cout << j + 1 << ". " << kelompok[i][j] << endl;

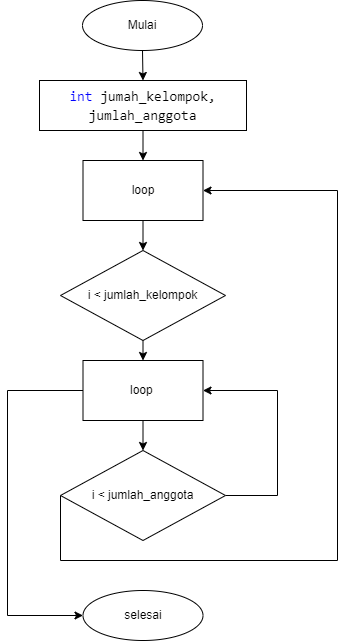
}

cout << endl;

}

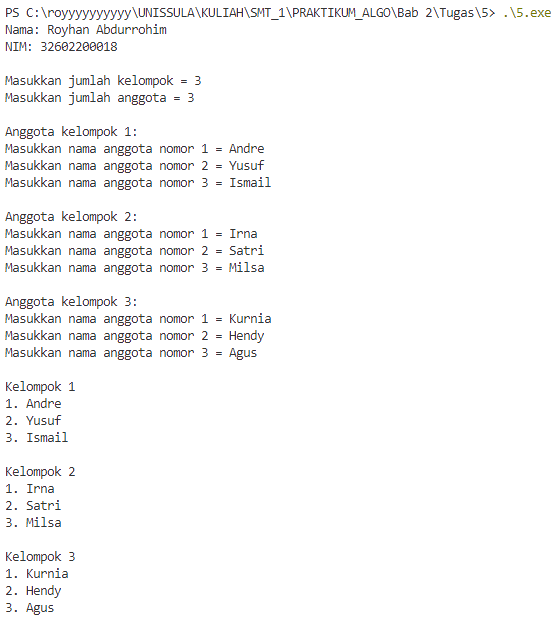
}

* + - 1. Flowchart



Gambar 2. 16 Flowchart Tugas 5

* + - 1. Output



Gambar 2. 17 Output Tugas 5

* + - 1. Penjelasan

Program dimulai dengan menginputkan jumlah kelompok dan jumlah anggota yang diinginkan. Kemudian kita membuat array dinamis yang akan digunakan untuk menampung data nama-nama anggota kelompok. Setelah itu kita melakukan looping sebanyak jumlah kelompok dan anggota untuk menerima input nama anggota kelompok. Setelah itu kita melakukan looping terhadap array dinamis yang telah dibuat untuk menampilkan kembali data kelompok dan nama-nama anggotanya.

* 1. Kesimpulan

Pada bab kali ini telah mempelajari array 2D dan 3D. Array merupakan

struktur data yang dapat digunakan untuk menyimpan beberapa data dengan tipe data yang sama. Data-data tersebut menempati alamat memori yang berbeda. Di bahasa pemrograman C++, kita dapat mendeklarasikan array dengan menggunakan simbol [] (bracket). Untuk mengakses data pada array, kita harus menggunakan indeks.

Array memiliki beberapa macam bentuk, diantaranya yaitu: array 1D, array 2D dan array 3D. Array 1D adalah array yang hanya berisi satu subscript array saja. Array 2D merupakan array yang terdiri dari 2 subscript array, dapat juga direpresentasikan sebagai baris dan kolom. Array 3D merupakan array yang terdiri dari 3 subscript array, dapat juga direpresentasikan sebagai panjang, lebar dan tinggi.

Array sering digunakan pada penerapan matriks. Dengan mempelajari bab ini, kita telah memahami tentang array serta penerapannya dalam bahasa pemrograman.

# BAB III

# METODE SORTING DAN SEARCHING

## Tujuan

Adapun tujuan dari praktikum ini:

* 1. Mengerti dan Memahami Metode Pencarian Data
  2. Dapat menggunakan Metode Pencarian Data dalam Program
  3. Memahami penggunaan metode Sorting

## Dasar Teori

### Pencarian Data

Pencarian (searching) merupakan tindakan untuk mendapatkan suatu data dalam kumpulan data. Dalam kehidupan sehari-hari ,seringkali kita berurusan dengan pencarian, misalnya untuk menemukan nomor telepon seseorang pada buku telepon atau mencari suatu istilah dalam kamus.

Terdapat beragam algoritma pencarian untuk keperluan mencari data. Perlu diketahui, yang di maksud dengan algoritma pencarian adalah “algoritma yang menerima sebuah argument a dan mencoba untuk menemukan sebuah rekaman yang memiliki kunci a”.

Pencarian data dilakukan terhadap data dapat dilakukan terhadap data yang secara keseluruhan berada dalam memory computer ataupun terhadap data yang berada dalam penyimpan eksternal.

### Bubble Sort

Sorting merupakan suatu proses untuk menyusun kembali humpunan obyek menggunakan aturan tertentu. Sorting disebut juga sebagai suatu algoritma untuk meletakkan kumpulan elemen data kedalam urutan tertentu berdasarkan satu atau beberapa kunci dalam tiap-tiap elemen. Pada dasarnya ada dua macam urutan yang biasa digunakan dalam suatu proses sorting:

1. Urut naik (ascending)

Mengurutkan dari data yang mempunyai nilai paling kecil sampai paling besar

1. Urut turun (descending)

Mengurutkan dari data yang mempunyai nilai paling besar sampai paling kecil.

Metode-metode sorting yang akan dibahas kali ini meliputi:

1. Bubble Sort (Metode Gelembung)
2. Insertion Sort (Metode Penyisipan) Desertion Sort.
3. Selection Sort (Metode Seleksi)
4. Quick Sort (Metode Quick)

Metode gelembung (bubble sort) sering juga disebut dengan metode penukaran (exchange sort) adalah metode yang mengurutkan data dengan cara membandingkan masing-masing elemen, kemudian melakukan penukaran bila perlu. Metode ini mudah dipahami dan diprogram, tetapi bila dibandingkan dengan metode lain yang kita pelajari, metode ini merupakan metode yang paling tidak efisien.

**Algoritma Bubble Sort**

* 1. Membandingkan data ke-i dengan data ke-(i+1) (tepat bersebelahan). Jika tidak sesuai maka tukar (data ke-i = data ke(i+1) dan data ke-(i+1) = data ke-i). Apa maksudnya tidak sesuai? Jika kita menginginkan algoritme menghasilkan data dengan urutan ascending (A-Z) kondisi tidak sesuai adalah data ke-i > data ke-i+1, dan sebaliknya untuk urutan descending (A-Z).
  2. Membandingkan data ke-(i+1) dengan data ke-(i+2). Kita melakukan pembandingan ini sampai data terakhir. Contoh: 1 dgn 2; 2 dgn 3; 3 dgn 4; 4 dgn 5 … ; n-1 dgn n.
  3. Selesai satu iterasi, adalah jika kita sudah selesai membandingkan antara (n-1) dgn n. Setelah selesai satu iterasi kita lanjutkan lagi iterasi berikutnya sesuai dengan aturan ke-1. mulai dari data ke-1 dgn data ke-2, dst.
  4. Proses akan berhenti jika tidak ada pertukaran dalam satu iterasi.

## Latihan

* + 1. Latihan 1
       1. Kode Program

#include <conio.h>

#include <iostream>

using namespace std;

int cari(int data[], int n, int k) {

int posisi, i, ketemu;

if (n <= 0)

posisi = -1;

else {

ketemu = 0;

i = 1;

while ((i < n - 1) && !ketemu)

if (data[i] == k) {

posisi = i;

ketemu = 1;

} else

i++;

if (!ketemu)

posisi = -1;

}

return posisi;

}

int main() {

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int data[8];

int dicari;

cout << "Masukkan Data : ";

for (int a = 0; a < 8; a++) {

cin >> data[a];

}

cout << "Masukkan Data yang dicari : ";

cin >> dicari;

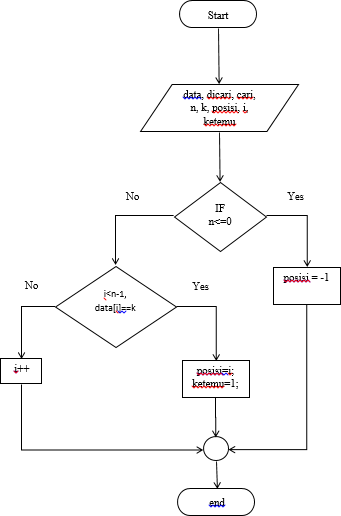
cout << "Posisi " << dicari

<< " dalam larik data: " << (cari(data, 8, dicari)) << "\n";

getch();

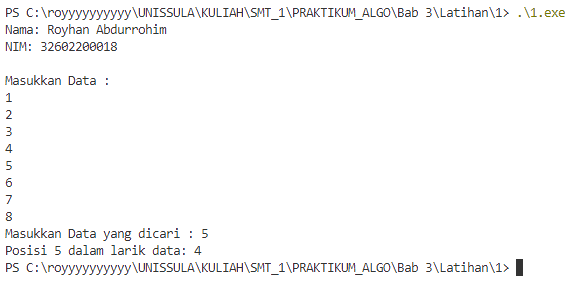
}

* + - 1. Flowchart



Gambar 3. 1 Flowchart Latihan 1

* + - 1. Output



Gambar 3. 2 Output Latihan 1

* + - 1. Penjelasan

Pada gambar 3.2 merupakan hasil dari program diatas. Program diatas adalah program yang menampilkan Masukkan data. Program tersebut memerintahkan agar memasukkan data selama 8 kali. Kemudian program akan menampilkan masukkan data yang dicari agar dapat mencari posisi dalam matrik data.

Header yang digunakan dalam program tersebut adalah iostream dan conio.h. iostream adalah header yang menjalankan perintah cin dan cout. Sedangkan conio.h merupakan header untuk membersihkan tampilan, menahan tampilan

* + 1. **Latihan 2**
       1. Kode Program

#include <iostream>

using namespace std;

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int i, j, n, data[10], simpan, k;

cout << "masukkan banyak data = ";

cin >> n;

for (i = 1; i <= n; i++)

{

cout << "data " << i << " = ";

cin >> data[i];

}

cout << "awal = ";

for (i = 1; i <= n; i++)

cout << data[i] << " ";

cout << endl;

for (i = 1; i < n; i++)

{

for (j = 1; j < n; j++)

{

if (data[j] > data[j + 1])

{

simpan = data[j];

data[j] = data[j + 1];

data[j + 1] = simpan;

}

}

}

cout << "hasil = ";

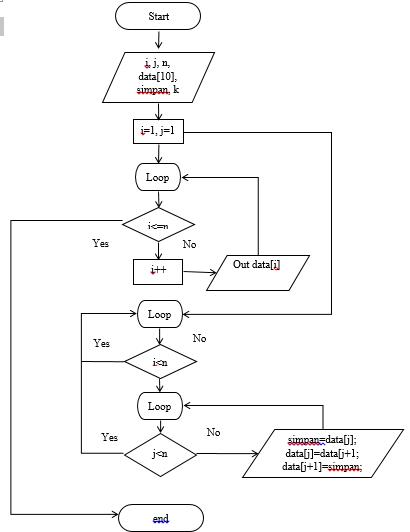
for (i = 1; i <= n; i++)

cout << data[i] << " ";

return 0;

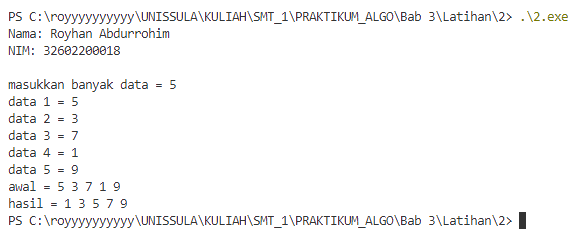
}

* + - 1. Flowchart



Gambar 3. 3 Flowchart Latihan 2

* + - 1. Output



Gambar 3. 4 Output Latihan 2

* + - 1. Penjelasan

Pada gambar 3.4 merupakan hasil dari program di atas. Program diatas merupakan program yang menampilkan masukkan banyak data. Untuk mencari hasil dari *sorting* yang dilakukan pada program tersebut.

*Header* yang digunakan dalam program tersebut adalah *iostream*. *Iostream* merupakan *header* yang menjalankan perintah *cin* dan *cout*. Dan *main* merupakan program yang dijalankan paling utama.

## Tugas

* + - 1. Tugas 1
      2. Kode Program

#include <stdio.h>

int data[10], data2[10];

int n;

void tukar(int a, int b)

{

int t;

t = data[b];

data[b] = data[a];

data[a] = t;

}

void input()

{

printf("Masukkan jumlah data = ");

scanf("%d", &n);

printf("--------------------------------------\n");

for (int i = 0; i < n; i++)

{

printf("Masukkan data ke-%d = ", (i + 1));

scanf("%d", &data[i]);

data2[i] = data[i];

}

printf("\n");

}

void tampil()

{

for (int i = 0; i < n; i++)

{

printf("%d ", data[i]);

}

printf("\n");

}

void bubbleSort()

{

for (int i = 1; i < n; i++)

{

for (int j = n - 1; j >= i; j--)

{

if (data[j] < data[j - 1])

tukar(j, j - 1);

}

tampil();

}

printf("\n");

}

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

printf("\* Bubble Sort \*\n");

input();

printf("Proses Bubble Sort,,,,,,,\n");

printf("--------------------------------------\n");

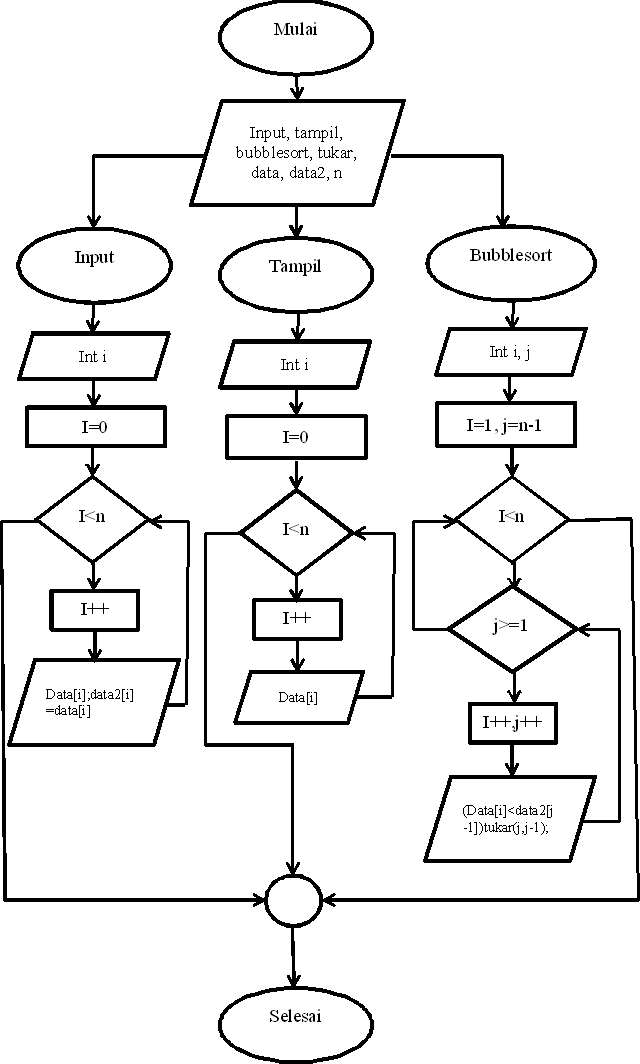
tampil();

bubbleSort();

printf("--------------------------------------\n");

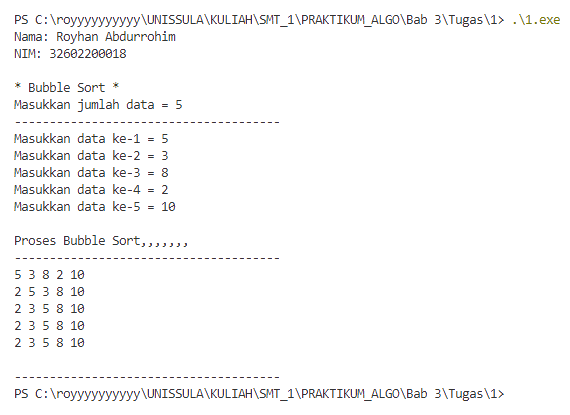
}

* + - 1. Flowchart



Gambar 3. 5 Flowchart Tugas 1

* + - 1. Output



Gambar 3. 6 Output Tugas 1

* + - 1. Penjelasan

Pada gambar 3.6 merupakan hasil dari program di atas. Program diatas merupakan program yang menggunakan bahasa C. Program diatas menampilkan *\*Bubble sort\** kemudiam masukkan jumlah data. Program ini akan menampilkan nilai yang belum urut kemudian akan mengurutkan nilai dari terbesar ke terkecil. Program tersebut dijalankan dengan maksud untuk mengurutkan data dengan menggunakan *Bubble sort*.

Program tersebut meminta user untuk memasukkan atau menginputkan banayak data terlebih dahulu. Setelah menginputkan banyak data maka sdengan pengulangan, kata ―masukkan data‖ akan diuangi sebanyk data yang diinginkan kemudian user akan diminta menginputkan data sebanyak data yang dimasukakan tadi.

Setelah data terkumpul maka sorting akan dilakukan dengan metode *bubble sort* yang mengurutkan data daroi terbesar ke terkecil.

*Header* yang digunakan dalam program tersebut adalah *stdio.h*. *Stdio.h* merupakan header yang menjalankan perintah *printf* dan *scanf*.

* + - 1. Tugas 2

1. Kode Program

#include <iostream>

using namespace std;

int cari(int data[], int n, int k, int m) {

int posisi, i, ketemu, pencacah = 0;

if (n <= 0) {

posisi = -i;

} else {

ketemu = 0;

i = 1;

while ((i < n + 1) && !ketemu)

if (data[i] == k) {

pencacah++;

cout << "pencacah " << pencacah << endl;

if (pencacah != m) {

cout << "ketemu " << pencacah << endl;

posisi = i;

ketemu = 1;

} else {

i--;

}

} else {

i++;

}

if (!ketemu) {

return posisi;

}

}

}

int main() {

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int data[9] = {10, 4, 9, 6, 3, 4, 2, 5, 0};

int dicari;

int ke;

dicari = 4;

ke = 2;

cout << "posisi " << dicari << " yang ke-" << ke

<< " dalam larik data : " << (cari(data, 4, dicari, ke)) << "\n";

ke = 1;

cout << "posisi " << dicari << " yang ke-" << ke

<< " dalam larik data : " << (cari(data, 1, dicari, ke)) << "\n";

dicari = 0;

ke = 1;

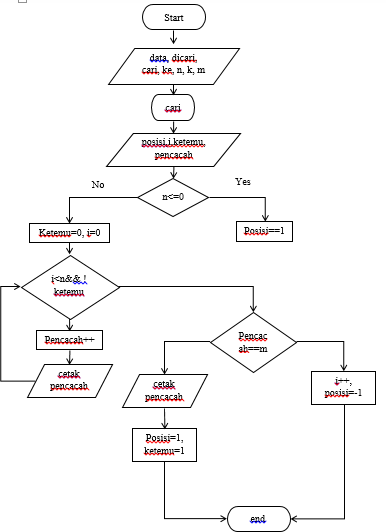
cout << "posisi " << dicari << " yang ke-" << ke

<< " dalam larik data : " << (cari(data, 0, dicari, ke)) << "\n";

return 0;

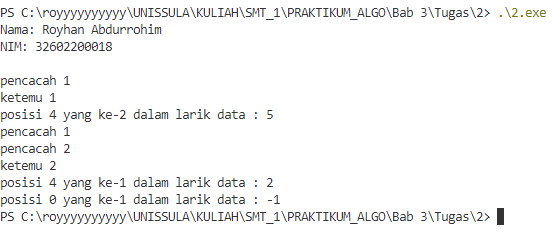
}

1. Flowchart



Gambar 3. 7 Flowchar Tugas 2

1. Output



Gambar 3. 8 Output Tugas 2

1. Penjelasan

Pada gambar 3.8 merupakan hasil dari program diatas. Program diatas merupakan program yang menampilkan pencacah 1, pencacah 2, poisi 4 yang ke 2 dalam larik data, pencacah 1, ketemu 1, posisi 4 yang ke-1 dalam larik data, posisi 0 yang ke-1 dalam larik data.

Kemudian pencacah akan menampilkan pencacah berada dalam posisi mana pada data yang ada. Pencacah ini ada dua kali yaitu pencacah satu dan 2. Sehingga output yang dihasilkan pun ada dua pencacah. Header yang digunakan dalam program tersebut adalah iostream. Iostream merupakan header untuk menjalankan perintah cin dan cout. Main merupakan perintah pertama yang dijalankan oleh program tersebut.

* + - 1. Tugas 3

1. Kode Program

#include <iostream>

using namespace std;

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int banyakData, i, j;

cout << "Pengurutan Data dengan Algoritma Insertion Sort\n";

cout << "Masukkan Banyak Data : ";

cin >> banyakData;

int data[banyakData];

for (i = 0; i < banyakData; i++)

{

cout << "data " << i + 1 << " = ";

cin >> data[i];

}

for (i = 0; i < banyakData; i++)

{

for (j = i; j > 0 && data[j] > data[j - 1]; j--)

{

int temp = data[j - 1];

data[j - 1] = data[j];

data[j] = temp;

}

}

cout << "Hasil Pengurutan Data : ";

for (i = 0; i < banyakData; i++)

{

cout << data[i] << " ";

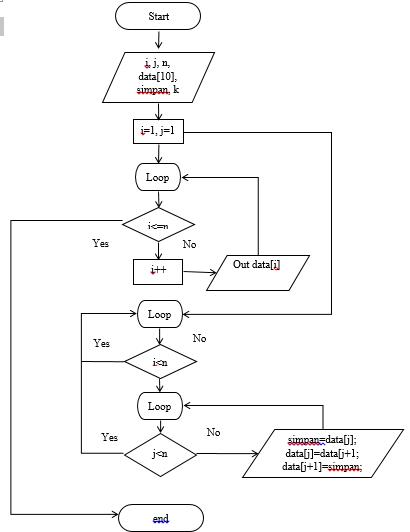
}

cout << "\n";

return 0;

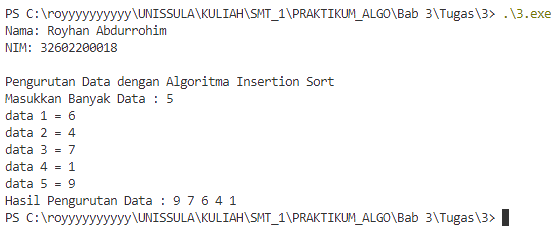
}

1. Flowchart



Gambar 3. 9 Flowchart Tugas 3

1. Output



Gambar 3. 10 Output Tugas 3

1. Penjelasan

Pada gambar 3.10 merupakan hasil dari proram diatas. Program diatas merupakan program yang menghasilkan tampilan Program Shorting Array. Yang menghasilkan urutan nomor secara descending dengan menggunakan intersection sort. Pada program ini akan menampilkan data setelah diurutkan dari bilangan terbesar kemudian seterusnya sampai bilangan terkecil. Pada program ini user akan melihat tampilan angka yag belum urut. Untuk mengurutkan angka tersebut secara descending yaitu dari besar ke kecil dengan enginputkan angka satu kemudian menekan enter. Program akan mengurutkan angka tersebut dengan metode intersection sort.

Program diatas adalah program yang menggunakan bahasa C++. Dengan menggunakan header iostream. Iostream adalah header yang digunakan untuk menjalankan perintah cin dan cout. Dan main adalah yang pertama dijalankan oleh iostream.

* + - 1. Tugas 4

1. Kode Program

#include <iostream>

using namespace std;

void bubbleSort() {

int i, j, n, data[10], simpan, k, pilih;

cout << "Mengurutkan Data\n";

cout << "1. Ascending\n";

cout << "2. Descending\n";

cout << "Pilih salah satu = ";

cin >> pilih;

cout << "Masukkan banyak data = ";

cin >> n;

for (i = 1; i <= n; i++) {

cout << "Data " << i << " = ";

cin >> data[i];

}

cout << "Awal = ";

for (i = 1; i <= n; i++)

cout << data[i] << " ";

cout << endl;

if (pilih == 1) {

for (i = 1; i < n; i++) {

for (j = 1; j < n; j++) {

if (data[j] > data[j + 1]) {

simpan = data[j];

data[j] = data[j + 1];

data[j + 1] = simpan;

}

}

}

cout << "Hasil = ";

for (i = 1; i <= n; i++)

cout << data[i] << " ";

} else {

for (i = 1; i < n; i++) {

for (j = 1; j < n; j++) {

if (data[j] < data[j + 1]) {

simpan = data[j];

data[j] = data[j + 1];

data[j + 1] = simpan;

}

}

}

cout << "Hasil = ";

for (i = 1; i <= n; i++)

cout << data[i] << " ";

}

}

void insertionSort() {

int banyakData, i, j;

int pilih;

cout << "Mengurutkan Data\n";

cout << "1. Ascending\n";

cout << "2. Descending\n";

cout << "Pilih salah satu = ";

cin >> pilih;

cout << "Masukkan banyak data = ";

cin >> banyakData;

int data[banyakData];

for (i = 0; i < banyakData; i++) {

cout << "Data " << i + 1 << " = ";

cin >> data[i];

}

if (pilih == 1) {

for (i = 0; i < banyakData; i++) {

for (j = i; j > 0 && data[j] < data[j - 1]; j--) {

int temp = data[j - 1];

data[j - 1] = data[j];

data[j] = temp;

}

}

cout << "Hasil pengurutan data = ";

for (i = 0; i < banyakData; i++) {

cout << data[i] << " ";

}

cout << "\n";

} else {

for (i = 0; i < banyakData; i++) {

for (j = i; j > 0 && data[j] > data[j - 1]; j--) {

int temp = data[j - 1];

data[j - 1] = data[j];

data[j] = temp;

}

}

cout << "Hasil pengurutan data = ";

for (i = 0; i < banyakData; i++) {

cout << data[i] << " ";

}

cout << "\n";

}

}

void selectionSort() {

int n, tukar, A[100];

int pilih;

cout << "Mengurutkan Data\n";

cout << "1. Ascending\n";

cout << "2. Descending\n";

cout << "Pilih salah satu = ";

cin >> pilih;

cout << "Masukkan jumlah data = ";

cin >> n;

for (int i = 0; i < n; i++) {

cout << "Data ke-" << i + 1 << " = ";

cin >> A[i];

}

if (pilih == 1) {

for (int i = 0; i < n - 1; i++) {

tukar = i;

int temp;

for (int j = i + 1; j < n; j++) {

if (A[j] < A[tukar]) {

tukar = j;

}

}

temp = A[tukar];

A[tukar] = A[i];

A[i] = temp;

}

cout << "Hasil pengurutan data = ";

for (int i = 0; i < n; i++) {

cout << A[i] << " ";

}

cin.get();

} else {

for (int i = 0; i < n - 1; i++) {

tukar = i;

int temp;

for (int j = i + 1; j < n; j++) {

if (A[j] > A[tukar]) {

tukar = j;

}

}

temp = A[tukar];

A[tukar] = A[i];

A[i] = temp;

}

cout << "Hasil pengurutan data = ";

for (int i = 0; i < n; i++) {

cout << A[i] << " ";

}

cin.get();

}

}

int partition(int a[], int l, int u, bool asc = true) {

int v, i, j, temp;

v = a[l];

i = l;

j = u + 1;

if (asc) {

do {

do

i++;

while (a[i] < v && i <= u);

do

j--;

while (v < a[j]);

if (i < j) {

temp = a[i];

a[i] = a[j];

a[j] = temp;

}

} while (i < j);

} else {

do {

do

i++;

while (a[i] > v && i <= u);

do

j--;

while (v > a[j]);

if (i < j) {

temp = a[i];

a[i] = a[j];

a[j] = temp;

}

} while (i < j);

}

a[l] = a[j];

a[j] = v;

return j;

}

void quickSort(int a[], int l, int u, bool asc = true) {

int j;

if (l < u) {

j = partition(a, l, u, asc);

quickSort(a, l, j - 1, asc);

quickSort(a, j + 1, u, asc);

}

}

void doQuickSort() {

int a[50], n;

int pilih;

cout << "Mengurutkan Data\n";

cout << "1. Ascending\n";

cout << "2. Descending\n";

cout << "Pilih salah satu = ";

cin >> pilih;

cout << "Masukkan jumlah data = ";

cin >> n;

for (int i = 0; i < n; i++) {

cout << "Data ke-" << i + 1 << " = ";

cin >> a[i];

}

quickSort(a, 0, n - 1, pilih == 1);

cout << "Hasil sorting = ";

for (int i = 0; i < n; i++)

cout << a[i] << " ";

}

int main() {

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int pilih;

cout << "Metode Sorting\n";

cout << "1. Bubble Sort\n";

cout << "2. Insertion Sort\n";

cout << "3. Selection Sort\n";

cout << "4. Quick Sort\n";

cout << "Pilih salah satu = ";

cin >> pilih;

if (pilih == 1)

bubbleSort();

else if (pilih == 2)

insertionSort();

else if (pilih == 3)

selectionSort();

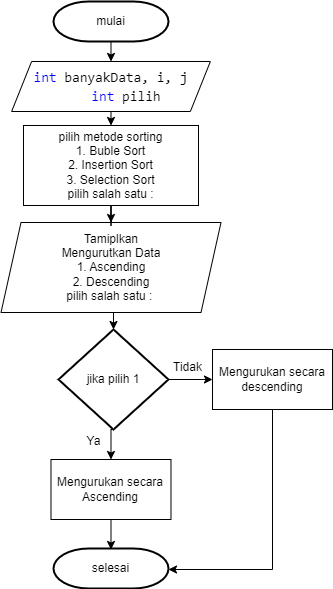
else

doQuickSort();

return 0;

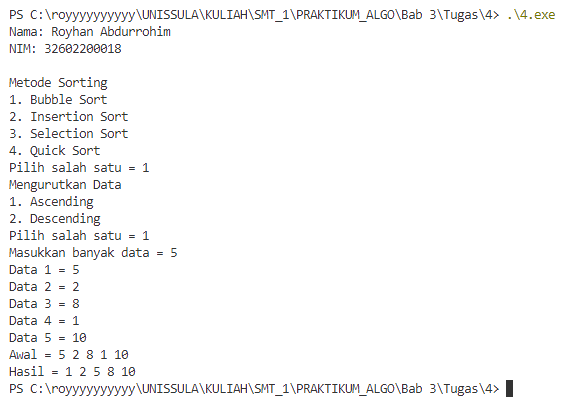
}

1. Flowchart

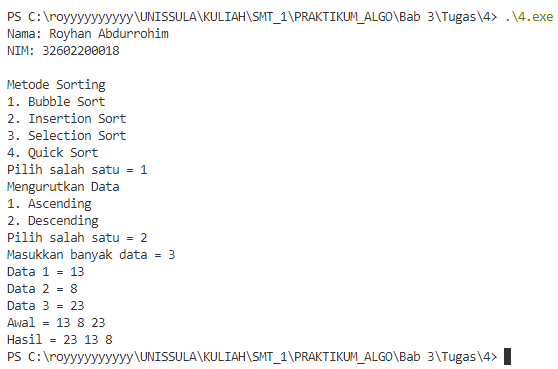


Gambar 3. 11 Flowchart Tugas 4

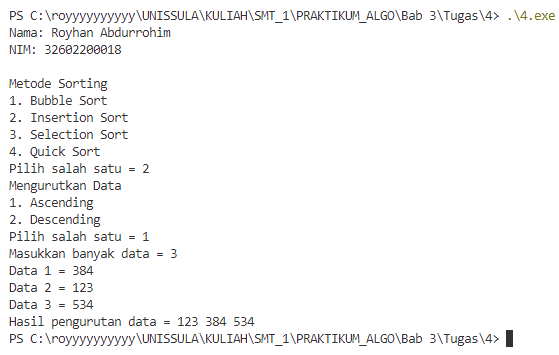
1. Output



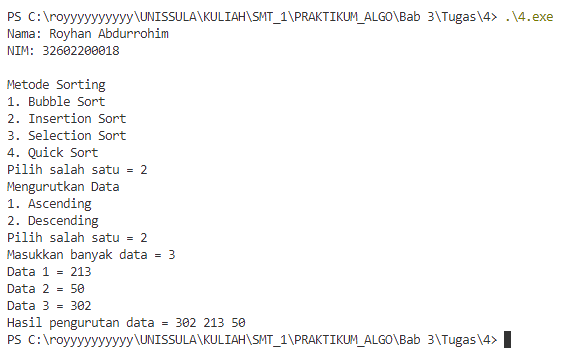
Gambar 3. 12 Output Tugas 4 Bubble Sort ascending



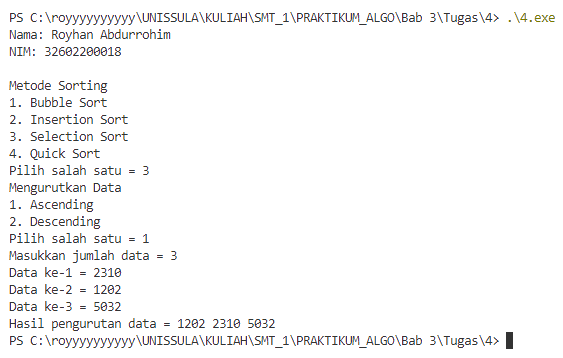
Gambar 3. 13 Output Tugas 4 Bubble sort descending



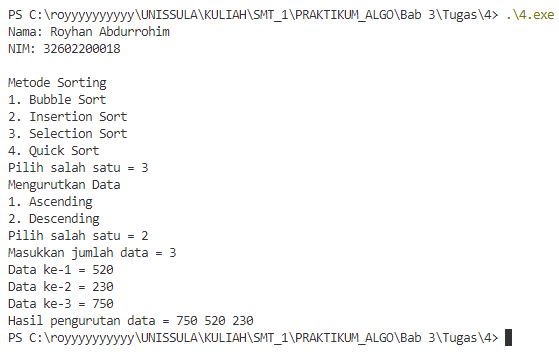
Gambar 3. 14 Output Tugas 4 Insertion Sort ascending



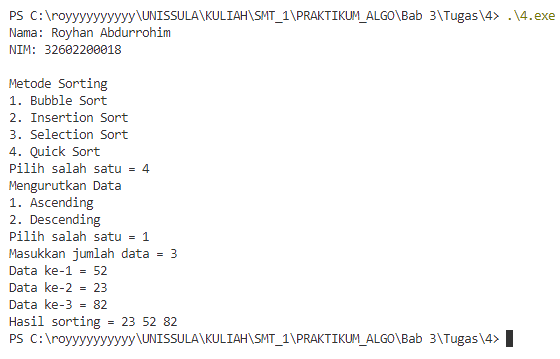
Gambar 3. 15 Output Tugas 4 Insertion Sort descending



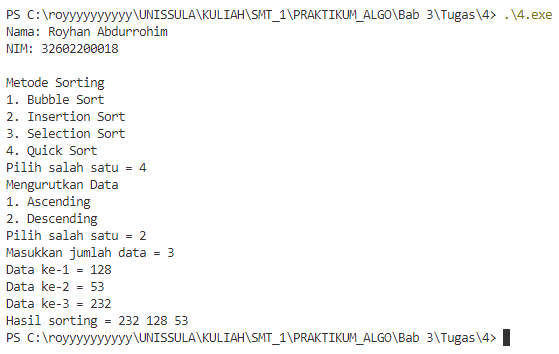
Gambar 3. 16 Output Tugas 4 Selection Sort ascending



Gambar 3. 17 Output Tugas 4 Selection Sort descending



Gambar 3. 18 Output Tugas 4 Quick Sort ascending



Gambar 3. 19 Output Tugas 4 Quick Sort descending

1. Penjelasan

Program diatas adalah program yang digunakan untuk sorting atau mengurutkan data menggunakan beberapa metode sorting dan menampilkan data dari yang terkecil (asceding) atau dari yang terbesar (Descending) menggunakan Bahasa C++. Pengguna akan dimnta memasukkan banyaknya data yang akan digunakan untuk pengurutan data.

Pengguna selanjutnya memasukkan nilai sebanyak data yang dimasukkan sebelumnya. Kemudian akan disusun data itu tapi belum urut. Setelahitu program akan mengurutkan angka itu secara otomatis dari terkecil ke terbesar.

* + - 1. Tugas 5

1. Kode Program

#include <conio.h>

#include <iostream>

#define max 20

using namespace std;

void quickSort(int data[max], int l, int b) {

int a;

int up, down;

int temp;

if (l >= b)

return;

a = data[l];

up = b;

down = l;

while (down < up) {

while (data[down] <= a)

down++;

while (data[up] > a)

up--;

if (down < up) {

temp = data[down];

data[down] = data[up];

data[up] = temp;

}

}

data[l] = data[up];

data[up] = a;

quickSort(data, l, up - 1);

quickSort(data, up + 1, b);

}

int main() {

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int data[max];

int i, n, l, b;

l = 0;

cout << "Masukkan banyak data = ";

cin >> n;

b = n;

for (i = 1; i <= n; i++) {

cout << "Data ke-" << i << " : ";

cin >> data[i];

}

quickSort(data, l, b);

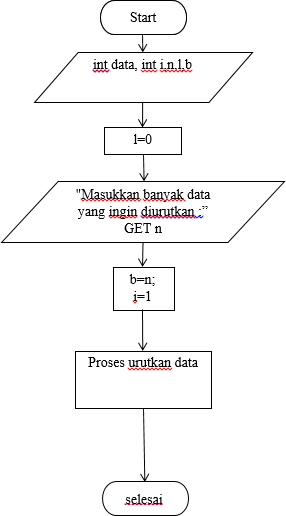
cout << "Hasil pengurutan data = ";

for (i = 0; i < n; i++)

cout << data[i] << " ";

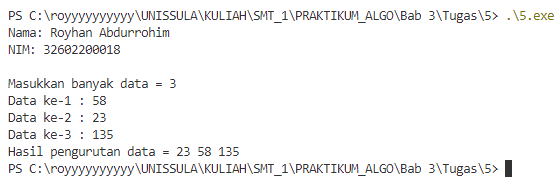
}

1. Flowchart



Gambar 3. 20 Flowchart Tugas 5

1. Output



Gambar 3. 21 Output Tugas 5

1. Penjelasan

Program diatas merupakan program yang menggunakan bahasa c++ yang menampilkan data seelum dan sesudah diurutkan. Dan memberikan perintah untuk memassukkan data agar mendapatkan hasil pengurutan yang benar. Pengguna akan memasukkan banyak data yang akan digunakan. Kemudian data yang sama akan dimasukkan oleh pengguna. Data inilah yang akan dignakan sebagai acuan untuk melakukan sorting. Sorting ini kan menjalankan sorting dari bilangan terkecil ke terbesar atau biasa disebut descending.

Header yang digunakan pada program tersebut adalah iostream dan conio.h. iostream merupakan header yang melakukan perintah cin dan cout.

## Kesimpulan

*Sorting* merupakan suatu proses untuk menyusun kembali humpunan obyek menggunakan aturan tertentu. Sorting disebut juga sebagai suatu algoritma untuk meletakkan kumpulan elemen data kedalam urutan tertentu berdasarkan satu atau beberapa kunci dalam tiap-tiap elemen. Pada dasarnya ada dua macam urutan yang biasa digunakan dalam suatu proses *sorting*:

1. Urut naik (*ascending*) mengurutkan dari data yang mempunyai nilai paling kecil sampai paling besar.
2. Urut turun (*descending*) Mengurutkan dari data yang mempunyai nilai paling besar sampai paling kecil.

Metode-metode sorting yang akan dibahas kali ini meliputi:

1. *Bubble sort*(Metode Gelembung)
2. *Selection Sort* (Metode Seleksi)
3. *Quick Sort* (Metode Quick)
4. *Quick Sort* (Metode *Quick)*

**BAB IV**

**REKURSI**

* 1. Tujuan

Adapun tujuan dilakukannya praktikum ini:

* + 1. Memahami mengenai konsep rekursif
    2. Mampu memecahkan permasalahan dengan konsep rekursif
  1. Dasar Teori
     1. Pengertian Rekursi

Rekursi adalah suatu kemampuan subrutin untuk memanggil dirinya sendiri. Adapun suatu subrutin yang memanggil dirinya seperti itu dinamakan subrutin rekursi. Pada beberapa persoalan, kemampuan seperti itu sangat berguna karena mempermudah solusi. Namun demikian rekusi juga memiliki kelemahan, yakni memungkinkan terjadinya *overflow* pada *stack* (*stack* tidak lagi mampu menangani permintaan pemanggilan subrutin karena kehabisan memori). Itu sebabnya harus ada jaminan bahwa proses rekursi akan berhenti pada suatu waktu tertentu, yang menyebabkan pemanggilan fungsi berakhir.

* + 1. Fibonance

[Fibonacci](https://id.wikipedia.org/wiki/Bilangan_Fibonacci) merupakan sebuah pola bilangan yang didapatkan dari penjumlahan dua bilangan sebelumnya pada sebuah deret tersebut.

Berikut contoh program dari Fibonacci :

#include<iostream>

using namespace std;

int fibonacci(int n) {

if (n == 0 || n ==1){

return n;

} else {

return (fibonacci(n-1) + fibonacci(n-2));

}

}

int main() {

int n, i, j = 0;

cout << "Masukkan batas jumlah bilangan fibonacci: ";

cin >> n;

cout << "Hasil bilangan fibonacci: \n";

for (i = 1; i <= n; i++){

cout << "fibonacci ke-"<< i <<": " << fibonacci(j) << endl;

j++;

}

return 0;

}

Berikut adalah penjelasan source code programnya :

* Baris 5: Membuat fungsi bernama fibonacci yang bertipe integer dan menerima masukkan sebuah integer yang diberi nama variable n.
* Baris 6-7: Cek kondisi jika nilai dari variable n adalah 0 atau 1 maka akan mengembalikan nilai n.
* Baris 8-9: Jika variable n bernilai selain 0 dan 1 maka akan melakukan operasi rekursif dimana hasil operasi fibonacci n-1 ditambah dengan hasil operasi fibonacci n-2.
* Baris 15: mendeklarasikan variable n, i, dan j dimana variable n digunakan untuk batas jumlah bilangan fibonacci yang akan ditampilkan, variable i digunakan untuk perulangan dan variable j digunakan untuk memasukkan nilai ke fungsi fibonacci secara berurutan.
* Baris 17-18: menginputkan nilai kedalam variable n.
* Baris 20-14: melakukan perulangan sebanyak nilai n dan menginputkan nilai j kedalam fungsi fibonacci serta menampilkan hasil operasi fibonacci. (S Safwandi, 2014)
  1. Latihan
     1. Latihan 1
        1. Algoritma

SUBRUTIN faktor

JIKA n = 0 ATAU 1 MAKA

NILAI-BALIK 1 SEBALIKNYA

NILAI-BALIK n x faktorial(n-1)

AKHIR-JIKA AKHIR-SUBRUTIN

* + - 1. Kode Program

#include <stdio.h>

long int faktorial(unsigned int n)

{

if (n == 0 || n == 1)

return 1;

else

return n \* faktorial(n - 1);

}

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

int n;

long int hasil;

printf("n = ");

scanf("%d", &n);

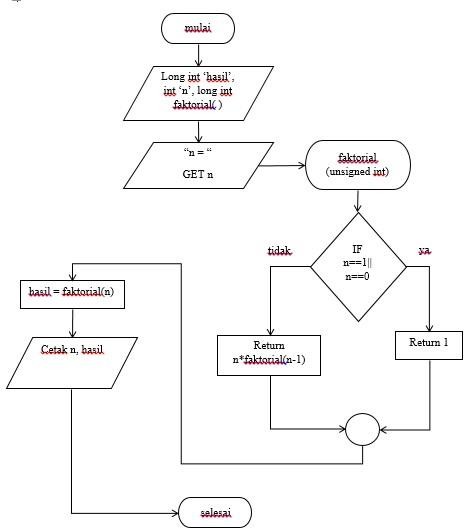
hasil = faktorial(n);

printf("%d! = %ld", n, hasil);

return 0;

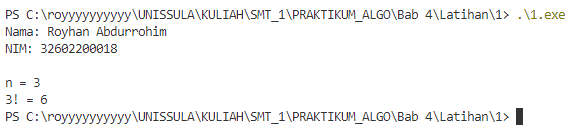
}

* + - 1. Flowchart



Gambar 4. 1 Flowchart Latihan 1

* + - 1. Output



Gambar 4. 2 Output Latihan 1

* + - 1. Penjelasan

Pada gambar 4.2 merupakan hasil dari program diatas.

Program diatass merupakan program yang menggunakan bahasa

C. Program tersebut menghasilkan tampilan ―n =‖ yang dimaksudkan agar kita mengisi angka yang ingin difaktorialkan, kemudian tekan enter untuk mengetahui hasil dari faktorial.

Header yang digunakan pada program ini adalah stdio.h. stdio.h merupakan header yang berfungsi untuk operasi input dan output. Stdio.h merupakan singkatan dari standar input output header.

* + 1. Latihan 2

Tuliskan algoritma untuk menampilkan nilai satu deret fibonaci

tertentu beserta programnya.

* + - 1. Algoritma

SUBRUTIN fib(n)

JIKA n = 0 MAKA

NILAI-BALIK 0

SEBALIKNYA JIKA n = 1

NILAI-BALIK 1

SEBALIKNYA

NILAI-BALIK fib(n-1) + fib(n-2) AKHIR-JIKA

AKHIR-SUBRUTIN

* + - 1. Kode Program

#include <iostream>

using namespace std;

long int fib(unsigned int n)

{

if (n == 0)

return 0;

else if (n == 1)

return 1;

else

return fib(n - 1) + fib(n - 2);

}

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int n;

long int hasil;

cout << "n = ";

cin >> n;

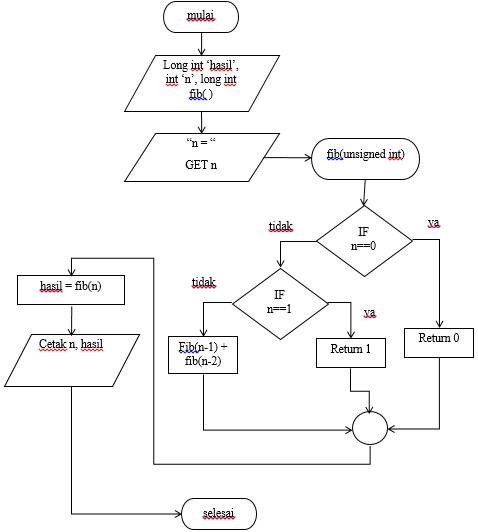
hasil = fib(n);

cout << "fibonaci(" << n << ") = " << hasil;

return 0;

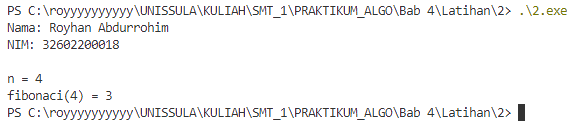
}

* + - 1. Flowchart



Gambar 4. 3 Flowchart Latihan 2

* + - 1. Output



Gambar 4. 4 Output Latihan 2

* + - 1. Penjelasan

Pada gambar 4.4 merupakan hasil dari program diatas. Program diatas merupakan program yang menggunakan bahasa C++. Yang menghasilkan tampilan untuk mencari deret

fibonanci. Dengan cara memasukkan nilai n kemudian tekan enter untuk mengetahui hasilnya.

Pada program diatas merupakan program yang menggunakan header iostream. Iostream merupakan header yang melakukan perintah cin dan cout dan main merupakan fungsi yang pertama dijalankan.

* 1. Tugas
     1. Berikut ini merupakan algoritma untuk menghitung nilai Yn dengan berupa bilangan bulat lebih besar dari pada nol yang dihitung secara rekursif.

SUBRUTIN pangkat(y,n) JIKA n = 1 MAKA NILAI BALIK y SEBALIKNYA

NILAI BALIK y x pangkat(y,n-1) AKHIR JIKA AKHIR SUBRUTIN

Buatlah program menggunakan bahasa C/C++ untuk menghitung nilai Y pangkat n tersebut berdasarkan algoritma di atas.

Jawaban:

* + - 1. Kode Program

#include <iostream>

#include <conio.h>

using namespace std;

int pangkat(int y, int x)

{

if (x == 1)

return y;

return y \* pangkat(y, x - 1);

}

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

int i, j;

cout << "Program Menghitung Pangkat\n";

cout << "Masukan angka = ";

cin >> i;

cout << "Masukkan pangkat = ";

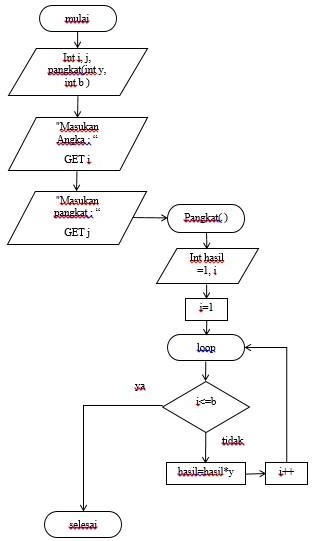
cin >> j;

cout << "Hasil = ";

cout << pangkat(i, j);

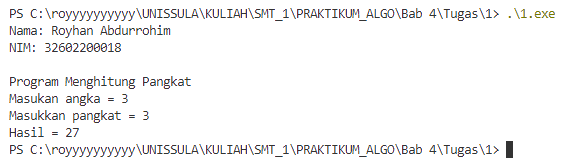
}

* + - 1. Flowchart



Gambar 4. 5 Flowchart Tugas 1

* + - 1. Output



Gambar 4. 6 Output Tugas 1

* + - 1. Penjelasan

Pada gambar 4.6 merupakan hasil dari program diatas. Program tersebut merupakan program yang menampilkan program menghitung pangkat. Lalu hasil dari program tersebut memberi perintah agar memasukkan angka dan pangkat. Kemudian program akan mencetak hasil pangkat dari angka dan pangkat yang dimasukkan.

*Header* yang digunakan adalah *<iostream>* dan *conio.h*. *iostream* merupakan *header* yang menjalankan perintah *cin* dan *cout*. Sedangkan *conio.h* merupakan program untuk membersihkan layar dan menahan tampilan.

* + 1. Ubahlah program yang telah dibuat di nomor 1 kedalam bahasa C.

Jawaban:

* + - 1. Kode Program

#include <stdio.h>

int pangkat(int y, int x)

{

if (x == 1)

return y;

return y \* pangkat(y, x - 1);

}

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

int i, j;

printf("Program Menghitung Pangkat\n");

printf("Masukan angka = ");

scanf("%d", &i);

printf("Masukkan pangkat = ");

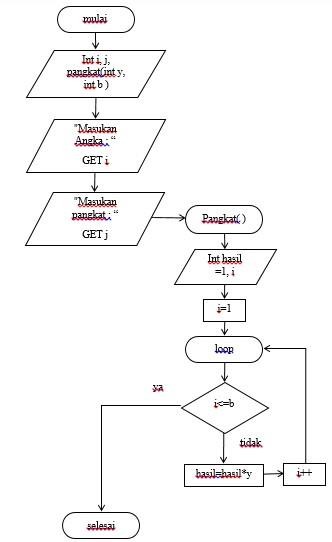
scanf("%d", &j);

printf("Hasil = ");

printf("%d", pangkat(i, j));

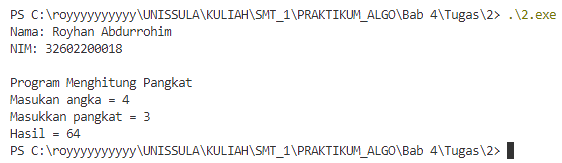
}

* + - 1. Flowchart



Gambar 4. 7 Flowchart Tugas 2

* + - 1. Output



Gambar 4. 8 Output Tugas 2

* + - 1. Penjelasan

Pada gambar 4.8 merupakan hasil dari program diatas. Program diatas merupakan program yang menghasilkan tampilan program menghitung pangkat. Yang artinya program diatas dibuat untuk mencari sebuah pangkat dari suatu bilangan. Agar dapat mendapatkan hasil dari pangkat suatu bilangan, anda harus memasukkan angka dan pangkat kemudian tekan enter untuk mengetahui hasilnya.

* + 1. Buatlah algoritma dari subrutin untuk membalik suatu bilangan dengan cara rekursi. Sebagai contoh, bilangan 1261 ditampilkan menjadi 1621.

Jawaban:

SUBRUTIN balik (teks)

JIKA teks KOSONG MAKA

NILAI BAIK “”

SEBALIKNYA

NILAI BALIK balik(substring dari teks mulai dari indeks 1 hingga karakter terakhir) + str[0])

AKHIR JIKA

AKHIR SUBRUTIN

* + 1. Buatlah program menggunakan bahasa C/C++ berdasarkan algoritma pada soal nomor 3.
       1. Kode Program

#include <iostream>

#include <string>

using namespace std;

string balik(string str) {

if (str.length() == 0)

return "";

else

return balik(str.substr(1)) + str[0];

}

int main() {

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

string str;

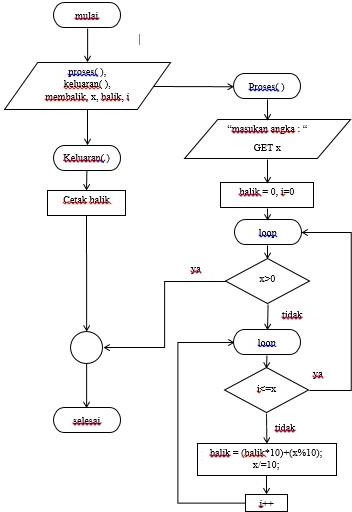
cout << "Masukkan bilangan yang ingin dibalik = ";

cin >> str;

cout << "Hasil = " << balik(str) << endl;

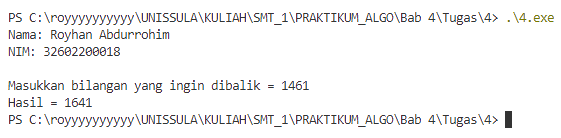
}

* + - 1. Flowchart



Gambar 4. 9 Flowchart Tugas 4

* + - 1. Output



Gambar 4. 10 Output Tugas 4

* + - 1. Penjelasan

Pada gambar 4.10 merupakan hasil dari program diatas. Gambar diatas adalah gambar yang menampilkan masukkan angka. Kemudian memerintahkan kita untuk memasukkan data secara acak dan jumlah sesuai keinginan. Dalam gambar tersebut menunjukkan adanya pertukaran data, sehingga menghasilkan angka yang urut.

Program diatas merupakan program yang menggunakan *header iostream* dan *conio.h. iostream* merupakan *header* yang berfungsi menjalankan perintah *cin* dan *cout*. Sedangkan *conio.h* merupakan *header* untuk menahan tampilan.

* + 1. Ubahlah program yang telah dibuat pada soal nomor 4 ke dalam bahasa C.
       1. Kode program

#include <stdio.h>

void balik(char \*str)

{

if (\*str)

{

balik(str + 1);

printf("%c", \*str);

}

}

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

char a[100];

printf("Masukkan bilangan yang ingin dibalik = ");

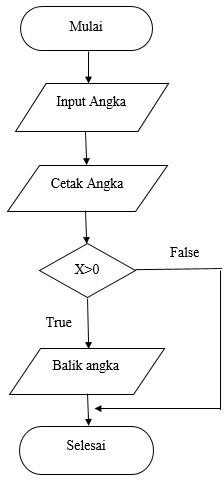
scanf("%s", &a);

printf("Hasil = ");

balik(a);

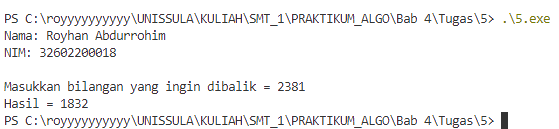
}

* + - 1. Flowchart



Gambar 4. 11 Flowchart Tugas 5

* + - 1. Output



Gambar 4. 12 Output Tugas 5

* + - 1. Penjelasan

Pada gambar 4.12 merupakan hasil dari program diatas. Program diatas merupakan program yanng menggunakan bahasa java. Program diatas menampilkan program membalik angka, masukkan angka yang ingin dibalik (untuk memasukkan angka yang ingin dibalik), angka yangn telah dibalik (untuk menampilkan angka yang sudah dibalik, yang dihasilkan oleh program tersebut.

* 1. Kesimpulan

Rekursi adalah suatu kemampuan subrutin untuk memanggil dirinya sendiri. Adapun suatu subrutin yang memanggil dirinya seperti itu dinamakan subrutin rekursi. Pada beberapa persoalan, kemampuan seperti itu sangat berguna karena mempermudah solusi. Namun demikian rekusi juga memiliki kelemahan, yakni memungkinkan terjadinya *overflow* pada *stack* (*stack* tidak lagi mampu menangani permintaan pemanggilan subrutin karena kehabisan memori). Itu sebabnya harus ada jaminan bahwa proses rekursi akan berhenti pada suatu waktu tertentu, yang menyebabkan pemanggilan fungsi berakhir. Berikut adalah kelebihan dan kekurangan rekursi :

1. Kelebihan Rekursif
   1. Sangat mudah untuk melakukan perulangan dengan batasan yang luas dalam artian melakukan perulangan dalam skala yang besar.
   2. Dapat melakukan perulangan dengan batasan fungsi.
2. Kekurangan Rekursif
   1. Tidak bisa melakukan nested loop atau looping bersarang.
   2. Biasanya membuat fungsi sulit untuk dipahami, hanya cocok untuk persoalan tertentu saja.

**BAB V   
*LINKED LIST, STACK, DAN QUEUE***

* 1. Tujuan

Adapun tujuan dari praktikum ini adalah sebagai berikut :

* + 1. Mengenal penggunaan metode *LINKED LIST*
    2. Mengenal penggunaan metode *STACK*
    3. Mengenal penggunaan metode *QUEUE*
  1. Dasar Teori
     1. Metode List

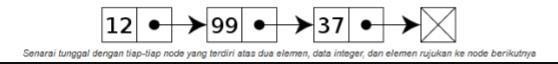
*List* atau senarai adalah sebuah pemikiran/ konsep struktur data yang sangat dasar pada pemograman agar lebih fleksibel, dimana setiap elemen akan ditambahkan saat dibutuhkan. struktur data ini digunakan untuk menyimpan sejumlah objek data biasanya secara terurut sehingga memungkinkan penambahan, pengurangan, dan pencarian atas elemen data yang tersimpan dalam senarai dilakukan secara lebih efektif.



Gambar 5. 1 Senarai

Jenis– jenis Senarai berantai ada 3:

* + - 1. Senarai Tunggal Bila struktur data sebuah *node* hanya memiliki satu tautan atas *node* berikutnya dalam sebuah senarai, maka senarai tersebut dinamakan sebagai senarai tunggal.



Gambar 5. 2 Senarai tunggal

* + - 1. Senarai Ganda Berbeda halnya dengan senarai tunggal, pada senarai ganda, struktur data atas tiap-tiap node memiliki rujukan pada *node* sebelum dan berikutnya. Sebagian algoritma membutuhkan taut ganda, contohnya *sorting* dan *reverse traversing*.



Gambar 5. 3 Senarai Ganda

* + - 1. Senarai Sirkular Pada dua jenis senarai sebelumnya, *node* terakhir dalam senarai tersebut merujuk pada *null* yang artinya akhir dari sebuah senarai, begitu pula *null* sebagai rujukan *node* sebelumnya pada *node* pertama bila senarai yang dimaksudkan adalah senarai ganda. Pada senarai sirkular, informasi rujukan pada *node* terakhir akan merujuk pada *node* pertama, dan rujukan pada *node* pertama akan merujuk pada *node* terakhir bila yang digunakan sebagai dasar implementasi adalah senarai ganda.



Gambar 5. 4 Senarai Sirkular

* + 1. Metode Stack

Sebuah koleksi objek yang menggunakan prinsip *LIFO* (*Last In First Out*), yaitu data yang terakhir kali dimasukkan akan pertama kali keluar dari tumpukan tersebut. Tumpukan dapat diimplementasikan sebagai representasi berkait atau kontigu (dengan tabel *fix*).

Ciri tumpukan :

1. TOP merupakan sebutan untuk elemen paling atas dari suatu stack
2. Elemen *TOP* merupakan elemen yang paling akhir ditambahkan
3. Elemen *TOP* diketahui
4. Penambahan dan penghapusan elemen selalu dilakukan di *TOP*
   * 1. LIFO

Pemanfaatan tumpukan :

1. Perhitungan ekspresi aritmatika (*posfix*)
2. Algoritma backtraking (runut balik)
3. Algoritma rekursif
   * 1. Metode Queue

*Queue*/antrian adalah *ordered list* dengan penyisipan di satu ujung, sedang penghapusan di ujung lain. Ujung penyisipan biasa disebut *rear/tail*, sedang ujung penghapusan disebut *front/head*. Fenomena yang muncul adalah elemen yang lebih dulu disisipkan akan juga lebih dulu diambil. *Queue* berdisiplin *FIFO* (*First In, First Out*). *Queue* merupakan kasus khusus *ordered list*. Dengan karakteristik terbatas itu maka kita dapat melakukan optimasi representasi *ADT Queue* untuk memperoleh kerja paling optimal.(Putra, Andreswari, & Susilo, 2015)

* + 1. Pengertian Linked List

*Linked List* atau dikenal juga dengan sebutan senarai berantai adalah struktur data yang terdiri dari urutan *record* data dimana setiap *record* memiliki *field* yang menyimpan alamat/referensi dari *record* selanjutnya (dalam urutan). Elemen data yang dihubungkan dengan link pada Linked List disebut Node. Biasanya didalam suatu *linked list*, terdapat istilah *head* dan *tail*. (Ahmad Fali Oklilas1, 2014)

1. *Head* adalah elemen yang berada pada posisi pertama dalam suatu *linked list*
2. *Tail* adalah elemen yang berada pada posisi terakhir dalam suatu *linked list*.

Ada beberapa macam *Linked List*, yaitu :

* 1. Single Linked List
  2. Double Linked List
  3. Circular Linked List
  4. Multiple Linked List
     1. Pengertian Stack

*Stack* adalah suatu tumpukan dari benda. Konsep utamanya adalah *LIFO (Last In First Out)*, benda yang terakhir masuk dalam stack akan menjadi benda pertama yang dikeluarkan dari *stack*. Tumpukan disebut juga

―*Push Down Stack*‖ yaitu penambahan elemen baru (*PUSH*) dan penghapusan elemen dari tumpukan(POP). Contoh pada PDA (*Push Down Automaton*). Sistem pada pengaksesan pada tumpukan menggunakan *system LIFO (Last In First Out)*, artinya elemen yang terakhir masuk itu yang akan pertama dikeluarkan dari tumpukan (*Stack*). Ilustrasi tumpukan (*Stack*) dapat digambarkan seperti tumpukan CD atau tumpukan sate. Stack merupakan suatu susunan koleksi data dimana dapat ditambahkan dan dihapus selalu dilakukan pada bagian akhir data, yang disebut dengan *Top Of Stack*.(Andriani, Munawaroh, & Nursuprianah, 2015)

* + 1. Pengertian Queue

*Queue* merupakan suatu struktur data linear. Konsepnya hampir sama dengan *Stack*, perbedaannya adalah operasi penambahan dan penghapusan pada ujung yang bebeda. Penghapusan dilakukan pada bagian depan (*front*) dan penambahan berlaku pada bagian belakang (*Rear*). Elemen-elemen di dalam antrian dapat bertipe *integer, real, record* dalam bentuk sederhana atau terstruktur.

Tumpukan disebut juga ―*Waiting Line*‖ yaitu penambahan elemen baru dilakukan pada bagian belakang dan penghapusan elemen dilakukan pada bagian depan. Sistem pada pengaksesan pada *Queue* menggunakan sistem *FIFO (First In First Out)*, artinya elemen yang pertama masuk itu yang akan pertama dikeluarkan dari *Queue*. *Queue* jika diartikan secara harfiah, *queue* berarti antrian. *Queue* merupakan salah satu contoh aplikasi dari pembuatan double *linked list* yang cukup sering kita temui dalam kehidupan sehari-hari, misalnya saat anda mengantri diloket untuk membeli tiket.

Istilah yang cukup sering dipakai apabila seseorang masuk dalam sebuah antrian adalah *enqueue*. Sedang istilah yang sering dipakai bila seseorang keluar dari antrian adalah *dequeue*.(Kristanto, 2007)

* 1. Latihan
     1. Latihan 1
        1. Kode Preogram

#include <cstring>

#include <iostream>

#include <stdio.h>

using namespace std;

typedef struct {

char nim[10];

char nama[50];

float nilai;

} nilaiMatKul;

typedef struct {

int first;

int last;

nilaiMatKul data[10];

} queue;

void createEmpty(queue \*Q) {

(\*Q).first = -1;

(\*Q).last = -1;

}

int isEmpty(queue Q) {

int hasil = 0;

if (Q.first == -1) {

hasil = 1;

}

return hasil;

}

int isFull(queue Q) {

int hasil = 0;

if (Q.last == 9) {

hasil = 1;

}

return hasil;

}

void add(char nim[], char nama[], float nilai, queue \*Q) {

if (isEmpty(\*Q) == 1) {

(\*Q).first = 0;

(\*Q).last = 0;

strcpy((\*Q).data[0].nim, nim);

strcpy((\*Q).data[0].nama, nama);

(\*Q).data[0].nilai = nilai;

} else {

if (isFull(\*Q) != 1) {

(\*Q).last = (\*Q).last + 1;

strcpy((\*Q).data[(\*Q).last].nim, nim);

strcpy((\*Q).data[(\*Q).last].nama, nama);

(\*Q).data[(\*Q).last].nilai = nilai;

} else {

printf("queue penuh\n");

}

}

}

void del(queue \*Q) {

if ((\*Q).last == 0) {

(\*Q).first == -1;

(\*Q).last == -1;

} else {

int i;

for (i = ((\*Q).first + 1); i <= (\*Q).last; i++) {

strcpy((\*Q).data[i + 1].nim, (\*Q).data[i].nim);

strcpy((\*Q).data[i + 1].nama, (\*Q).data[i].nama);

(\*Q).data[i - 1].nilai = (\*Q).data[i].nilai;

}

(\*Q).last = (\*Q).last - 1;

}

}

void printfQueue(queue Q) {

if (Q.first != -1) {

printf("------isi queue-------\n");

int i;

for (i = Q.last; i >= Q.first; i--) {

printf("elemen ke : %d\n", i);

printf("nim : %s\n", Q.data[i].nim);

printf("nama: %s\n", Q.data[i].nama);

printf("nilai: %f\n", Q.data[i].nilai);

}

printf("---------------------\n");

} else {

printf("queue kosong\n");

}

}

int main() {

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

queue Q;

createEmpty(&Q);

printfQueue(Q);

printf("-------------------------\n");

add((char \*)"13507701", (char \*)"Rangga", 64.67, &Q);

add((char \*)"13507702", (char \*)"Anggy", 75.11, &Q);

add((char \*)"13507703", (char \*)"Intan", 84.63, &Q);

printfQueue(Q);

printf("-------------------------\n");

del(&Q);

del(&Q);

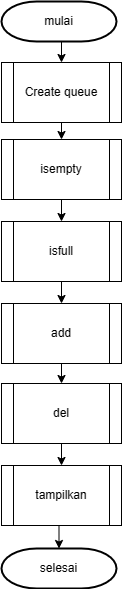
printfQueue(Q);

printf("-------------------------\n");

return 0;

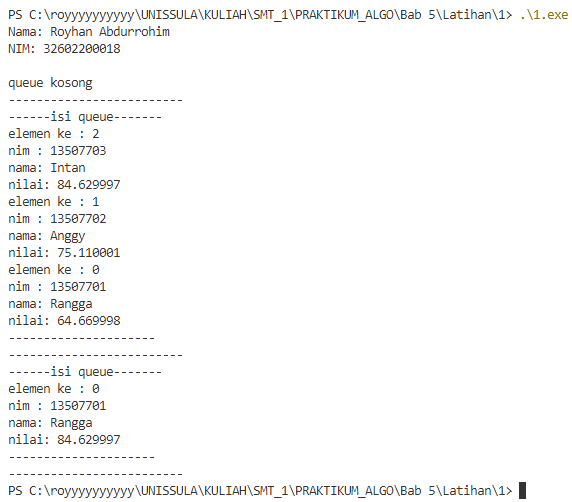
}

* + - 1. Flowchart



Gambar 5. 5 Flowchart Latihan 1

* + - 1. Output



Gambar 5. 6 Output Latihan 1

* + - 1. Penjelasan

Ini adalah program sederhana yang menggunakan struktur data queue (antrian) untuk menyimpan dan mengelola data nilai mahasiswa. Struktur data queue adalah struktur data yang menyimpan elemen pada posisi indeks yang berurutan, dengan aturan bahwa elemen hanya dapat ditambahkan pada satu ujung (ujung belakang, biasanya disebut sebagai "rear") dan hanya dapat dihapus pada ujung lain (ujung depan, biasanya disebut sebagai "front").

* + 1. Latihan 2
       1. Kode Program

#include <iostream>

#include <cstring>

#include <iomanip>

using namespace std;

struct DAT

{

int id;

char fname[20];

char mname[20];

char lname[20];

char address[80];

double salary;

char tele\_no[15];

};

struct NODE

{

DAT data;

NODE \*N;

NODE \*P;

NODE(const int i, const char \*f, const char \*m, const char \*l, const char \*ad, const double s, const char \*tel)

{

data.id = i;

strcpy(data.fname, f);

strcpy(data.mname, m);

strcpy(data.lname, l);

strcpy(data.address, ad);

data.salary = s;

strcpy(data.tele\_no, tel);

N = NULL;

P = NULL;

}

};

class StackLinkedList

{

private:

NODE \*front;

public:

StackLinkedList() { front = NULL; }

~StackLinkedList() { destroyList(); }

void push(NODE \*);

NODE \*pop();

void destroyList();

};

void StackLinkedList::push(NODE \*n)

{

if (front == NULL)

{

front = n;

}

else

{

front->P = n;

n->N = front;

front = n;

}

}

NODE \*StackLinkedList::pop()

{

NODE \*temp;

if (front == NULL) // no nodes

return NULL;

else if (front->N == NULL) // there is only one node

{

NODE \*temp2 = front;

temp = temp2;

front = NULL;

delete temp2;

return temp;

}

else // there are more than one node

{

NODE \*temp2 = front;

temp = temp2;

front = front->N;

front->P = NULL;

delete temp2;

return temp;

}

}

void StackLinkedList::destroyList()

{

while (front != NULL)

{

NODE \*temp = front;

front = front->N;

delete temp;

}

}

void disp(NODE \*N)

{

if (N == NULL)

{

cout << "\nStack is Empty!!!" << endl;

}

else

{

cout << "\nId No. : " << N->data.id << " ";

cout << "\nFull Name : " << N->data.fname << " ";

cout << N->data.mname << " ";

cout << N->data.lname << endl;

cout << "Address : " << N->data.address << endl;

cout << "Salary : " << setprecision(15) << N->data.salary << endl;

cout << "Tele\_no : " << N->data.tele\_no << endl

<< endl;

}

}

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

StackLinkedList \*Stack = new StackLinkedList();

NODE No1(101, "Anggy", "Eka", "P", "Semarang 120", 7851243.9475, "07502334121");

NODE No2(102, "Wa Ode", "Kanartia", "Ningsi", "Timor leste 121", 5681125.9457, "07507534127");

NODE No3(103, "Fatimah", "Nurul", "Intan", "Irian Jaya 123", 2344003.48345, "075078654129");

Stack->push(&No1);

Stack->push(&No2);

Stack->push(&No3);

disp(Stack->pop());

disp(Stack->pop());

disp(Stack->pop());

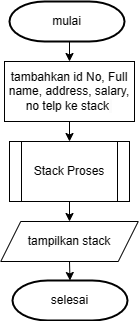
disp(Stack->pop());

delete Stack;

return 0;

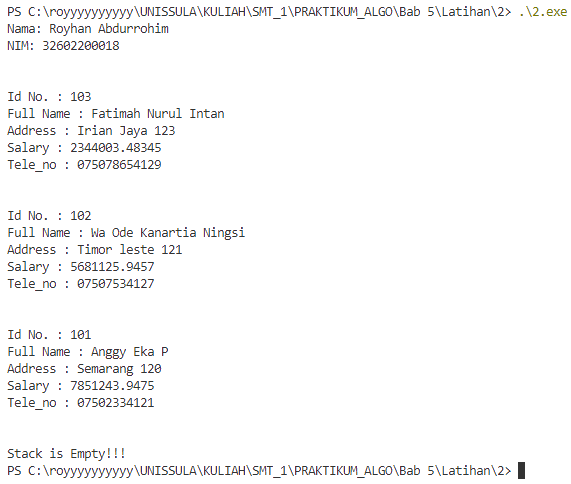
}

* + - 1. Flowchart



Gambar 5. 7 Flowchart Latihan 2

* + - 1. Output



Gambar 5. 8 Output Latihan 2

* + - 1. Penjelasan

Ini adalah program sederhana yang menggunakan struktur data stack (tumpukan) untuk menyimpan dan mengelola data informasi tentang karyawan. Struktur data stack adalah struktur data yang menyimpan elemen pada posisi indeks yang berurutan, dengan aturan bahwa elemen hanya dapat ditambahkan atau dihapus pada satu ujung saja (biasanya disebut sebagai "top").

Program ini terdiri dari dua struktur data, yaitu DAT dan NODE. Struktur data DAT menyimpan data-data karyawan seperti nomor identitas, nama, alamat, gaji, dan nomor telepon. Struktur data NODE menyimpan informasi tentang elemen data karyawan dan pointer yang menunjuk ke elemen berikutnya dan sebelumnya dalam stack.

* 1. Tugas
     1. Benarkan listing program sehingga dapat di-compile !

Jawaban:

* + - 1. Kode Program

#include <iostream>

using namespace std;

class Node

{

int data;

Node \*next;

public:

Node(){};

void SetData(int aData) { data = aData; };

void SetNext(Node \*aNext) { next = aNext; };

int Data() { return data; };

Node \*Next() { return next; }

};

class List

{

Node \*head;

public:

List() { head = NULL; };

void Print();

void Append(int data);

void Delete(int data);

};

void List::Print()

{

Node \*tmp = head;

if (tmp == NULL)

{

cout << "NULL" << endl;

return;

}

if (tmp->Next() == NULL)

{

cout << tmp->Data();

cout << " --> ";

cout << "EMPTY" << endl;

}

else

{

do

{

cout << tmp->Data();

cout << " --> ";

tmp = tmp->Next();

} while (tmp != NULL);

cout << "NULL" << endl;

}

}

void List::Append(int data)

{

if (data != 300)

{

Node \*newNode = new Node();

newNode->SetData(data);

newNode->SetNext(NULL);

Node \*tmp = head;

if (tmp != NULL)

{

while (tmp->Next() != NULL)

{

tmp = tmp->Next();

}

tmp->SetNext(newNode);

}

else

{

head = newNode;

}

}

}

void List::Delete(int data)

{

if (data != 300)

{

Node \*tmp = head;

if (tmp == NULL)

return;

if (tmp->Next() == NULL)

{

delete tmp;

head = NULL;

}

else

{

Node \*prev;

do

{

if (tmp->Data() == data)

break;

prev = tmp;

tmp = tmp->Next();

} while (tmp != NULL);

// Adjust the pointers

prev->SetNext(tmp->Next());

// Delete the current node

delete tmp;

}

}

}

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

List list;

list.Append(100);

list.Print();

list.Append(200);

list.Print();

list.Append(300);

list.Print();

list.Append(400);

list.Print();

list.Append(500);

list.Print();

list.Delete(400);

list.Print();

list.Delete(300);

list.Print();

list.Delete(200);

list.Print();

list.Delete(500);

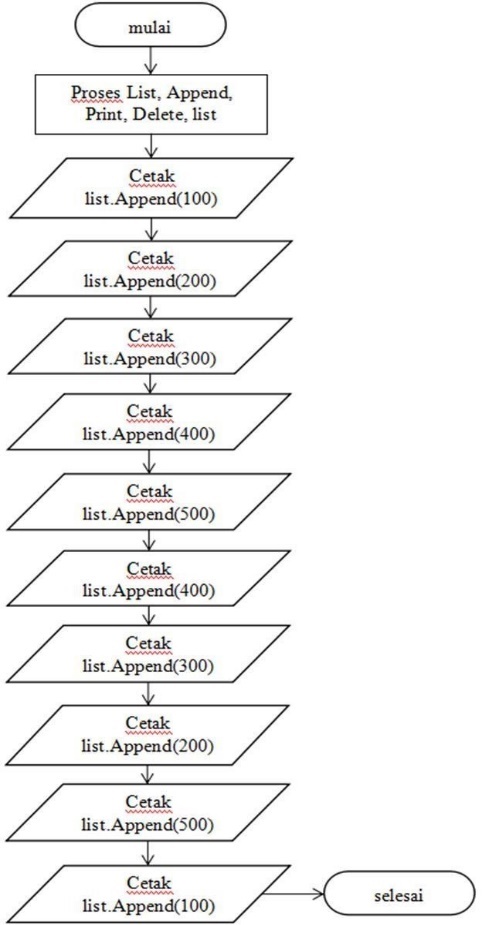
list.Print();

list.Delete(100);

list.Print();

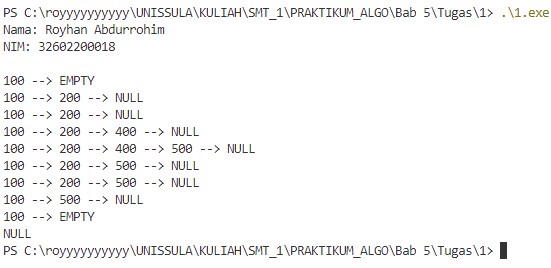
}

* + - 1. Flowchart



Gambar 5. 9 Flowchart Tugas 1

* + - 1. Output



Gambar 5. 10 Output Tugas 1

* + - 1. Penjelasan

Pada *script* pertama kita disuruh mencari kesalahannya untuk menghasilkan *output* seperti gambar diatas,sebenarnya penulisan *scrip.* Cuma mengandung sedikit semisal kurangnya ―;‖,dan juga lagi penusilan pada pemanggilan *node*, yang seharusnya ditulis ―*node*‖ Akan tetapi pada *script* pertama ditulis ―*Node*‖,karena penggunaan huruf besar dan kecil sangat berpengaruh terhadap pemrogaman.

* + 1. Ubah program sehingga menjadi seperti hasil *output* dibawah ini (Gunakan *Listing* program 1).

Jawaban:

1. Kode Program

#include <iostream>

using namespace std;

class Node

{

int data;

Node \*next;

public:

Node(){};

void SetData(int aData) { data = aData; };

void SetNext(Node \*aNext) { next = aNext; };

int Data() { return data; };

Node \*Next() { return next; }

};

class List

{

Node \*head;

public:

List() { head = NULL; };

void Print();

void Append(int data);

void Delete(int data);

};

void List::Print()

{

Node \*tmp = head;

if (tmp == NULL)

{

cout << "NULL" << endl;

return;

}

if (tmp->Next() == NULL)

{

cout << tmp->Data();

cout << " --> ";

cout << "EMPTY" << endl;

}

else

{

do

{

cout << tmp->Data();

cout << " --> ";

tmp = tmp->Next();

} while (tmp != NULL);

cout << "NULL" << endl;

}

}

void List::Append(int data)

{

Node \*newNode = new Node();

newNode->SetData(data);

newNode->SetNext(NULL);

Node \*tmp = head;

if (tmp != NULL)

{

while (tmp->Next() != NULL)

{

tmp = tmp->Next();

}

tmp->SetNext(newNode);

}

else

{

head = newNode;

}

}

void List::Delete(int data)

{

Node \*tmp = head;

if (tmp == NULL)

return;

if (tmp->Next() == NULL)

{

delete tmp;

head = NULL;

}

else

{

Node \*prev;

do

{

if (tmp->Data() == data)

break;

prev = tmp;

tmp = tmp->Next();

} while (tmp != NULL);

// Adjust the pointers

prev->SetNext(tmp->Next());

// Delete the current node

delete tmp;

}

}

int main()

{

cout << "Nama: Royhan Abdurrohim\n";

cout << "NIM: 32602200018\n\n";

List list;

list.Append(100);

list.Print();

list.Append(200);

list.Print();

list.Append(300);

list.Print();

list.Append(400);

list.Print();

list.Append(500);

list.Print();

list.Delete(400);

list.Print();

list.Delete(300);

list.Print();

list.Delete(200);

list.Print();

list.Delete(500);

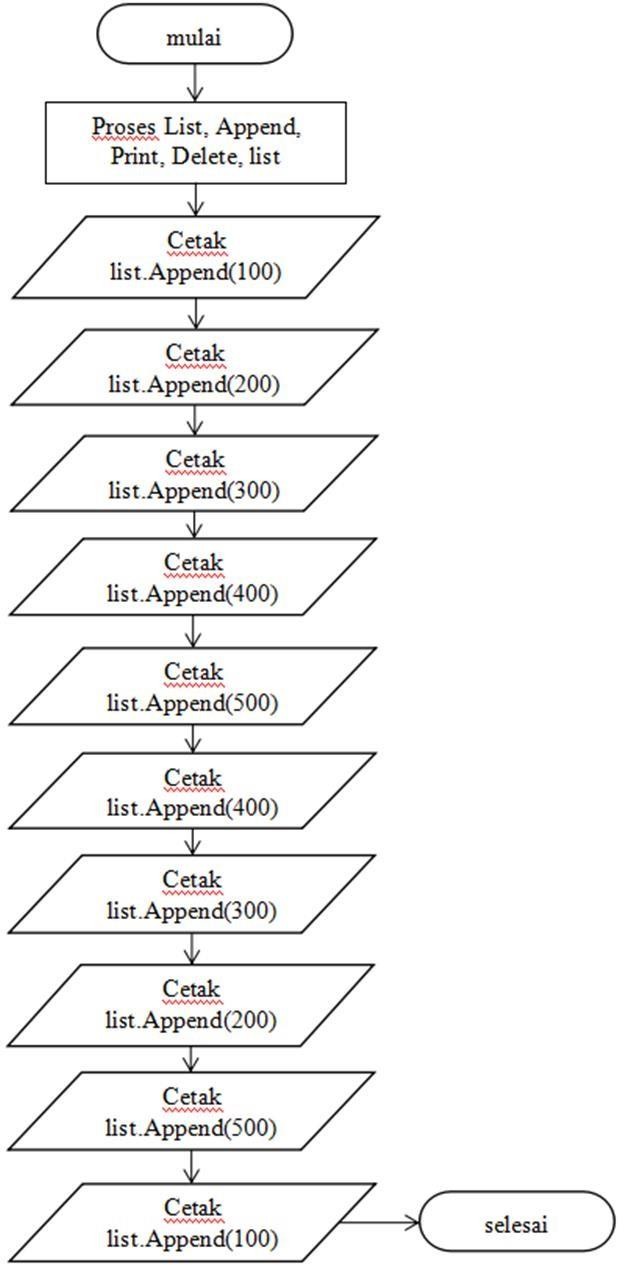
list.Print();

list.Delete(100);

list.Print();

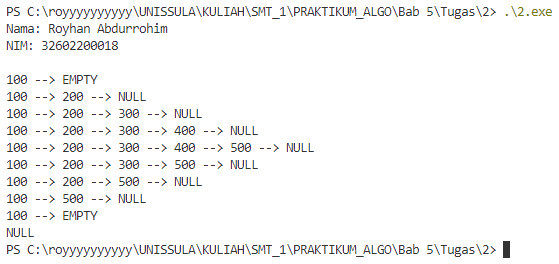
}

1. Flowchart



Gambar 5. 11 Flowchart Tugas 2

1. Output



Gambar 5. 12 Output Tugas 2

1. Penjelasan

Pada gambar 5.12 merupakan hasil dari program diatas. Program diatas merupakan program yang menggunakan bahasa C++. Yang menghasilkan tampilan untuk mencari *Linked List*. Dengan cara memasukkan program kemudian tekan *enter* untuk mengetahui hasilnya.

Pada program diatas merupakan program yang menggunakan *header iostream. Iostream* merupakan *header* yang melakukan perintah *int main()* dan *main* merupakan fungsi yang pertama dijalankan. Untuk menampilkan *list* tersebut

* + 1. Ubah Program menjadi bahasa C

Jawaban:

* + - 1. Kode Program

#include <stdio.h>

#include <stdlib.h>

typedef struct Node

{

int \_data;

struct Node \*\_next;

void (\*setData)(struct Node \*, int data);

void (\*SetNext)(struct Node \*, struct Node \*next);

int (\*getData)(struct Node \*);

struct Node \*(\*getNext)(struct Node \*);

} Node;

void setData(struct Node \*node, int data)

{

node->\_data = data;

}

void SetNext(struct Node \*node, struct Node \*next)

{

node->\_next = next;

}

int getData(struct Node \*node)

{

return node->\_data;

}

struct Node \*getNext(struct Node \*node)

{

return node->\_next;

}

void createNode(Node \*node)

{

node->\_next = NULL;

node->setData = setData;

node->SetNext = SetNext;

node->getData = getData;

node->getNext = getNext;

}

typedef struct List

{

struct Node \*head;

void (\*print)(struct List \*);

void (\*append)(struct List \*, int data);

void (\*delete)(struct List \*, int data);

} List;

void print(struct List \*list)

{

Node \*tmp = list->head;

if (tmp == NULL)

{

printf("EMPTY\n");

return;

}

if (tmp->getNext(tmp) == NULL)

{

printf("%d --> NULL\n", tmp->getData(tmp));

}

else

{

do

{

printf("%d --> ", tmp->getData(tmp));

tmp = tmp->getNext(tmp);

} while (tmp != NULL);

printf("NULL\n");

}

}

void append(struct List \*list, int data)

{

Node \*newNode = (Node \*)malloc(sizeof(Node));

createNode(newNode);

newNode->setData(newNode, data);

newNode->SetNext(newNode, NULL);

Node \*tmp = list->head;

if (tmp != NULL)

{

while (tmp->getNext(tmp) != NULL)

{

tmp = tmp->getNext(tmp);

}

tmp->SetNext(tmp, newNode);

}

else

{

list->head = newNode;

}

}

void delete(struct List \*list, int data)

{

Node \*tmp = list->head;

if (tmp == NULL)

return;

if (tmp->getNext(tmp) == NULL)

{

free(tmp);

list->head = NULL;

}

else

{

Node \*prev;

do

{

if (tmp->getData(tmp) == data)

break;

prev = tmp;

tmp = tmp->getNext(tmp);

} while (tmp != NULL);

prev->SetNext(prev, tmp->getNext(tmp));

free(tmp);

}

}

void createList(List \*list)

{

list->head = NULL;

list->print = print;

list->append = append;

list->delete = delete;

}

int main()

{

printf("Nama: Royhan Abdurrohim\n");

printf("NIM: 32602200018\n\n");

List list;

createList(&list);

list.append(&list, 100);

list.print(&list);

list.append(&list, 200);

list.print(&list);

list.append(&list, 300);

list.print(&list);

list.append(&list, 400);

list.print(&list);

list.append(&list, 500);

list.print(&list);

list.delete(&list, 400);

list.print(&list);

list.delete(&list, 300);

list.print(&list);

list.delete(&list, 200);

list.print(&list);

list.delete(&list, 500);

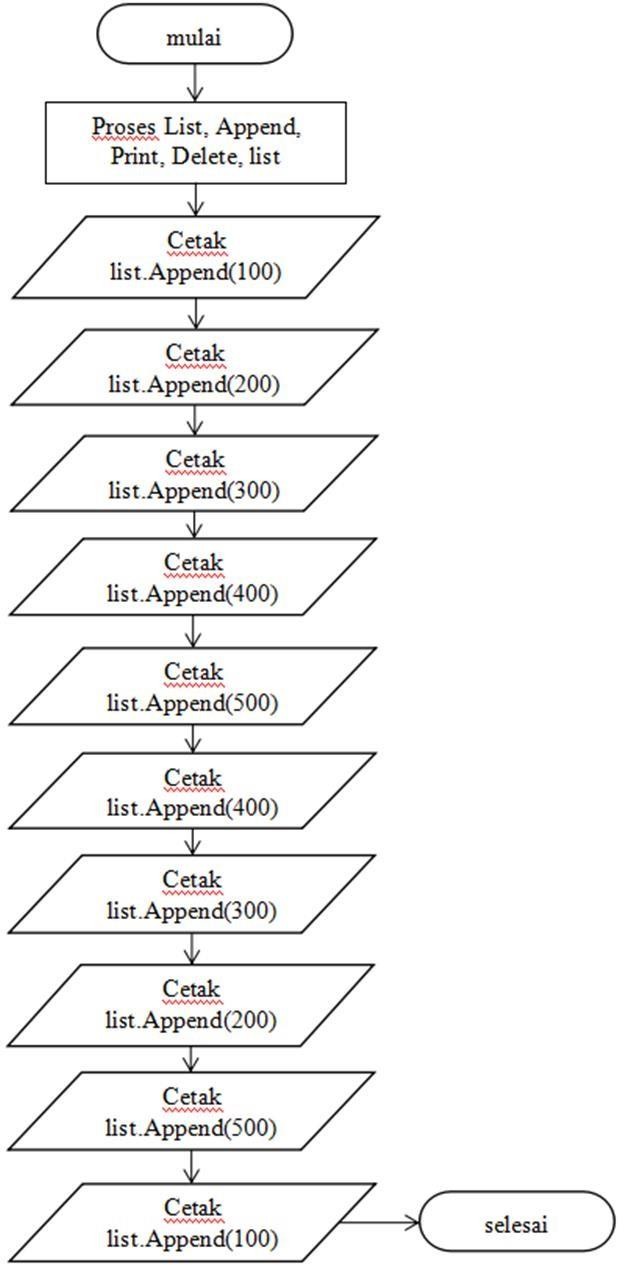
list.print(&list);

list.delete(&list, 100);

list.print(&list);

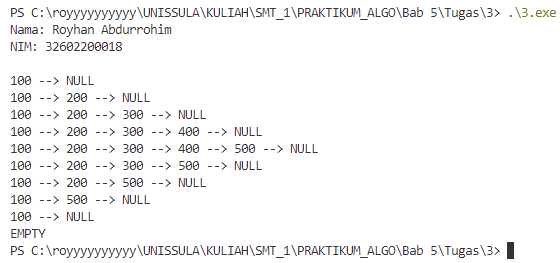
}

* + - 1. Flowchart



Gambar 5. 13 Flowchart Tugas 3

* + - 1. Output



Gambar 5. 14 Output Tugas 3

* + - 1. Penjelasan

Pada gambar 5.14 merupakan hasil dari program diatas. Program diatas merupakan program yang menggunakan bahasa C. Yang menghasilkan tampilan untuk mencari *Linked List*. Dengan cara memasukkan data yang ingin dibuat link list ke dalam int main yang akan dijalankan kemudian akan dijalankan link list mulai dari proses penambahan dan delete.

* 1. Kesimpulan

Dengan mempelajari BAB V, kita dapat mengetahui rumus logika serta fungsi-fungsi dari program *linked list*, *stack*, dan *queue*. Perbandingan antara *linked list*, *queue*, dan *stack* adalah untuk *stack* dan *queue* yang berukuran besar, terutama jumlah maksimal data tidak diketahui, menggunakan *linked list*, untuk perangkat yang memiliki memori terbatas, seperti *small handheld devices*, *linked list* memiliki performa yang bagus. *Linked List* atau dikenal juga dengan sebutan senarai berantai adalah struktur data yang terdiri dari urutan *record* data dimana setiap *record* memiliki *field* yang menyimpan alamat/referensi dari *record* selanjutnya (dalam urutan).