Aplikasi Graf Berarah dan Berlabel untuk Merancang Algoritma Pemrograman dengan Paradigma Prosedural

Imam Nurul Hukmi - 13519150
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13519150@std.stei.itb.ac.id

***Abstract*—Pemrograman merupakan salah satu bidang ilmu yang mempelajari tentang cara merancang sebuah solusi dari persoalan yang dihadapi, biasanya pada komputer. Rancangan solusi yang dibentuk disebut dengan algoritma. Sesuai masalahnya, terdapat beberapa paradigma atau sudut padang pendekatan dalam merancang sebuah algoritma, namun yang paling sering dipakai adalah paradigma prosedural atau imperatif. Terdapat beberapa metode yang dapat dipakai untuk merancang sebuah algoritma dengan paradigma prosedural, salah satunya adalah dengan metode bagan alir atau *flowchart* yang memanfaatkan graf berarah.**

***Keywords*—Algoritma, Graf, Graf Berarah, Pemrograman.**

I. PENDAHULUAN

Saat ini, komputer tidak dapat dipisahkan dari kehidupan manusia. Komputer adalah instrumen yang paling berpengaruh dalam kehidupan manusia. Hampir semua hal di dunia ini telah dipengaruhi oleh komputer, mulai dari pertanian, perdagangan, hingga kesenian.

Komputer adalah alat yang dapat mempermudah pekerjaan manusia. Bahkan, karena cara kerjanya yang merupakan tiruan cara berpikir manusia, banyak sekali bidang pekerjaan yang dapat diotomasikan dengan memanfaat komputer ini. Namun, seberapa canggih pun sebuah komputer, pada awalnya komputer tersebut haruslah diprogram terlebih dahulu sebelum komputer tersebut dapat digunakan.

Program adalah sekumpulan instruksi pada komputer yang berfungsi untuk menentukan bagaimana sebuah komputer bekerja. Pemrograman sebuah komputer memiliki tingkat kesulitan sesuai dengan seberapa rumit jalan untuk mencapai tujuan tersebut. Sebuah komputer yang hanya bertugas untuk menghitung aritmetika sederhana pasti memiliki kerumitan program yang lebih rendah dibandingkan komputer yang berada pada *server* pusat laman web media sosial.

Setelah ditemukannya komputer pada tahun 1938, internet menjadi temuan lain yang memengaruhi bagaimana peradaban manusia bekerja. Dengan internet, semua kendala yang berhubungan dengan jarak dan waktu bukan lagi menjadi kendala. Internet telah menjadi alat yang menghubungkan semua orang di seluruh dunia. Dua pihak yang secara fisik berada pada belahan dunia yang berbeda dapat berinteraksi seakan-akan mereka berada pada meja yang sama. Mereka dapat saling bertukar pikiran dan melontarkan ide satu sama lain tanpa harus repot-repot beranjak dari rumah mereka masing-masing.

Selain untuk berkomunikasi, internet digunakan sebagai tempat berkumpulnya segala informasi, mulai dari sejarah terdahulu hingga berita terkini. Setiap orang dapat dengan mudahnya mencari informasi yang ingin ia ketahui hanya dengan mengetikkan *query* pada layar. Walaupun tidak semua informasi dapat diakses dengan mudah, bila informasi tersebut bukanlah hal yang bersifat rahasia, setiap orang pasti bisa mendapatkannya.

Pemanfaatan internet sebagai media bertukar informasi telah lama dilakukan oleh manusia. Banyak pihak-pihak yang memanfaatkan internet sebagai tempat belajar dan memberikan pengajaran antar satu sama lain. Bidang-bidang pendidikan yang tercakup pun telah sangat luas, mulai dari kuliner, konstruksi bangunan, hingga pemrograman.

Mempelajari pemrograman atau hal-hal lainnya saat ini merupakan hal yang sangat mudah bila dibandingkan dengan masa-masa dahulu. Seseorang hanya perlu mencari informasi dan *tutorial* "cara memrogram" dari internet. Setelah itu, mereka hanya perlu mengikuti panduan-panduan dari laman *tutorial* yang mereka dapatkan untuk menjadi seorang *programmer* andal.

Akan tetapi, jauh sebelum menjadi seorang pemrogram ulung, terdapat sebuah aspek yang wajib dimiliki oleh setiap pembelajar, yaitu keinginan untuk belajar. Keinginan untuk belajar adalah faktor yang dapat menentukan apakah seseorang memiliki kemungkinan untuk menjadi seorang profesional ataukah tidak. Selama seseorang memiliki tekad yang kuat, ia akan dapat mengejar cita-citanya, meskipun terlihat mustahil. Sebaliknya, seberapa berbakatpun seseorang, apabila ia tidak memiliki kemauan untuk terus belajar dan mengembangkan diri, maka hampir dapat dipastikan ia akan menjadi salah satu dari ribuan orang yang gagal.

Keinginan untuk belajar adalah salah satu hal yang penting bila seseorang ingin belajar di bidang pemrograman, apalagi bagi seseorang yang belum pernah sama sekali terlibat dalam dunia pemrograman. Pasalnya, ilmu pemrograman adalah salah satu ilmu yang dipandang sebagai ilmu yang sangat rumit untuk

dipelajari. Bila pada bidang kuliner kita telah mengamati secara langsung praktik ilmu tersebut dalam level rendah, seperti cara menggoreng telur dengan baik, lain halnya dengan bidang pemrograman. Praktik-praktik pemrograman yang selalu ditampilkan oleh media adalah pemrograman tingkat tinggi yang melibatkan banyak orang dan berbaris-baris kode. Hal tersebut tentu akan membuat kaum awam menjadi ragu untuk mencoba mempelajari ilmu pemrograman. Padahal, sama seperti bidang-bidang ilmu lainnya, pemrograman selalu dimulai dari hal-hal yang sederhana.

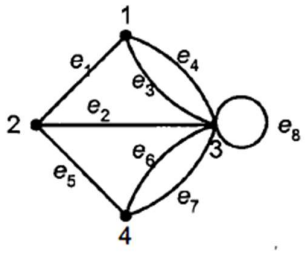
Permasalahan kedua yang sering dialami oleh kaum awam yang ingin mulai belajar pemrograman adalah wujud dari sebuah program itu sendiri. Bahasa pemrograman saat ini kebanyakan memiliki bentuk teks dengan format penulisan yang relatif tidak wajar. Meski saat ini sudah ada beberapa bahasa yang sintaksnya lebih manusiawi, namun hanya dengan wujud teksnya saja sudah membuat orang-orang malas untuk berkutat di dunia pemrograman. Padahal, yang terpenting dalam dunia pemrograman adalah cara berpikirnya dalam memecahkan masalah dibandingkan dengan kemahiran dalam menulis kode. Sehingga, pada dasarnya seseorang dapat menjadi seorang *programmer* tanpa harus menulis satu barispun kode.

Berdasarkan pernyataan di atas, seseorang dapat merancang suatu program tanpa harus berada di depan komputer. Meski begitu, diperlukan suatu metode yang baik untuk menyampaikan rancangannya kepada pihak lain. Beberapa cara yang dapat dilakukan adalah penyampaian secara lisan atau dengan menuliskan program dengan notasi algoritma di atas kertas. Kedua metode tersebut adalah metode yang paling sering digunakan oleh pemrogram profesional dalam menyampaikan idenya kepada orang lain. Namun, bagi orang yang belum terbiasa menyampaikan pendapatnya kepada orang lain dan belum nyaman merealisasikan idenya dalam notasi algoritma, terdapat solusi ketiga yang lebih mudah dipandang dan dipahami, yaitu dengan memanfaatkan graf berarah dalam bentuk bagan alir atau *flowchart*.

II. Landasan Teori

A. Graf

Graf adalah sebuah objek yang terdiri dari himpunan simpul (*node* atau *vertice*) dan himpunan sisi (*arc* atau *edge*) yang berfungsi untuk menghubungkan dua buah simpul. Secara umum graf dapat dituliskan dengan notasi $G = (V, E)$ di mana G adalah graf itu sendiri, V adalah himpunan simpul pada graf dan E adalah himpunan yang menghubungkan simpul graf tersebut. Penulisan simpul dapat dilakukan dengan simbol huruf, angka, maupun gabungan keduanya. Sedangkan, penulisan sisi biasa dilakukan dengan lambang $e_1, e_2, ..., e_n$ atau dengan menuliskan kedua simpul yang dihubungkan oleh sisi tersebut, seperti (a, b) untuk menuliskan sisi yang menghubungkan simpul a dengan simpul b. Pernyataan sisi secara lengkap dapat dilakukan dengan menuliskan kedua cara dalam sebuah persamaan $e_1 = (a, b)$.



Gambar 2.1 Sebuah graf

Berdasarkan gambar, kita dapat menuliskan graf di atas dengan notasi $G = (V, E)$ dengan anggota tiap himpunan: $V = \{1, 2, 3, 4\}$ dan $E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$.

Selain simpul dan sisi, dalam graf terdapat beberapa istilah yang bersangkutan dengan simpul, sisi, ataupun graf itu sendiri:

1. Ketetanggaan, dua buah simpul pada graf tak berarah disebut bertetangga bila terdapat sebuah sisi yang menghubungkan keduanya.
2. Bersisian, sebuah sisi dikatakan bersisian dengan sebuah simpul bila sisi tersebut menghubungkan simpul terkait dengan simpul lainnya.
3. Simpul terpencil, yaitu simpul yang tak bertetanggaan dengan simpul manapun.
4. Graf kosong, yaitu graf yang himpunan sisinya kosong.
5. Derajat, yaitu jumlah sisi yang bersisian dengan sebuah simpul.
6. Lintasan, yaitu sisi-sisi berhubungan yang dapat ditempuh dari simpul A ke simpul B.
7. Sirkuit atau Siklus, yaitu lintasan yang berasal dan berakhir di simpul yang sama.

Berdasarkan orientasi arah dari sisi yang dimiliki oleh sebuah graf, graf dapat dibagi menjadi dua jenis:

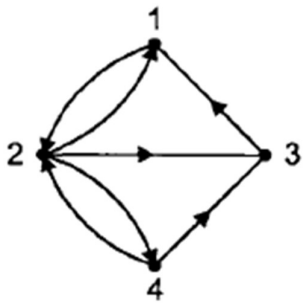
1. Graf tak berarah, yaitu graf tanpa orientasi arah pada sisinya,
2. Graf berarah, yaitu graf dengan orientasi arah pada sisinya.

Selain itu, berdasarkan kemungkinan simpul dan sisi untuk menampung sebuah data, graf dibedakan menjadi dua jenis:

1. Graf berlabel, yaitu graf yang simpul dan/atau sisinya menampung data,
2. Graf tak berlabel, yaitu graf yang identitas setiap simpul dan sisi hanya ditunjukkan oleh namanya.

Penggambaran himpunan objek dengan graf berguna untuk menunjukkan hubungan antar objek tersebut. Salah satu contoh penggunaan graf adalah penggambaran peta sebuah kota. Apabila diamati dengan cermat, kita dapat memandang peta tersebut sebagai sebuah graf dengan persimpangan dianggap sebagai simpul yang dihubungkan oleh jalan yang berperan sebagai sisi dari graf.

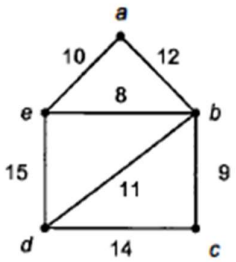
B. Graf Berarah



Gambar 2.2 Graf berarah

Graf berarah adalah graf yang setiap sisinya memiliki orientasi arah. Graf jenis ini memungkinkan penggambaran himpunan simpul yang setiap simpulnya memiliki relasi atau hubungan yang berlaku satu arah atau tidak bolak-balik. Beberapa contoh aplikasi dari graf berarah adalah penggambaran rantai makanan pada suatu ekosistem dan permainan ular tangga.

C. Graf Berlabel



Gambar 2.3 Graf berbobot, salah satu jenis dari graf berlabel

Graf berlabel merupakan graf yang simpul dan sisinya menyimpan sebuah data lain selain nama dari simpul dan sisi itu sendiri. Graf ini biasa digunakan untuk merancang graf sebuah himpunan objek yang harus mampu menyimpan data selain dari nama simpul itu sendiri dan/atau setiap sisi yang menghubungkan simpul-simpul tersebut memiliki data tersendiri dan harus dapat diakses Ketika seseorang membaca graf tersebut. Salah satu contoh dari graf berlabel adalah graf yang menggambarkan peta jarak antarkota. Pada graf ini nama simpul akan menunjukkan setiap kota. Data pada setiap sisi pada diperlukan untuk menunjukkan jarak yang harus ditempuh bila seseorang bermigrasi dari kota a ke kota b dengan jalur yang dilambangkan dengan sisi penghubungnya.

D. Algoritma Pemrograman

Algoritma dalam pemrograman adalah serangkaian perintah terhingga yang tersusun secara benar, terdefinisi dengan jelas, dan dapat diimplementasikan oleh komputer. Selain struktur data, algoritma adalah komponen terpenting lain dalam perancangan sebuah program. Bahkan, pada program yang sederhana, struktur data tidak terlalu perlu diperhatikan, sehingga algoritma itu sendiri adalah programnya.

Pada dasarnya, dalam menyampaikan sebuah ide algoritma, secara nonformal seseorang dapat menyampaikannya dengan bahasa sehari-hari. Namun, secara formal penulisan algoritma memiliki aturan penulisan khusus yang disebut sintaks. Sama seperti operator dalam aritmetika, operator-operator dalam algoritma diatur dalam sintaks. Saat ini terdapat banyak sekali bahasa pemrograman dengan sintaks-sintaksnya sendiri. Sehingga, telah ditetapkan sebuah sintaks yang khusus digunakan untuk menyampaikan sebuah algoritma pemrograman, yaitu notasi algoritmik.

E. Paradigma Prosedural

Paradigma adalah sudut pandang yang dijadikan sebagai acuan utama dalam menyelesaikan sebuah persoalan atau tujuan. Paradigma mempersempit alur berpikir kita dengan memokuskan kita pada kenyataan yang dapat dipandang dari paradigma tersebut tanpa menunjukkan fakta yang dapat diamati dari sudut pandang yang lain. Oleh karena itu, mengandalkan hanya satu saja paradigma untuk menyelesaikan sebuah masalah bukan merupakan hal yang bijak.

Pernyataan di atas berlaku untuk paradigma-paradigma yang ada pada bidang pemrograman. Terdapat beberapa paradigma yang telah berkembang di dunia saat ini. Berkembangnya paradigma-paradigma dalam dunia perprograman didasarkan pada kenyataan bahwa tidak semua permasalahan dapat dipecahkan hanya dengan satu paradigma saja. Paradigma-paradigma itu ialah:

1. Paradigma prosedural atau imperatif,
2. Paradigma fungsional,
3. Paradigma deklaratif, predikatif, atau lojik,
4. Paradigma berorientasi objek,
5. Paradigma konkuren, dan
6. Paradigma relasional.

Dari keenam paradigma di atas, paradigma prosedural merupakan paradigma yang paling banyak digunakan. Paradigma prosedural atau imperatif adalah cara berpikir dalam menyelesaikan masalah dengan metode *step-by-step.* Sebuah solusi permasalahan akan ditampilkan dalam bentuk perintah-perintah sederhana yang dijalankan satu persatu sesuai dengan urutan pemaparannya. Meski begitu, perintah-perintah kecil tersebut pada akhirnya akan berhasil menyelesaikan masalah yang ada.

Populernya paradigma prosedural dalam dunia pemrograman disebabkan karena pada dasarnya komputer bekerja dengan memroses setiap perintah tahap demi tahap, baris demi baris. Selain itu, komputer pada dasarnya hanya dapat menangani perintah-perintah aritmetika sederhana, seperti pertambahan, pengurangan, dan perkalian. Sehingga, setiap paradigma yang ada pada setiap bahasa pemrograman pada akhirnya akan diproses sebagai algoritma dengan paradigma prosedural.

III. Perancangan Algoritma Pemrograman Berparadigma Prosedural dengan Graf

Dalam perancangan algoritma pemrograman, graf berperan sebagai pengontrol alur berjalannya program tersebut. Graf berarah memiliki bentuk penunjukan alur yang jelas sehingga orang awam yang mengamati algoritma program tidak akan terlalu kebingungan. Penggunaan label pada setiap simpul

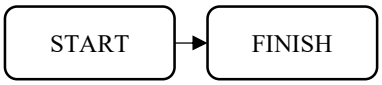
diperlukan untuk mencantumkan instruksi yang harus dilakukan pada setiap simpul yang dilewati dalam menelusuri program.

Perlu diketahui, penggambaran sebuah graf algoritma pemrograman pada dasarnya tidak memiliki sintaks khusus, karena biasanya graf ini hanya berfungsi untuk memperjelas algoritma yang akan disampaikan. Pemrogram dapat menggunakan sintaks yang mereka inginkan, mulai dari bahasa manusia hingga bahasa pemrograman khusus, asalkan pihak yang akan memperhatikan penyampaian kita mengerti dengan sintaks tersebut. Pada contoh-contoh di bawah, penggambaran graf algoritma pemrograman akan menggunakan sintaks yang mendekati notasi algoritmik untuk mempermudah adaptasi dari graf menuju notasi algoritmik nantinya.

A. Penggolongan Simpul

Dalam membuat sebuah graf, biasanya semua objek yang diwakilkan oleh setiap simpul memiliki jenis yang serupa, termasuk dalam perancangan algoritma. Pada dasarnya simpul pada graf algoritma pemrograman memiliki fungsi untuk menampung instruksi program. Penggolongan simpul di sini dimaksudkan untuk memperapi dan mempermudah pembacaan algoritma.

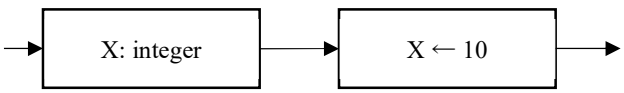
1. Simpul Terminal



Gambar 3.1.1 Simpul terminal pada graf algoritma

Dalam graf algoritma, simpul terminal berfungsi sebagai awalan dan akhiran sebuah program. Simpul ini digambarkan dengan bentuk segi empat yang sisinya melengkung. Label dalam simpul tersebut berisi "START" atau "MULAI" untuk simpul awalan dan "FINISH" atau "SELESAI" untuk simpul akhiran. Namun, terkhusus untuk upaprogram (prosedur/fungsi), isi dari simpul awalan merupakan nama dari upaprogram itu dengan parameter yang diterimanya, sedangkan untuk simpul akhiran pada sebuah fungsi berisi nilai yang dikeluarkan. Simpul khusus yang menandakan awalan dan akhiran dibutuhkan karena dalam algoritma yang rumit, akan sulit untuk mencari simpul yang harus pertama kali dibaca apabila simpul tersebut tidak memiliki ciri tersendiri. Selain itu, simpul akhiran dibutuhkan untuk memperjelas kapan sebuah program harus berhenti.

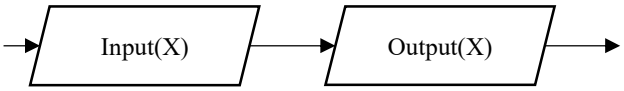
2. Simpul Pemrosesan



Gambar 3.1.2 Simpul pemrosesan dalam graf algoritma

Simpul pemrosesan merupakan jenis simpul yang akan paling banyak digunakan dalam rancangan algoritma pemrograman. Simpul ini memiliki fungsi untuk memberikan instruksi selama program berjalan sesuai dengan perintah yang ada di dalam simpul tersebut. Simpul pemrosesan memiliki bentuk segi empat biasa.

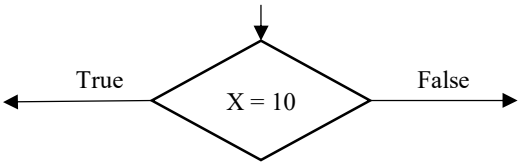
3. Simpul Masukan dan Keluaran



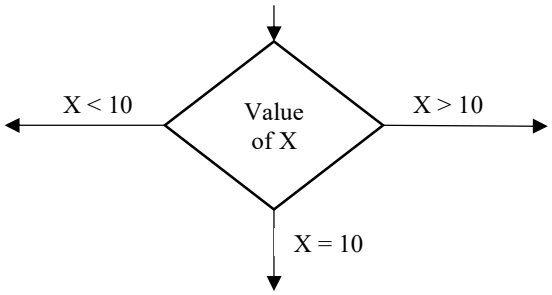
Gambar 3.1.3 Simpul masukan dan keluaran dalam graf

Simpul masukan dan keluaran berfungsi untuk memberikan perintah yang berisi penerimaan masukan (*input)* dari pengguna atau pemberian keluaran (*output*) ke pengguna. Simpul ini memiliki bentuk jajar genjang.

4. Simpul Keputusan



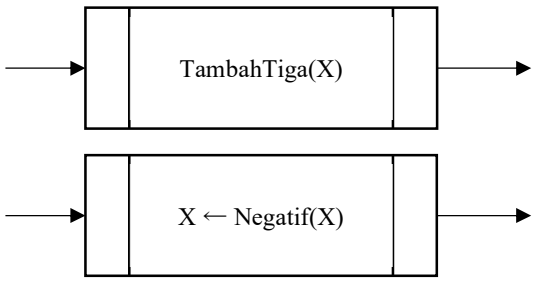
Gambar 3.1.4 Simpul keputusan dengan dua kemungkinan



Gambar 3.1.5 Simpul keputusan dengan lebih dari dua kemungkinan

Simpul keputusan pada graf algoritma berfungsi sebagai tempat terjadinya pemisahan alur program. Penentuan jalan program ditentukan oleh parameter yang dicantumkan dalam simpul. Sisi yang berasal dari simpul ini memiliki bobot yang menentukan alur program selanjutnya. Simpul ini memiliki bentuk belah ketupat.

5. Simpul Pemanggilan Fungsi/Prosedur

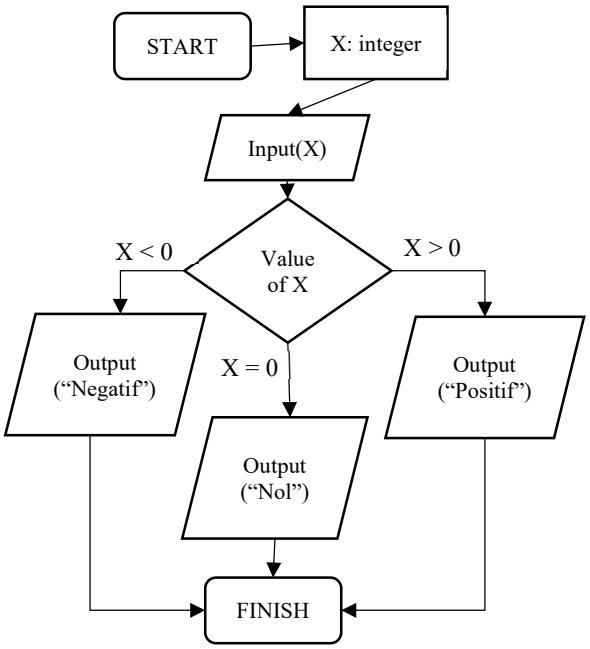


Gambar 3.1.6 Simpul pemanggilan fungsi/prosedur

Seperti namanya, simpul pemanggilan fungsi/prosedur berfungsi sebagai wadah untuk instruksi yang berisi pemanggilan fungsi atau prosedur terkait. Simpul ini berbentuk segi empat dengan garis ganda pada bagian sisi sampingnya.

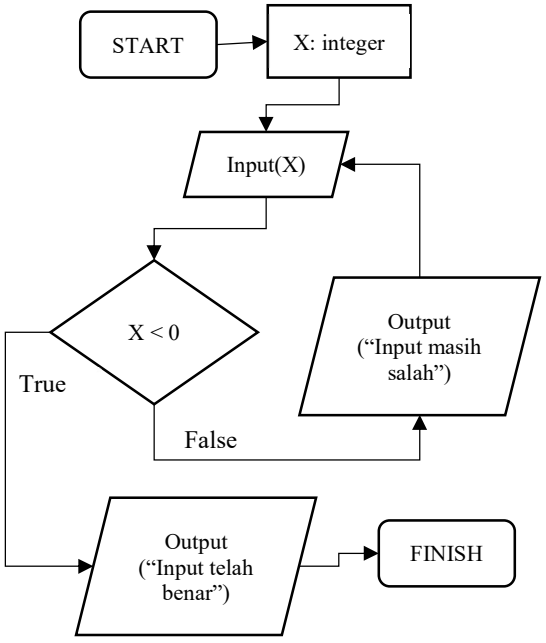
B. Contoh Program dengan Algoritma Khusus

1. Program dengan Analisis Kasus

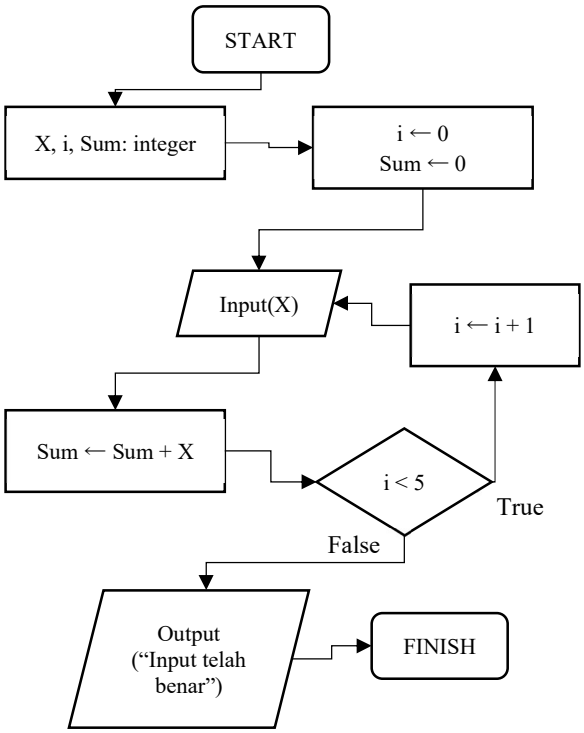


Gambar 3.2.1 Contoh program dengan analisis kasus

2. Program dengan Aksi Sekuensial



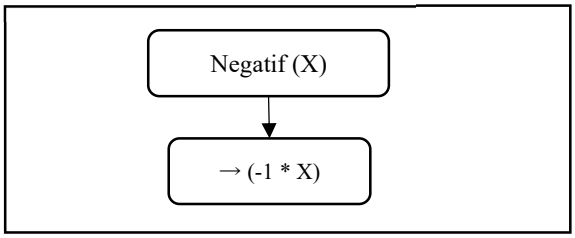
Gambar 3.2.2.1 Contoh program dengan aksi sekuensial berdasarkan kasus



Gambar 3.2.2.2 Contoh program dengan aksi sekuensial dengan jumlah tetap

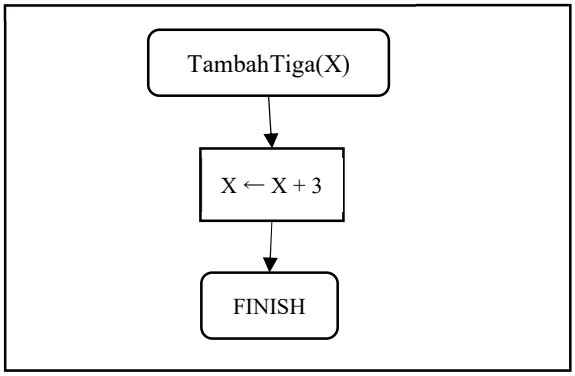
3. Realisasi Fungsi dan Prosedur

Negatif (X: integer) → integer



Gambar 3.2.3.2 Contoh realisasi sebuah fungsi

TambahTiga (input X: integer)



Gambar 3.2.3.1 Contoh realisasi sebuah prosedur

C. Kelebihan dan Kekurangan Penggunaan Graf dalam Merancang Algoritma Pemrograman

Sama seperti metode lainnya, penggunaan graf dalam merancang sebuah algoritma memiliki kelebihan dan kekurangannya sendiri. Berbeda dengan notasi algoritma, graf memiliki wujud yang bersifat mengalir dari satu simpul ke simpul lainnya sehingga lebih mudah diikuti dan dipahami oleh pemula. Selain itu, bentuknya yang lebih menarik dibandingkan penulisan kode yang berbaris-baris tidak akan mengintimidasi pemula. Berdasarkan kedua kelebihan di atas, penggunaan graf dalam merancang algoritma pemrograman cocok dilakukan dalam rangka memperkenalkan kaum awam dengan dunia pemrograman.

Dari wujud graf itu sendiri pun kita dapat melihat kekurangan dari metode ini. Metode graf dalam merancang sebuah algoritma memakan banyak waktu dan ruang bila dibandingkan dengan penulisan dalam bentuk notasi algoritmik atau sintaks bahasa pemrograman pada umumnya. Dalam praktiknya semakin kompleks suatu algoritma maka maka graf yang harus digambarkan akan semakin besar dan rumit. Penggunaan graf yang awalnya bertujuan untuk mempermudah pemahaman pun lama-kelamaan akan menjadi faktor yang malah akan mempersulit para pengamat untuk memahami algoritmanya. Sehingga, sangat dianjurkan apabila seseorang yang telah mulai paham dengan cara kerja algoritma dan program untuk cepat diarahkan untuk berpindah dari graf menuju notasi algoritmik.

IV. KESIMPULAN

Pemrograman merupakan bidang keilmuan yang dibutuhkan saat ini. Dengan semakin banyaknya individu-individu yang tertarik dengan bidang ini, diperlukan pula metode untuk mempermudah mereka dalam proses awal mempelajari pemrograman. Untuk itu, pemanfaatan graf sebagai alternatif perancangan algoritma pemrograman. Selain lebih mudah dipahami daripada notasi algoritmik bagi para pemula, metode ini pun memiliki wujud yang lebih bersahabat dibandingkan pesaingnya, sehingga para pemula tidak akan terlalu ragu untuk mencoba belajar pemrograman. Namun, penggambarannya yang memakan waktu dan tempat dapat menjadi masalah dalam perancangan algoritma yang kompleks. Sehingga, para pelajar yang sudah mulai mengerti cara kerja algoritma dengan paradigma prosedural untuk segera berpindah dari perancangan dengan metode graf menuju notasi algoritmik.

V. UCAPAN SYUKUR DAN TERIMA KASIH

Penulis memanjatkan rasa syukur sebesar-besarnya kepada Tuhan Yang Maha Esa karena dengan kehendak-Nya penulis memiliki kesehatan dan kesanggupan untuk menyelesaikan makalah ini. Penulis juga berterima kasih kepada segenap tenaga dosen IF2120 – Matematika Diskrit karena telah memberikan ilmu dan wawasan luas sebagai bekal penulis dalam menyelesaikan makalah ini.

REFERENSI

Munir, Rinaldi. 2014. *Matematika Diskrit*. Bandung: Informatika Bandung.

Liem, Inggriani. 2007. "Draft Diktat Kuliah Dasar Pemrograman (Bagian Pemrograman Prosedural)". Unpublished.

Anonim. 2020. "Preparation of Papers for IF2120 Matematika Diskrit". Bandung: Program Studi Teknik Informatika.

Flowchart in Programming. Pragramiz: Learn Code for Free. diakses pada 11 Desember 2020. https://www.programiz.com/article/flowchart-programming

The Definitive Glossary of Higher Mathematical Jargon. Math Vault | Learn Higher Mathematics The Online Way. diakses pada 10 Desember 2020. https://mathvault.ca/math-glossary/#algo

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bogor, 11 Desember 2020

Imam Nurul Hukmi - 13519150