## Pengantar Program dan Bahasa Pemrograman

[](https://www.dicoding.com/academies/620/tutorials/32395?from=32390)

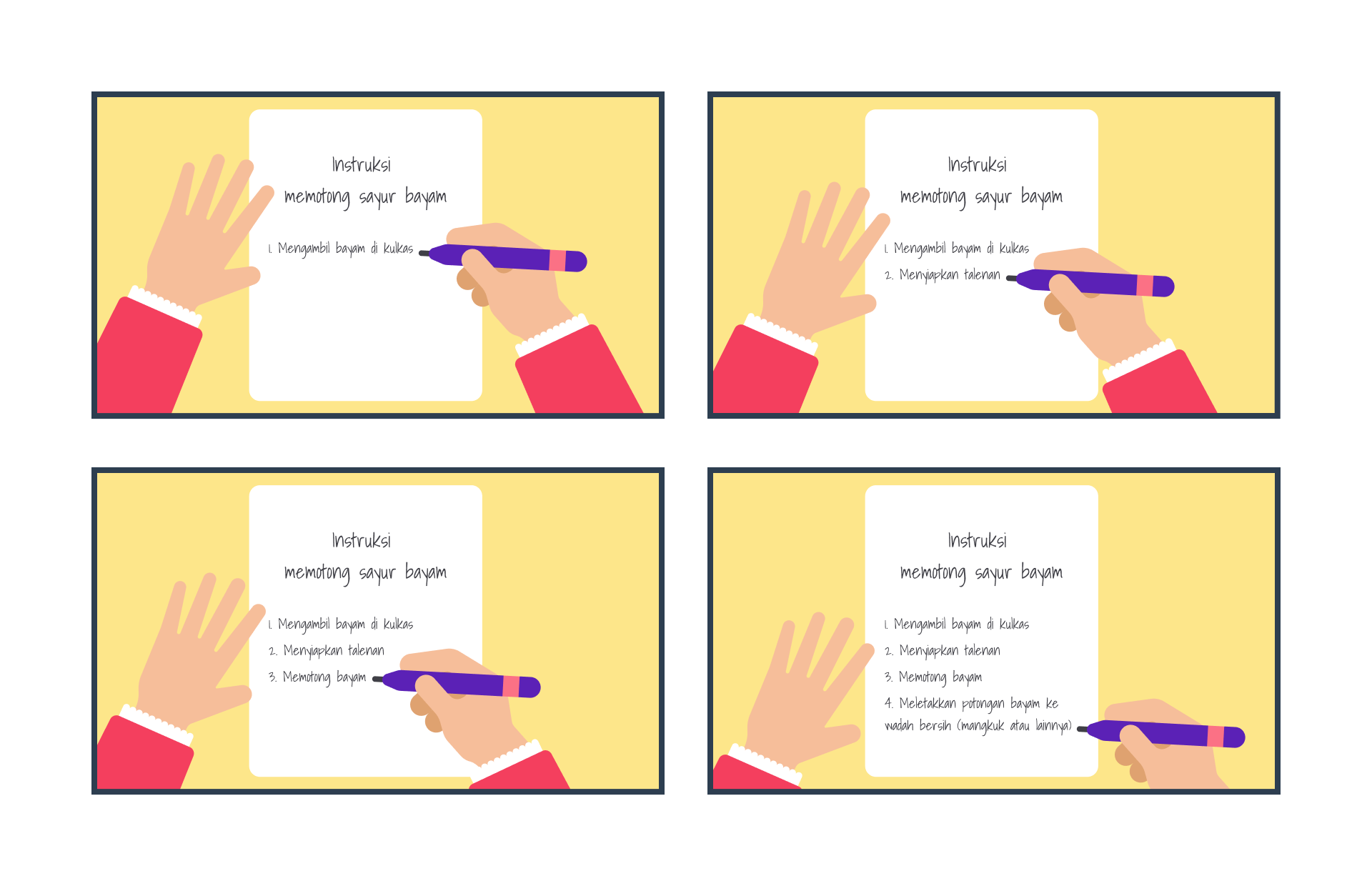
Halo! Selamat datang di kelas Belajar Pemrograman Prosedural dengan Python. Dalam materi Program dan Bahasa Pemrograman, Anda akan mengetahui perbedaan antara program dan bahasa pemrograman secara mendasar. Untuk mengenal perbedaan kedua materi tersebut, Anda akan belajar beberapa materi:

* memahami cara berpikir jalannya sebuah program;
* mengetahui macam-macam bahasa pemrograman berdasarkan paradigmanya;
* menganalisis karakteristik serta perbedaan antara program dan bahasa pemrograman; serta
* memahami lebih dalam pengertian program dalam konteks pengembangan perangkat lunak.

Mungkin materi di atas terlihat menantang. Akan tetapi, percayalah bahwa materi di atas akan membawa Anda lebih terbuka dan menyadari perbedaan satu bahasa pemrograman dengan lainnya.

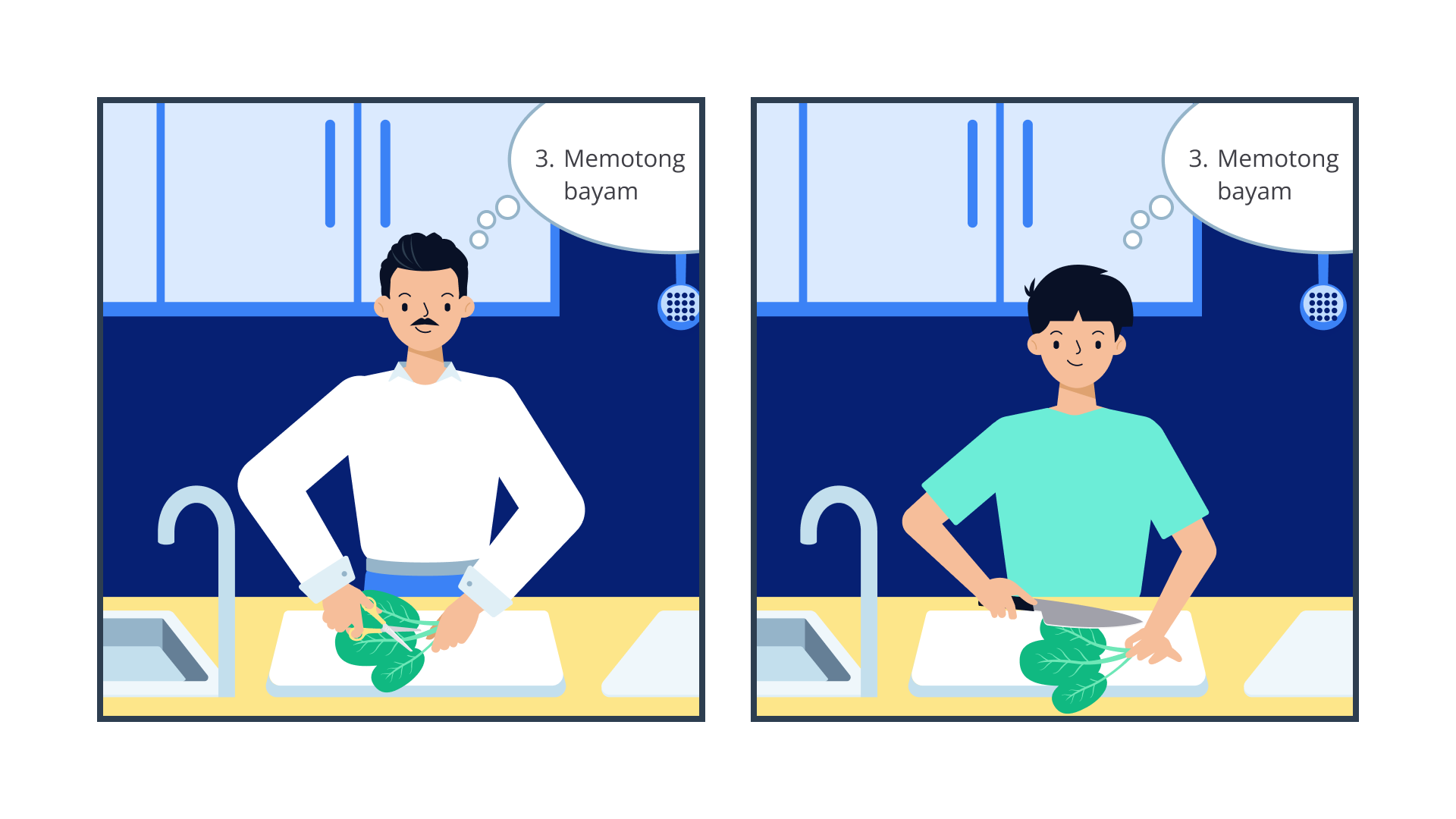
Sebelum memulai materi ini lebih dalam, kita lakukan senam otak, yuk. Permulaan ini akan membawa Anda lebih paham perbedaan program dengan bahasa pemrograman. Silakan amati analogi di bawah ini, ya.

Bayangkan Ibu ingin mengajarkan anaknya memotong bayam. Dalam proses memotong, Ibu melakukannya dengan hati-hati. Memulai dengan mengambil bayam dari lemari es, menyiapkan talenan, memotong bayam, dan meletakkan potongan bayam ke dalam sebuah wadah bersih. Ibu menuliskan seluruh instruksi tersebut pada sebuah kertas dengan harapan supaya anaknya atau orang lain dapat mengikuti proses yang sama.

[](https://www.dicoding.com/academies/620/tutorials/32395?from=32390)

Catatan instruksi tersebut disimpan dan dapat digunakan oleh siapa pun. Apabila catatan diberikan ke anaknya atau ayahnya, mereka akan paham dengan mudah. Instruksi yang Ibu tulis harus tepat dan benar juga. Jika salah menulis, instruksi akan membahayakan bahkan mencelakakan orang lain.

Aksi Ibu tersebut sama seperti ketika Anda menuliskan program. Program dalam hal ini adalah langkah demi langkah tentang hal yang perlu dilakukan, sedangkan bahasa pemrograman adalah alat yang digunakan untuk menuliskan instruksi tersebut.

[](https://www.dicoding.com/academies/620/tutorials/32395?from=32390)

Ketika orang membaca instruksi tersebut, ia akan mencoba mengikuti setiap langkahnya dengan saksama. Anaknya mungkin akan menggunakan pisau dapur pada umumnya. Berbeda cerita saat bapaknya mencoba memotong sayur menggunakan gunting dengan alasan lebih mudah dan nyaman. Setiap orang memiliki caranya tersendiri dalam urusan memotong sayur.

Sama halnya dengan program pada sebuah komputer. Komputer akan mengikuti setiap langkah instruksi dan menghasilkan keluaran yang sesuai dengan keinginan kita apabila kode ditulis dengan benar. Ketika Anda menggunakan bahasa pemrograman apa pun, compiler atau interpreter akan menerjemahkan setiap instruksi yang diberikan menjadi bahasa biner yang dimengerti oleh komponen komputer.

Dengan kata lain, bahasa pemrograman adalah sebuah alat yang mampu menerjemahkan setiap instruksi yang kita berikan. Walau menggunakan bahasa pemrograman apa pun, kita dapat menjalankan instruksi dengan keluaran yang sesuai harapan kita.

Nah, apakah Anda sudah membayangkan bedanya program dan bahasa pemrograman? Ini masih kulitnya, lo. Kita akan membahas lebih dalam mengenai program dan bahasa pemrograman. Let’s go!

## Paradigma Pemrograman

[](https://www.dicoding.com/academies/620/tutorials/32400?from=32395)

Apakah Anda ingin menjadi seorang programmer yang kreatif dan solutif? Mau tahu bagaimana caranya? Cobalah dengan memahami paradigma pemrograman. Lalu, apa maksud dari paradigma pemrograman itu?

Paradigma sendiri memiliki arti kerangka berpikir, pola pikir, sudut pandang, atau perspektif yang mendasari suatu bidang pengetahuan. Paradigma membatasi dan mengondisikan jalan berpikir kita terhadap suatu pendekatan tertentu serta mengabaikan pendekatan lain. Oleh karena itu, jelas bahwa suatu paradigma hanya memberikan pandangan terbatas terhadap suatu realitas dan mempersempit wawasan.

Dalam konteks pemrograman, paradigma pemrograman merujuk pada kerangka kerja atau pendekatan yang digunakan untuk merancang, mengembangkan, dan memahami suatu program komputer. Paradigma pemrograman menggambarkan aturan, prinsip, dan konvensi yang mengatur cara pengembang memikirkan serta merancang solusi perangkat lunak.

Dalam pemrograman pun ada beberapa paradigma. Masing-masing paradigma mempunyai strategi analisis khusus untuk memecahkan persoalan. Yuk, kita bedah satu per satu beberapa paradigma dalam pemrograman.

### Paradigma Prosedural

Paradigma prosedural merupakan pendekatan pemrograman ketika program terdiri dari serangkaian prosedur atau fungsi yang mengandung urutan instruksi. Program ini tersusun atas blok-blok kode atau instruksi yang terorganisasi dengan baik. Setiap instruksi akan dieksekusi satu per satu secara sekuensial. Terkadang, beberapa instruksi menentukan instruksi berikutnya yang akan dieksekusi.

Paradigma ini didasari oleh konsep mesin Von Neumann (stored program concept) [2], yaitu sekelompok tempat penyimpanan (**memori**), yang dibedakan menjadi memori **instruksi** dan memori **data**. Kedua memori tersebut dapat diisi dengan **nama**, yang biasa kita kenal adalah variabel, dan **harga**atau disebut juga dengan nilai. Pada setiap instruksi, program akan memanipulasi kedua memori tersebut supaya menghasilkan keluaran yang kita harapkan.

|  |
| --- |
| Bayangkan Ibu ingin memasak sup untuk seluruh anggota keluarganya. Ibu memiliki beberapa jenis sayuran yang perlu dipersiapkan. Dalam paradigma prosedural, Ibu akan menjalankan beberapa langkah untuk menyelesaikan sup tersebut. Dimulai dari mencuci sayuran, memotongnya menjadi bagian kecil-kecil, memasak kaldu, dan menyajikannya dalam mangkuk bersih. Setiap langkah harus dijalankan secara sekuensial.    Perumpamaan di atas merupakan salah satu analogi paradigma prosedural. Permasalahan membuat sup dapat diselesaikan dengan menjalankan setiap instruksi secara berurutan. Setiap instruksinya memiliki tugas masing-masing dan perannya dapat mengubah alur instruksi selanjutnya. Pada dasarnya, paradigma prosedural mengedepankan instruksi yang berkelanjutan atau berurutan. |

Penggunaan paradigma prosedural ini sangat berguna ketika Anda ingin memulai belajar pemrograman. Mengapa demikian? Dengan berpikir secara prosedural, Anda dapat merancang kode dengan struktur yang runtut dan teratur. Selain itu, penyelesaian masalah dapat dipisahkan ke dalam bagian-bagian yang lebih kecil. Dengan begitu, kode program dapat dipanggil kembali pada berbagai bagian program yang berbeda.

### Paradigma Fungsional

Paradigma fungsional didasari oleh konsep pemetaan dan **fungsi** pada matematika. Fungsi dapat berbentuk fungsi “primitif” (mendasar) atau komposisi dari fungsi-fungsi lain yang telah terdefinisi.

Dengan kata lain, seluruh isi program terdiri dari fungsi matematis yang menerima masukan dan menghasilkan keluaran tanpa memikirkan keadaan atau perubahan di luar fungsi tersebut. Hal ini ditujukan supaya menghindari perubahan data yang dapat menyebabkan side effect (efek samping). Fungsi dianggap sebagai objek utama layaknya variabel. Di sisi lain, fungsi juga dapat disisipkan ke fungsi lainnya sebagai argumen (higher-order function).

|  |
| --- |
| Coba simak cerita Ibu berikut. Ibu adalah seorang pengusaha katering. Oleh sebab itu, Ibu memiliki kitchen set yang lengkap di dalam rumahnya. Dia memiliki tempat khusus untuk menjalankan beberapa kegiatan di dapur, seperti kitchen sink (tempat untuk mencuci atau membersihkan), tempat memotong, tempat memasak, tempat menyajikan makanan, dll. Setiap tempat memiliki tugasnya sendiri dengan keluaran (output) yang sudah ditetapkan. Hal ini sangat berguna ketika Ibu harus membuat masakan yang beragam.  [Anggota keluarga Fajar sedang membuat sup dengan membagi tugas. Bapak Fajar sedang mencuci bayam, Bima sedang memotong bayam, dan Ibu Fika sedang memasak sup.](https://www.dicoding.com/academies/620/tutorials/32400?from=32395)  Ketika memiliki pesanan katering yang cukup banyak, Ibu meminta Bapak dan Bima untuk membantu pekerjaannya. Ibu mengajari Bapak untuk mengambil dan membersihkan sayur serta mengajari Bima untuk memotong sayur. Sisanya, seperti memasak hingga menghidangkan masakan, akan dilakukan oleh Ibu. Setiap aksinya ada proses delegasi output ketika pekerjaan Bapak diserahkan ke Bima, hasil potongan Bima diserahkan ke Ibu, dan Ibu menyelesaikan masakannya hingga akhir.  Analogi ini memudahkan kita membayangkan konsep paradigma fungsional yang berfokus pada fungsi. Kita hanya memanggil fungsi yang sudah ada berdasarkan output yang diharapkan. Pemanggilan fungsi tersebut dapat berulang pada saat kita membutuhkannya kembali. |

Pendekatan ini sangat berguna dalam memecahkan masalah yang kompleks. Ketika setiap instruksi berbentuk sebuah fungsi, program akan lebih mudah untuk diperbaiki dan diuji. Selain itu, kode akan mudah untuk dipanggil kembali karena seluruh instruksi berupa fungsi.

### Paradigma Deklaratif

Paradigma ini mengedepankan **logika** komputasi tanpa menggambarkan alur kerja programnya. Ia lebih menekankan pada “apa” yang ingin dicapai atau diinginkan daripada “bagaimana” cara mencapainya.

Dalam paradigma ini, program terdiri dari kumpulan pernyataan untuk menyatakan hasil yang diinginkan dan bukan langkah-langkah terperinci untuk mencapai hal tersebut. Berbeda dengan paradigma prosedural, deklaratif lebih fokus pada spesifikasi masalah dan pemodelan logika.

|  |
| --- |
| Ketika pesanan katering meningkat, Ibu mencari asisten untuk memasak seluruh masakan. Ibu memberikan sebuah resep terperinci untuk menghidangkan masakan, lengkap dengan daftar bahan yang diperlukan. Ibu tidak memberikan instruksi spesifik tentang cara melaksanakan setiap langkahnya, tetapi memberikan panduan yang jelas tentang tahapan yang harus dilakukan.  [Ibu menyewa asisten rumah tangga (ART) untuk membantu membuat sup.](https://www.dicoding.com/academies/620/tutorials/32400?from=32395)  Berdasarkan paradigma deklaratif, Ibu berfokus pada hasil akhir yang diinginkan, yaitu hidangan enak dan siap disajikan. Ibu memberikan definisi tentang apa yang diinginkan (tujuan akhir) dan memercayakan asisten untuk menentukan langkah yang diperlukan untuk mencapai tujuan tersebut. Asisten tidak perlu khawatir tentang detail implementasinya, tetapi hanya berfokus pada tahapan yang perlu dilakukan untuk menuju hasil akhir. |

Program yang menerapkan paradigma deklaratif ini cenderung lebih mudah dipahami karena fokus pada tujuan permasalahan. Solusi yang diselesaikan dengan paradigma ini lebih elegan dan ringkas karena tidak perlu memikirkan instruksi yang rinci.

### Paradigma Berorientasi Objek

Paradigma Berorientasi Objek atau dikenal dengan Object Oriented Programming (OOP) ini didasari oleh konsep **objek**. Sebuah objek mempunyai **atribut** (kumpulan sifat) dan **properti** (kumpulan reaksi atau metode). Atribut ini berfungsi untuk menyimpan informasi tentang objek, sedangkan properti mendefinisikan perilaku yang dapat dilakukan oleh objek tersebut.

Dalam paradigma ini, satu objek dapat berinteraksi dengan objek lainnya melalui pesan. Maksudnya bagaimana? Objek dalam paradigma ini mampu berkomunikasi dengan mengirimkan pesan kepada objek lain. Pesan ini berisi permintaan untuk melakukan suatu tindakan atau operasi tertentu pada objek penerimanya. Dengan begitu, objek penerima akan menanggapi pesan berdasarkan metode yang dijalankan.

|  |
| --- |
| Ibu menyuruh Bima untuk mengelompokkan sayuran yang ada dalam kulkas. Supaya mudah, Ibu memberikan petunjuk dengan memberikan label berdasarkan nama, warna, dan harga setiap sayuran. Setiap sayuran juga memiliki perilaku yang berbeda-beda, mulai dari dipotong, direbus, dikupas, dimasak, dikukus, dll. Perilaku tersebut juga membedakan antara satu sayur dan lainnya. Saat Ibu meminta untuk mencari sayur bayam, Bima secara tanggap mencari label dan perilaku yang sesuai dengan yang diminta.  [Ada beberapa buah di meja dapur yang tersusun berdasarkan warnanya. Buah warna merah ada tomat dan cabai. Buah warna hijau ada kangkung, bayam, sawi, dan brokoli. Buah warna ungu ada kol ungu dan terong. Buah warna jingga ada wortel, labu, dan paprika orange.](https://www.dicoding.com/academies/620/tutorials/32400?from=32395)  Hal yang dilakukan Bima merupakan interaksi antara satu sayur dan lainnya. Dalam hal ini, sayur merupakan objek yang memiliki atribut, seperti nama, warna, dan harga. Selain itu, sayur memiliki properti, seperti dipotong, direbus, dll., yang akan dijalankan apabila dibutuhkan. Konsep interaksi antar objek inilah yang terjadi ketika kita menerapkan paradigma berorientasi objek. |

Paradigma ini menawarkan beberapa konsep **class**, **generic**, **inheritance**, **polymorphism**, dan **encapsulation**. Beberapa konsep tersebut ditawarkan untuk memungkinkan pengembangan program yang modular, mudah dipahami, dan dipelihara. Dengan berfokus pada objek dan interaksi antar objek, program dapat lebih efisien.

Selain keempat paradigma di atas, beberapa literatur masih sering menyebutkan paradigma lainnya, seperti berikut.

* **Paradigma konkuren [3]**: paradigma yang erat hubungannya dengan perangkat keras, memungkinkan untuk mengerjakan beberapa tugas atau proses secara bersamaan atau paralel; dan
* **Paradigma relasional [4]**: paradigma yang didasari pengolahan serta manajemen data menggunakan entitas dan relasi.

Suatu permasalahan dapat dipecahkan dengan baik menggunakan satu paradigma. Kondisi ini mengakibatkan kita dibatasi oleh satu konsep berpikir dalam menyelesaikan masalah. Padahal, suatu paradigma belum tentu cocok untuk semua persoalan yang ada.

Maka dari itu, kita perlu mempelajari konsep dasar seluruh paradigma pemrograman supaya dapat memilih pendekatan yang paling cocok untuk menyelesaikan masalah yang dihadapi. Dengan demikian, mempelajari paradigma pemrograman sangat penting bagi siapa saja yang ingin menjadi seorang programmer sukses dan berpengalaman.

Materinya sangat seru, bukan? Semoga dengan adanya penjelasan di atas, Anda menjadi makin paham tentang macam-macam paradigma pemrograman, ya.

## Bahasa Pemrograman Berdasarkan Paradigma Pemrograman

Pada materi sebelumnya, kita telah belajar macam-macam paradigma pemrograman yang ada dalam pengembangan aplikasi perangkat lunak. Suatu permasalahan dapat Anda selesaikan dengan salah satu dari beragam paradigma pemrograman. Namun, yang perlu Anda ingat bahwa satu paradigma pemrograman tidak selalu cocok untuk setiap permasalahan.

[](https://www.dicoding.com/academies/620/tutorials/32405?from=32400)

Nah, saat ini kita akan eksplorasi lebih jauh tentang bahasa pemrograman yang ada pada setiap paradigma pemrograman. Yuk, langsung saja kita bahas.

Ada banyak sekali bahasa pemrograman di dunia ini. Mulai dari bahasa tingkat rendah (bahasa mesin dalam biner), bahasa assembler (dalam kode mnemonik), bahasa tingkat tinggi, sampai bahasa generasi keempat (4GL) telah tersedia di dunia informatika. Uniknya, Anda bisa menemukan bahasa pemrograman yang diawali huruf abjad A hingga Z tanpa terkecuali. Selengkapnya, silakan kunjungi daftar bahasa pemrograman pada laman berikut [ini](https://en.wikipedia.org/wiki/List_of_programming_languages).

Bahasa pemrograman berkembang dengan cepat sejak tahun 60-an. Ibarat piramida yang semakin tinggi dan mengerucut, manusia tidak lagi memiliki pemahaman yang umum tentang berbagai bahasa pemrograman yang ada. Mereka tidak lagi saling mengerti antar bahasa pemrograman satu dengan lainnya.

Terdapat pengelompokan bahasa pemrograman berdasarkan paradigma untuk mengenal antar bahasa secara umum. Hal ini bertujuan supaya kita sebagai pengembang aplikasi dapat memilih bahasa pemrograman yang tepat berdasarkan permasalahan dan paradigma pemrograman. Berikut adalah contoh bahasa-bahasa pemrograman yang ada berdasarkan paradigma.

1. **Prosedural**: Algol, Pascal, Fortran, Basic, Cobol, C, dll.
2. **Fungsional**: Charity, Curry, F#, Haskel, Joy, dll.
3. **Deklaratif**: Prolog, Datalog, dll.
4. **Berorientasi Objek**: Smalltalk, Eiffel, Cobra, dll.
5. **Konkuren**: Occam, Ada, Java, dll.
6. **Relasional**: SQL pada basis data relasional.

Bahasa pemrograman di atas adalah beberapa dari sekian banyak bahasa yang tercatat pada paradigma pemrograman. Lantas, apakah bahasa pemrograman populer lainnya tidak termasuk bagian dari paradigma di atas? Tentu tidak. Saat ini banyak dari bahasa pemrograman yang menerapkan multi-paradigma, salah satunya adalah Python.

Multi-paradigma ini semacam pisau serbaguna, yaitu bisa dipakai dalam segala kondisi. Hal ini memungkinkan pengguna tidak perlu berganti pisau yang lain. Apabila diberikan sepotong daging atau buah, pisau serbaguna mampu memotongnya. Sama seperti pemrograman, ia mampu memilih pendekatan sesuai dengan kebutuhan.

[](https://www.dicoding.com/academies/620/tutorials/32405?from=32400)

Python adalah salah satu bahasa pemrograman yang mengakomodasi beberapa paradigma, seperti prosedural, fungsional, dan berorientasi objek. Pada konsep prosedural, Python menyediakan fitur seperti fungsi, control flow, dan variabel. Selain itu, Python juga mendukung gaya pemrograman fungsional dengan fitur higher-order function, mapping, dan filtering data. Di sisi lain, Python juga mendukung paradigma berorientasi objek dengan fitur seperti class, inheritance, dan polymorphism.

Tidak hanya Python, bahasa pemrograman yang populer juga menerapkan multi-paradigma, seperti Kotlin, Swift, Dart, dan Javascript. Dengan dukungan multi-paradigma, bahasa pemrograman tersebut dapat memberikan fleksibilitas dalam menyelesaikan berbagai masalah yang ada.

Keuntungan lain dari bahasa pemrograman multi-paradigma adalah kita lebih produktif dalam memilih pendekatan yang efisien sesuai kenyamanan pengembang aplikasi. Dari sisi keterbacaan kode, bahasa pemrograman multi-paradigma memiliki kode yang terstruktur sehingga memudahkan pemahaman dan kolaborasi antar anggota tim.

Meskipun demikian, bahasa pemrograman dengan multi-paradigma memiliki beberapa kekurangan. Dengan penulisan kode yang fleksibel, pengembang aplikasi pemula mudah mengalami kebingungan dalam menggabungkan paradigma yang berbeda. Apabila program dikerjakan secara kolaborasi, setiap pengembang aplikasi harus mampu menguasai berbagai konsep paradigma yang dipakai.

Dalam memilih bahasa pemrograman perlu adanya peninjauan dari beberapa faktor. Kita perlu mempertimbangkan kondisi anggota dalam tim dan kebutuhan proyek. Mempelajari kelebihan dan kekurangan dari setiap bahasa pemrograman akan membantu Anda dalam memilih keputusan yang tepat.

Bagaimana sampai sini? Apakah Anda sudah mampu mengelompokkan bahasa pemrograman berdasarkan paradigma? Materi selanjutnya tidak kalah menarik. Untuk itu, tetap semangat dan ikuti materinya hingga selesai, ya.

## Program Tidak Sama dengan Bahasa Pemrograman

[](https://www.dicoding.com/academies/620/tutorials/32410?from=32405)

Pada awal materi, kita diberi analogi oleh Ibu yang menjelaskan bahwa program adalah instruksi untuk menyelesaikan suatu permasalahan, sedangkan bahasa pemrograman adalah alat penerjemah setiap instruksi yang diberikan. Mungkin Anda merasa tertantang dalam memahami analogi tersebut. Yuk, kita bedah satu per satu arti program dan bahasa pemrograman.

Mempelajari pemrograman sebanding dengan belajar strategi pemecahan masalah. Metodologi dan sistematika pemecahan masalah tersebut tertuang dalam notasi yang disepakati bersama. Belajar pemrograman lebih bersifat pemahaman, analisis, dan sintesis suatu permasalahan.

Pada hakikatnya, program hanyalah instruksi yang kita olah supaya menyelesaikan suatu permasalahan. Cara berpikir kita menyusun strategi dan menguraikan permasalahan adalah hal mendasar dalam mendeskripsikan instruksi pemrograman. Metodologi pembelajaran yang baik akan membantu kita memahami cara merancang solusi yang efektif dan efisien.

Sementara itu, belajar bahasa pemrograman adalah sarana untuk belajar memakai suatu bahasa, menerapkan aturan sintaksis, dan menuruti tata cara pengoperasian instruksi pada bahasa yang bersangkutan. Jika maknanya diperluas, belajar bahasa pemrograman adalah belajar untuk memanfaatkan instruksi-instruksi yang dapat dipakai secara spesifik hanya pada bahasa itu saja. Belajar pemrograman lebih bersifat keterampilan daripada analisis dan sintesis permasalahan.

Bayangkan Ibu sedang mengajarkan Bima untuk memotong bayam. Ibu akan menguraikan permasalahan secara mendasar untuk memudahkan proses belajar Bima. Mulai dari membersihkan sayur, mencari talenan dan pisau, memotong sayur, dan meletakkan potongannya dalam mangkuk bersih. Itulah susunan instruksi program yang harus dijalankan untuk menguraikan masalah memotong bayam.

[](https://www.dicoding.com/academies/620/tutorials/32410?from=32405)

Jika permasalahan tersebut dituangkan ke dalam sebuah program, kita ubah instruksi tersebut menjadi sintaksis yang dapat dipahami oleh komputer. Apa pun bahasa pemrogramannya, instruksi yang dijalankan program selalu menyerupai cara berpikir kita dalam menguraikan permasalahan. Dengan begitu, program dapat menghasilkan keluaran sesuai dengan keinginan kita.

* [**C**](https://www.dicoding.com/academies/620/tutorials/32410?from=32405#tab1-code1)
* [Python](https://www.dicoding.com/academies/620/tutorials/32410?from=32405#tab1-code2)
* [Java](https://www.dicoding.com/academies/620/tutorials/32410?from=32405#tab1-code3)
* [Kotlin](https://www.dicoding.com/academies/620/tutorials/32410?from=32405#tab1-code4)
* [Swift](https://www.dicoding.com/academies/620/tutorials/32410?from=32405#tab1-code5)
* [Dart](https://www.dicoding.com/academies/620/tutorials/32410?from=32405#tab1-code6)

1. #include <stdio.h>
3. void main() {
4. // Mendefinisikan dua nilai yang akan dijumlahkan
5. int nilai1 = 1;
6. int nilai2 = 2;
8. // Menjumlahkan kedua nilai
9. int hasil = nilai1 + nilai2;
11. // Menjawab penjumlahan dua nilai
12. printf("Hasil penjumlahan nilai %d dan %d adalah %d", nilai1, nilai2, hasil);
13. }

Belajar pemrograman dan bahasa pemrograman memiliki tingkatan dan kesulitan yang berbeda-beda. Kita sebagai pengembang aplikasi sering kali dihadapkan dengan kedua kesulitan dalam waktu bersamaan. Terkadang kita sulit mendapatkan solusi permasalahan yang ada. Di sisi lain, kita juga sulit untuk menerjemahkan solusi tersebut ke dalam bahasa mesin.

Saat Ibu mengajarkan cara memotong bayam, instruksi yang diberikan bisa saja membuat Bima merasa kesusahan. Instruksi tersebut diibaratkan dengan program yang menuntun Bima memotong bayam. Di samping itu, Bima mungkin paham dengan instruksi yang diberikan, tetapi susah untuk menerjemahkan aksinya menggunakan pisau. Dalam proses menerjemahkan instruksi ke suatu aksi tentu keahlian dibutuhkan juga.

[](https://www.dicoding.com/academies/620/tutorials/32410?from=32405)

Oleh karena itu, kita sebagai pengembang aplikasi dituntut untuk menulis dan menganalisis program serta mengeksekusinya dalam bahasa target. Menulis program di atas kertas dan menganalisis suatu permasalahan tanpa mengeksekusi program tidaklah cukup. Sebaliknya, kita hanya menulis dan mengeksekusi program secara langsung tanpa adanya analisis yang kuat bukan merupakan praktik yang baik.

Produk yang dihasilkan oleh seorang pengembang aplikasi adalah program yang dirancang dengan metodologi dan sistem yang baik, dapat dieksekusi oleh mesin, serta berfungsi dengan benar. Selain itu, program harus mampu menangani segala kemungkinan dan disertai dengan adanya dokumentasi.

Metode terbaik untuk belajar apa pun adalah melalui contoh. Seseorang yang ingin mendalami suatu ilmu harus belajar melalui contoh nyata. Berkat contoh nyata, seseorang dapat melihat, mengalami, dan mempraktikkannya secara langsung. Dengan mempelajari melalui contoh nyata, kita dapat memperoleh pemahaman lebih dalam dan konkret tentang ilmu yang sedang dipelajari.

“**Belajar pemrograman**menitikberatkan pada cara membentuk seseorang menjadi designer (perancang) program, sedangkan **belajar bahasa pemrograman** mendidik seseorang untuk menjadi seorang coder (juru kode).” 一 Ibu Inggriani Liem

Wow, materi di atas cukup membuka wawasan kita terhadap program dan bahasa pemrograman, bukan? Sekarang, Anda telah mengerti perbedaan antara program dan bahasa pemrograman dalam proses pengembangan aplikasi. Apabila masih belum paham, Anda bisa bertanya di Forum Diskusi, ya.

## Perspektif Holistis tentang Program

[](https://www.dicoding.com/academies/620/tutorials/32415?from=32410)

Pernahkah Anda merasakan membuat program hanya mengharapkan output yang dihasilkan sesuai dengan kebutuhan tanpa memperhatikan proses atau kualitasnya? Apakah dengan cara itu membuat Anda disebut sebagai seorang desainer program atau coder? Apakah Anda berpikir bahwa membuat program hanya sebatas itu?

Sebagian besar dari Anda mungkin memiliki pertanyaan yang serupa akan hal tersebut. Kita sebagai pengembang aplikasi dituntut untuk cekatan dalam menyelesaikan permasalahan dengan tepat. Di sisi lain, tuntutan pasar yang tinggi mendesak kita untuk menghadirkan solusi inovatif, menarik, dan sesuai dengan kebutuhan pengguna dengan rentang waktu cukup cepat.

Kita harus memiliki “kacamata” atau pandangan yang jauh dalam membuat program. Buatlah program yang tidak hanya memecahkan masalah saat ini, tetapi juga untuk masa depan. Untuk itu, perlu adanya penjelasan konsep pembuatan program lebih dalam dengan harapan kita tidak asal-asalan dalam merancang dan mengembangkan solusi yang tepat.

Yuk, kita menggali ilmu lebih dalam tentang keterkaitan elemen pada pembuatan program. Let’s go!

### Produk vs. Proses

Ada pandangan pada setiap individu, termasuk para pengembang aplikasi, yang beranggapan bahwa produk merupakan aspek terpenting dalam pengembangan suatu program. Beberapa dari mereka cenderung mengabaikan peran penting proses yang terlibat dalam pembangunan program. Namun, opini tersebut kurang relevan lagi bila diterapkan di zaman sekarang.

Sebuah produk berkualitas tinggi mungkin dibangun melalui proses yang “kurang baik” atau “sangat buruk”. Jika terjadi kesalahan, pengembang aplikasi akan berupaya untuk memperbaiki dengan cara seadanya layaknya tambal sulam. Sebaliknya, proses yang baik pasti menghasilkan kualitas produk yang baik pula.

Apakah Anda familier dengan Robert C. Martin atau yang biasa dikenal sebagai Uncle Bob? Beliau adalah penulis buku terkenal tentang [Clean Code](https://learning.oreilly.com/library/view/clean-code-a/9780136083238/) dan [Clean Architecture](https://learning.oreilly.com/library/view/clean-architecture-a/9780134494272/). Kedua buku tersebut membahas cara kode dibangun melalui proses yang baik dan benar dengan tujuan meminimalkan kemungkinan error yang muncul di masa mendatang. Hadirnya kedua buku tersebut memberikan penekanan bagi kita mengenai pentingnya proses dalam pengembangan suatu program.

|  |
| --- |
| Bapak Fajar pernah mendapatkan proyek untuk mendesain dan membangun rumah. Beliau ditugaskan untuk mendesain eksterior rumah dan memantau proses pembuatan rumah. Pembangunan rumah dimulai dari memasang fondasi, membuat kerangka rumah, pemasangan dinding, jendela, pintu, hingga atap. Bapak Fajar mendesain rumah tersebut dengan terstruktur supaya pemilik rumah dapat memperbaiki dengan mudah apabila terjadi kerusakan atau kebocoran.  [Bapak Fajar sedang melihat konstruksi rumah yang belum selesai dan membayangkan hasil akhirnya.](https://www.dicoding.com/academies/620/tutorials/32415?from=32410)  Begitu pula dengan program. Program perlu dibangun secara terstruktur supaya mudah dikelola dan dipahami oleh pengembang aplikasi. Selain itu, jika program ditinggalkan oleh penulisnya, akan meninggalkan kesan baik dan membuat pengembang aplikasi yang mengambil alih merasa nyaman dalam melanjutkan pengembangan. Ketika dibutuhkan perbaikan atau perubahan pada kode, pengembang aplikasi juga akan sangat mudah memahami dan mengelolanya. Dengan begitu, program akan dapat bertahan dan relevan dalam jangka waktu yang lama jika mengalami “renovasi” atau “penambahan” fitur. |

Anda baiknya menekankan suatu produk dan proses agar berkualitas tinggi karena akan berdampak pada program dalam jangka waktu yang panjang. Anda dapat mempertahankan kondisi program dengan baik selama beberapa tahun mendatang. Apabila ada perbaikan atau penambahan fitur, program dapat beradaptasi menyesuaikan kualitas yang telah ada sebelumnya. Dengan memastikan kualitas yang baik sejak awal, program dapat berkembang dan berfungsi secara optimal untuk memenuhi kebutuhan dan tuntutan yang berubah dari waktu ke waktu.

### Skalabilitas Program

Dalam industri teknologi, skalabilitas program menjadi hal terpenting. Adakalanya, pertumbuhan suatu program meningkat seiring berjalannya waktu. Hal ini bisa diidentifikasi melalui meningkatnya jumlah pengguna, volume data, hingga kompleksitas fitur. Pembuatan suatu program seharusnya mampu beradaptasi dengan perkembangan yang ada supaya menghasilkan kinerja optimal.

Skalabilitas program ini menitikberatkan pada situasi dan kondisi program dalam menangani beban kerja yang meningkat tanpa mengurangi kualitas dan performa. Hal ini penting untuk diperhatikan agar mampu mempertahankan fungsionalitas program, waktu respons yang cepat, dan integritas program dalam menangani permasalahan.

Kita menyadari sebagai pengembang aplikasi pemula bahwa untuk memikirkan hal tersebut adalah tantangan yang berat. Program besar selalu dimulai dengan program yang kecil pula. Tidak serta-merta pembuatan program berisi fitur yang kompleks. Kita perlu berpikir selangkah demi selangkah dan lebih maju supaya pengembangan program dapat berjalan dengan lancar dan baik.

|  |
| --- |
| Pernahkah Anda melihat proses renovasi rumah? Bapak Fajar pernah mendapatkan proyek renovasi rumah dan beliau ditunjuk sebagai penanggung jawab atas proses renovasi tersebut. Bapak Fajar harus memastikan struktur rumah harus dalam kondisi yang baik dan kokoh tanpa adanya kerusakan serius. Selain itu, komponen dasar seperti fondasi, dinding, dan atap juga perlu diperiksa supaya dapat menahan beban tambahan dari renovasi.  [Bapak berada di area pembangunan rumah dan berkata, "Komponen dasar rumah harus dalam kondisi baik supaya renovasi rumah berjalan dengan baik."](https://www.dicoding.com/academies/620/tutorials/32415?from=32410)  Layaknya renovasi rumah, pembaruan atau perbaikan program pun harus memiliki fondasi yang kuat. Proses tersebut tidak tiba-tiba dilakukan walaupun terdapat kebutuhan yang mendesak. Pengembang aplikasi perlu memeriksa komponen dasar dari aplikasi untuk memastikan kemungkinan untuk diperbarui atau tidak. Jika sesuai standar, pengembang aplikasi mampu melanjutkan pembaruan program sesuai dengan kaidah penulisan program sebelumnya. Sebaliknya, pengembang aplikasi perlu mempertimbangkan ulang dan mencari solusi yang tepat untuk perbaikan program tersebut. |

### Kolaborasi Tim

Pada zaman dahulu, pengembang aplikasi mungkin bekerja secara individu dalam menyelesaikan permasalahan pengguna. Pembuatan program mungkin mustahil dilakukan secara bersama-sama karena minimalnya sumber daya pada zaman tersebut. Mereka bekerja untuk berhadapan dengan komputer, menyelesaikan masalah, dan membangun program “misterius” yang hanya dimengerti oleh dia sendiri [5].

Kondisi tersebut tidak relevan lagi pada zaman sekarang. Program yang dibangun pada masa sekarang jauh lebih besar dibandingkan dengan sebelumnya. Apabila menerapkan konsep skalabilitas program, kita perlu berpikir secara jauh mengenai hal-hal yang perlu dibangun. Untuk mencapai hal tersebut, kita perlu bekerja sama dalam pembuatan sebuah program.

Lalu, apa keuntungan dari pembuatan program secara bekerja sama? Kita dapat berkolaborasi dalam menyelesaikan suatu permasalahan yang sama. Dalam kerja sama tim, kita dapat berbagi penugasan agar lebih fokus dalam permasalahan yang kecil-kecil. Selain itu, setiap individu mampu berbagi pengalaman dan ilmu supaya menyelesaikan permasalahan dengan cepat. Dengan melibatkan berbagai pandangan dan keahlian, kualitas program menjadi lebih baik.

[](https://www.dicoding.com/academies/620/tutorials/32415?from=32410)

Apabila berkaca ke salah satu aplikasi ojek online dengan jutaan pengguna, apakah Anda pernah membayangkan proses pembuatan aplikasinya? Mungkin, saat debut pertama kali aplikasi hanya dibangun oleh beberapa orang bahkan satu orang saja. Namun, ketika aplikasi sudah berkembang pesat, mereka menambahkan sumber daya baru dengan harapan aplikasinya semakin berkualitas.

|  |
| --- |
| Ketika sedang bertugas dalam pembangunan atau renovasi rumah, mungkinkah Bapak Fajar mengerjakannya sendiri? Tentu saja tidak. Beliau dibantu oleh rekan kerja yang ahli di bidang arsitek, struktur bangunan, tukang kayu, tukang pipa, dll. Setiap anggota tim akan saling bekerja sama, berbagi informasi, dan menyelesaikan permasalahan yang ada. Walaupun keahliannya berbeda-beda, mereka tetapi memiliki satu tujuan yang sama, yaitu menghasilkan rumah yang kokoh dan fungsional.  [Bapak bersama dengan dua rekan lainnya (arsitek dan kuli bangunan) sedang berbincang dalam rangka menyelesaikan rumah.](https://www.dicoding.com/academies/620/tutorials/32415?from=32410)  Sama halnya dengan pembangunan rumah, keberhasilan suatu program dapat dicapai dengan kolaborasi tim yang solid dan saling mendukung satu sama lain. Adanya tim peneliti, desainer, coder, reviewer, dan quality control (QC) yang saling berkomunikasi satu sama lain. Setiap tim akan memiliki peran yang berbeda-beda, saling berkoordinasi, berbagi pengetahuan, dan menyelesaikan permasalahan pengguna demi menciptakan program yang berkualitas. |

Setelah membaca materi di atas, apakah Anda semakin terbuka wawasannya terhadap pembuatan program? Pada hakikatnya, kita sebagai pengembang aplikasi tidak hanya membuat program saja, tetapi perlu menitikberatkan proses pembuatan, skalabilitas program, serta berkolaborasi dengan rekan tim. Apabila ingin bertanya terkait materi ini, kita bisa berdiskusi lebih lanjut melalui forum diskusi, ya.