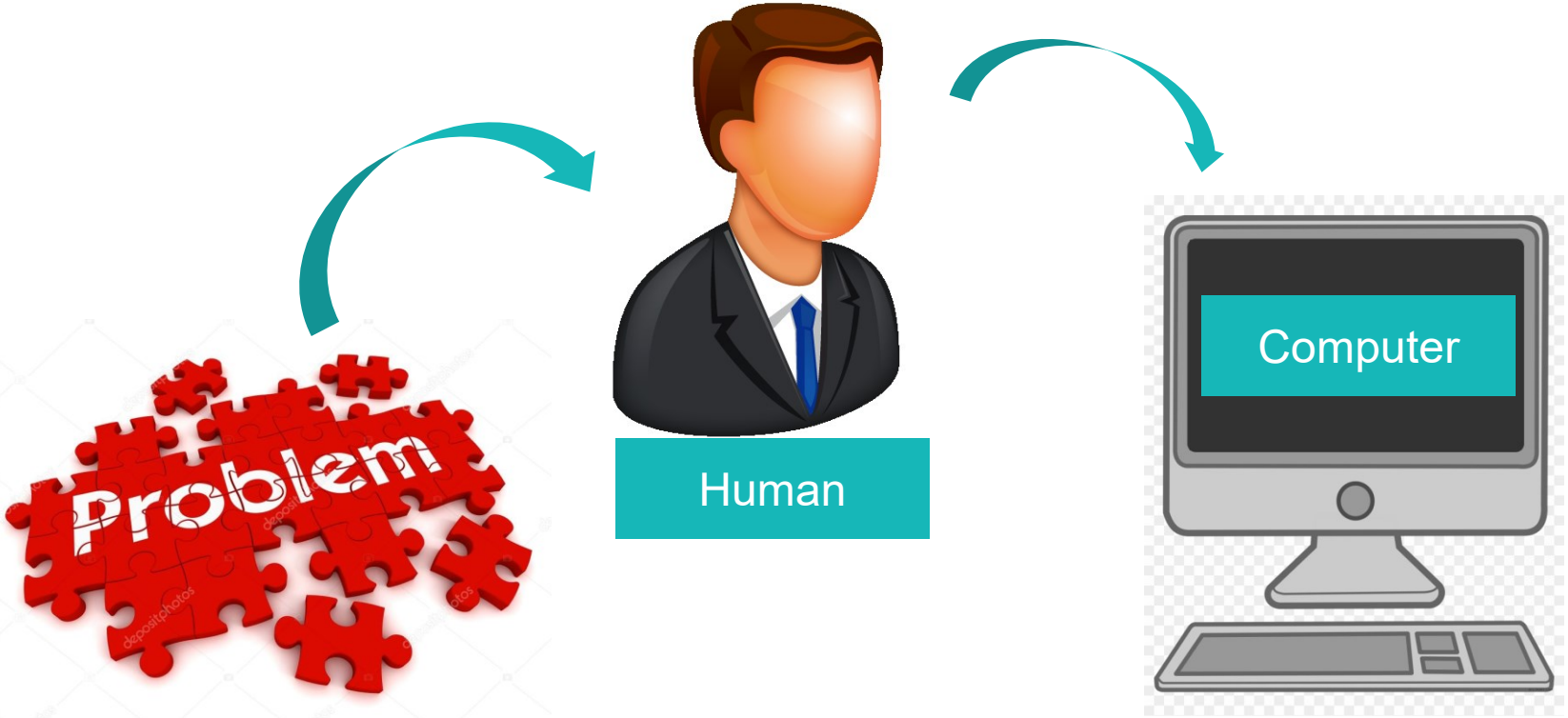
Paradigma Pemrograman

Program Studi S1 Informatika
Universitas Diponegoro
Semester Gasal 2020/2021

Outline

- Pendahuluan
- Paradigma Pemrograman
- Bahasa Pemrograman
- Belajar Memprogram vs Bahasa Pemrograman

Pendahuluan



Pendahuluan (lanj.)

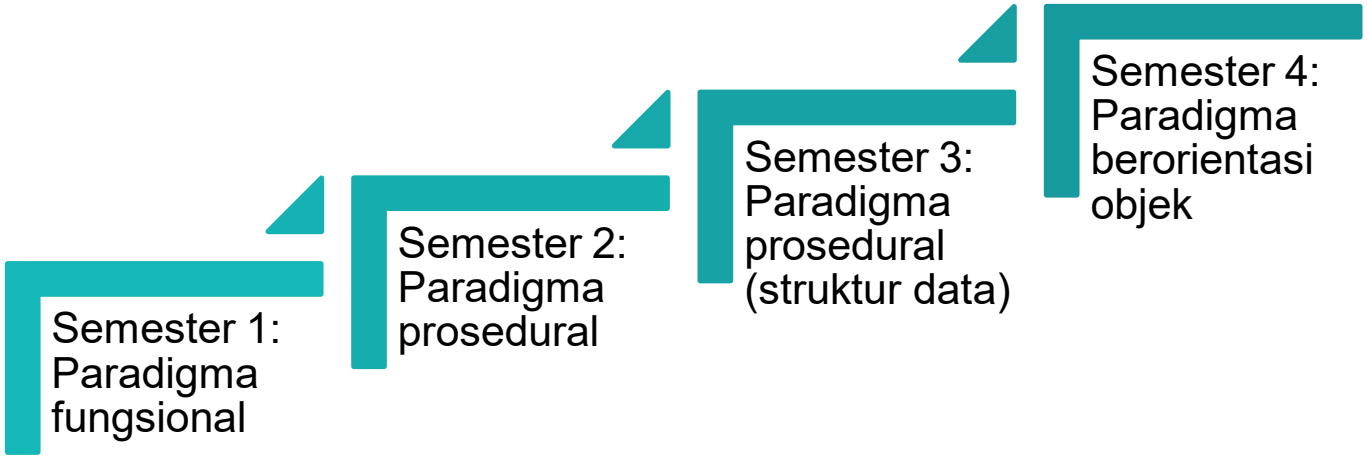
- Komputer digunakan oleh manusia sebagai alat bantu untuk memecahkan suatu permasalahan.
- Manusia tidak dapat “menyodorkan" permasalahannya begitu saja kepada komputer.
- Manusia harus menyusun strategi pemecahan masalah dalam bentuk “program” yang dapat dijalankan oleh komputer.
- Untuk menghasilkan suatu program, seseorang dapat memakai berbagai pendekatan → paradigma.

Paradigma Pemrograman

- ***A style or a "way" of programming.***
- Paradigma mengarahkan jalan berpikir untuk menyelesaikan masalah dalam bentuk program.
- Sebuah paradigma memiliki prioritas strategi analisa yang khusus dalam memecahkan persoalan.
- Satu paradigma tidak akan cocok untuk semua kelas permasalahan.

Paradigma Pemrograman (lanj.)

Mahasiswa informatika harus mendapatkan pengetahuan teoritis dan praktek pemrograman dalam beberapa paradigma agar sudut pandang mahasiswa tidak sempit.



Paradigma Pemrograman (lanj.)

- Paradigma pemrograman prosedural atau imperatif
- Paradigma pemrograman fungsional
- Paradigma pemrograman deklaratif, predikatif atau lojik
- Paradigma pemrograman berorientasi objek
- Paradigma konkuren

Paradigma Prosedural/ Imperatif

- Paradigma ini didasari oleh konsep mesin Von Newmann:
  - Tempat penyimpanan (memori) dibedakan menjadi memori instruksi dan memori data.
  - Masing-masing dapat diberi nama dan nilai.
- Instruksi dieksekusi satu per satu secara sekuensial oleh sebuah pemroses tunggal.
  - Beberapa instruksi menentukan instruksi berikutnya yang akan dieksekusi
- Data diperiksa dan dimodifikasi secara sekuensial.

Paradigma Prosedural/ Imperatif

- Paradigma ini didasari pada **strukturisasi data** dan **manipulasi** terhadap data tersebut.

Algoritma + Struktur Data = Program

- Keuntungan: efisiensi eksekusi karena dekat dengan mesin.
- Kelemahan: tidak “manusiawi” karena harus berpikir dalam batasan mesin.

Paradigma Pemrograman

Paradigma Fungsional

- Didasari oleh konsep pemetaaan dan fungsi pada matematika.
- Fungsi dapat berbentuk: fungsi "primitif“ atau komposisi
- Dasar pemecahan persoalan adalah **transformasional**.
  - Semua perilaku program adalah suatu rantai transformasi dari sebuah keadaan awal menuju keadaan akhir melalui aplikasi fungsi.
- Setiap fungsi adalah "kotak hitam", yang menjadi perhatiannya hanya keadaan awal dan akhir.

Paradigma Fungsional

- Paradigma ini tidak mempernasalahkan memorisasi dan struktur data.
  - Tidak ada pemisahan antara data dan program.
  - Tidak ada istilah "variabel“.
- Kelemahan: program fungsional harus diolah lebih dari program prosedural (oleh pemroses bahasanya)

Paradigma Deklaratif/ *Logic*

- Didasari oleh pendefinisian relasi antar individu yang dinyatakan sebagai predikat orde pertama.
- Pemrogram mendeklarasikan sekumpulan fakta dan aturan-aturan (*inference rules*).
- Ketika program dieksekusi, pemakai mengajukan pertanyaan (*query*).
- Program akan menjawab pertanyaan dengan menulusuri aturan-aturan yang ada dan mencocokkannya dengan fakta-fakta yang ada.

Paradigma Pemrograman

Paradigma Berorientasi Objek

- Paradigma ini didasari oleh kelas dan objek.
- Objek adalah instansiasi dari kelas.
- Objek mempunyai atribut dan metode.
- Sebuah objek dapat berkomunikasi dengan objek yang lain melalui "pesan", dengan tetap terjaga integritasnya.
- Kelas mempunyai hirarki.
- Keunggulan: konsep modularitas, penggunaan kembali, kemudahan modifikasi.
- Masih terkandung paradigma imperatif.

Paradigma Konkuren

- Sebuah sistem komputer harus menangani beberapa proses (task) yang harus dieksekusi bersama dalam sebuah lingkungan (baik mono atau multi prosesor).
- Pemrogram tidak lagi berpikir sekuensial, melainkan harus menangani komunikasi dan sinkronisasi antar task.
- Pada jaman sekarang, aspek konkuren semakin memegang peranan penting.

Bahasa Pemrograman

- **Program** merupakan sekumpulan instruksi yang ditulis mengikuti aturan-aturan pada bahasa pemrograman tertentu.
- **Bahasa pemrograman** → kumpulan kosakata (vocabulary/ symbol) dan grammatical rules yang digunakan untuk memberikan instruksi kepada komputer agar dapat menjalankan task tertentu.
- Sebuah program memiliki syntax dan semantic.
  - **Syntax** merujuk pada penyusunan simbol-simbol sehingga membentuk sebuah kalimat/ program yang sah sesuai aturan pada bahasa pemrograman.
  - **Semantic** merujuk pada makna sebuah kalimat/program.

Generasi Bahasa Pemrograman

1GL	• First Generation Laguage • Bahasa mesin dalam bentuk binary code.
2GL	• Second Generation Language • Low-level languge dalam bahasa assembly
3GL	• Third Generation Language • High-level language, seperti bahasa C.
4GL	• Statement mirip dengan bahasa manusia, contoh: database programming & script
5GL	• Bahasa program yang dilengkapi dengan visual tools, contoh: Visual Basic

Klasifikasi Bahasa Pemrograman

- **Low-level language**
  - Berhubungan langsung dengan hardware → Spesifik terhadap arsitektur dan jenis hardware/ komputer tertentu.
  - Lebih sulit dipahami.
  - Eksekusi lebih cepat dan efisiensi memory.
- **High-level language**
  - Statement program mendekati bahasa manusia (human-friendly)
  - Merupakan abtraksi tingkat tinggi dari bahasa mesin.
  - Tidak berhubungan langsung dengan mesin, digunakan untuk permasalahan yang lebih kompleks.

Bahasa Pemrograman

Low-Level Language

- Machine Language
  - Ditulis langsung untuk berinterkasi dengan mesin/hardware dalam bentuk binary code.
- Assembly language
  - Digunakan untuk berinteraksi langsung dengan mesin dan ditulis dalam kode mnemonics.
  - Menggunakan program khusus, yaitu assembler untuk menerjemahkan kode mnemonic ke dalam binary code.

Bahasa Pemrograman

High-Level Language

- Bahasa ini memerlukan compiler/ interpreter untuk menerjemahkan ke dalam bahasa mesin.
- Interpreter
  - Menerjemahkan langsung setiap statement dalam program ke dalam bahasa mesin.
  - Contoh: Ruby, JavaScript, Python
- Compiler
  - Mengkonversi keseluruhan source code ke dalam object code yang dapat dijalankan oleh mesin.
  - Contoh: C, C++

High-Level Language

- Untuk setiap paradigma, tersedia bahasa pemrograman yang mempermudah implementasi rancangan penyelesaian masalahnya. Contoh:
  - Prosedural: Pascal, Fortran, Basic, Cobol, C , Ada
  - Fungsional: LOGO, APL, LISP, Haskell
  - Deklaratif : Prolog
  - *Object oriented* murni: Smalltalk, Eifel, Java.

Belajar Memprogram vs. Belajar Bahasa Pemrograman

Belajar Memprogram	Belajar Bahasa Pemrograman
Belajar tentang strategi pemecahan masalah, metodologi dan sistematika pemecahan masalah tersebut.	Belajar memakai suatu bahasa, aturan sintaks (tata bahasa).
Belajar memprogram lebih bersifat keterampilan dalam melakukan analisis dan sintesa.	Memanfaatkan instruksi-instruksi dan kiat yang dapat dipakai secara spesifik hanya pada bahasa tersebut.
Menggunakan notasi yang disepakati bersama.	Menggunakan aturan notasi atau sintaks bahasa pemrograman.

Belajar Memprogram vs. Belajar Bahasa Pemrograman

Belajar Memprogram	Belajar Bahasa Pemrograman
Pemecahan persoalan dengan paradigma yang sama akan menghasilkan solusi yang "sejenis".	Pemecahan persoalan dalam satu paradigma dapat diterjemahkan ke dalam bahasa-bahasa yang berbeda.
Proses memprogram adalah proses yang memerlukan kepakaran.	Proses koding lebih merupakan proses semi otomatik dengan aturan pengkodean.
Titik beratnya adalah membentuk seorang perancang "designer" program,	Proses koding lebih merupakan proses semi otomatik dengan aturan pengkodean.

Belajar Memprogram dan Bahasa Pemrograman

- Produk yang dihasilkan oleh seorang pemrogram adalah:
  - program dengan rancangan yang baik (metodologis, sistematis)
  - dapat dieksekusi oleh mesin
  - berfungsi dengan benar
  - sanggup melayani segala kemungkinan masukan
  - didukung dengan adanya dokumentasi.
- Suatu rancangan harus dapat dikode untuk dieksekusi dengan mesin → Belajar memrogram dan bahasa pemrograman saling melengkapi.

Tujuan Belajar Pemrograman Fungsional

- Memecahkan masalah dengan paradigma fungsional tanpa tergantung pada bahasa pemrograman apapun.
- Menulis program berdasarkan pemrograman fungsional menjadi salah satu bahasa pemrograman yang menjadi target.

Mengapa Fungsional?

- Pada hakekatnya, program dibuat untuk melaksanakan suatu fungsi tertentu sesuai dengan kebutuhan pemakai.
- Pemrograman Fungsional (PF) mengajarkan cara berpikir melalui fungsi (apa yang akan direalisasikan).
  - Sebuah program hanyalah aplikasi terhadap sebuah fungsi.
- PF tidak mempedulikan bagaimana memori komputer dialokasi dan diorganisasi.
- PF juga menghiraukan sekuens/urutan instruksi.

Referensi

- Liem, I., 2008, ***Diktat Kuliah Dasar Pemrograman: Bagian Pemrograman Fungsional***, Institut Teknologi Bandung.

Thank you

Selamat Belajar dan Berlatih !!!