|  |  |  |  |
| --- | --- | --- | --- |
|  | Departemen | : | Teknologi Informasi |
| Kode Mata Kuliah | : | IT-304 |
| Bobot SKS | : | 3 |
| Status Revisi | : | 0 |
| Tanggal Efektif | : |  |

RENCANA

PEMBELAJARAN

(Core Course Plan)

NAMA MATA KULIAH

|  |
| --- |
| ELEMEN KONSTRUKSI PERANGKAT LUNAK 1 |

|  |  |  |
| --- | --- | --- |
| Disiapkan oleh | Diperiksa oleh | Disetujui oleh |
|  |  |  |
| Dr. Herry I. Sitepu/  Yoyok Gamaliel, M.Eng | Dr. Herry I. Sitepu | Dr. Ir. Roland Y.H. Silitonga, M.T. |
| Dosen/Dosen Pengampu | Kepala Departemen | Direktur Akademik |

**INSTITUT TEKNOLOGI HARAPAN BANGSA**

**2018**

**2018/2019**

**Core Course Plan**

**IT - 304**

**Elemen Konstruksi Perangkat Lunak 1**

Yoyok Gamaliel, M.Eng

**DEPARTEMEN TEKNOLOGI INFORMASI**

**INSTITUT TEKNOLOGI HARAPAN BANGSA**

**2018**

SEMESTER GANJIL 2018/2019

***(IT-304) Elemen Konstruksi Perangkat Lunak 1***

**KONTEKS MATA KULIAH DALAM *GRADUATE PROFILE***

Matakuliah ini bertujuan untuk mengembangkan:

1. **Kompetensi**: mampu menganalisa permasalahan dan memformulasikan solusi, rancang/bangun *software*, *unit testing*, dan kualitas dengan menggunakan bahasa pemrograman berorientasi obyek Java.
2. **Karakter**: sikap yang berorientasi pada tujuan, serta kemampuan bekerjasama.
3. **Komitmen**: kesadaran dan komitmen untuk melakukan hal-hal yang menambah nilai (*value creating*) di manapun mahasiswa kelak berkarir.

**CAPAIAN PEMBELAJARAN LULUSAN (*CPL*) PROGDI**

|  |  |
| --- | --- |
| **CPL** | **Keterangan** |
| 1. | Mampu merencanakan, mengimplementasikan, dan mengoptimalisasikan pengembangan perangkat lunak, jaringan komputer, telekomunikasi, dan otomasi, sesuai dengan spesifikasi yang dibutuhkan. |
| 2. | Menguasai keilmuan di bidang konstruksi perangkat lunak, jaringan komputer, telekomunikasi, dan otomasi. |
| 3. | Mampu menguji, mengukur, dan menganalisis kinerja sistem komputer, perangkat lunak, jaringan komputer, telekomunikasi, dan otomasi untuk menyajikan solusi terhadap permasalahan. |
| 4. | Bertanggung jawab pada pekerjaan sendiri dan dapat diberi tanggung jawab atas pencapaian hasil kerja organisasi. |
| 5. | Menguasai konsep dan teknik terkait dengan fungsi bisnis (kebijakan dan strategi bisnis, organisasi, SDM, sistem informasi, akuntansi, dan keuangan, pemasaran, operasi bisnis), prinsip kepemimpinan dan kewirausahaan dalam bidang teknologi informasi serta mampu mengevaluasi peluang di bidang teknologi informasi dan merealisasikannya menjadi bisnis teknologi informasi. |
| 6. | Mampu bekerja secara efektif dalam tim dan mampu berkomunikasi secara lisan dan tulisan dengan efektif menggunakan bahasa Indonesia dan Inggris. |
| 7. | Memiliki kesadaran, kemauan dan kemampuan untuk belajar secara mandiri dan berkelanjutan guna meningkatkan kapasitas dan kemampuan untuk menghadapi situasi/ tantangan hidup yang dinamis. |
| 8. | Memiliki sertifikasi profesional dalam bidang jaringan serta pemrograman komputer dari lembaga sertifikasi profesi internasional. |

**SASARAN KULIAH (*LEARNING OUTCOMES*)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CPMK** | **Keterangan** | **CPL PRODI** | | | | | | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |
| 1 | Mampu menganalisis permasalahan dalam kerangka berpikir obyek. | **✔** | **✔** | **✔** | **✔** |  | **✔** | **✔** |  |
| 2 | Mampu merancang dan mengimplementasikan solusi masalah dalam pemrograman berorientasi obyek Java. | **✔** | **✔** | **✔** | **✔** |  | **✔** | **✔** |  |
| 3 | Memahami prinsip *software engineering*, seperti decoupling dan abstraksi, serta *error-handling*. | **✔** | **✔** | **✔** | **✔** |  | **✔** | **✔** |  |
| 4 | Mampu menganalisis dan meningkatkan kualitas perangkat lunak dengan menerapkan *coding style*, *refactoring*, dan *object-oriented design*. | **✔** | **✔** | **✔** | **✔** |  | **✔** | **✔** |  |

**MATERI KULIAH**

Kelas dan obyek; Implementasi kelas dan obyek; Tipe Data; *Control Statements*; Perulangan; *Array* dan *List*; *Input*/ *Output*; Perancangan Kelas; *Inheritance*; *Interface*; *Exception handling* dan *debugger*; *Object-oriented design*; *Coding style* dan *refactoring*; *Software project workshop*.

**DAFTAR PUSTAKA**

1. Cay Horstman. *Java Concepts Early Objects, 7th Edition*. John Wiley & Sons, 2014.
2. Brett D. McLaughlin, Gary Pollice, Dave West. *Head First Object Oriented Analysis and Design, 1st Edition.* O’Reilly, 2016.
3. Tomek Kaczanowski. *Practical Unit Testing with JUnit and Mockito*, 2013.
4. Alan Vermeulen. *The Elements of Java Style*. Cambridge University Press, 2000.
5. Martin Fowler. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley Professional, 2002.

**EVALUASI DAN PENILAIAN**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***Learning Outcomes*** | **Tugas I**  **(10%)** | **Tugas II**  **(15%)** | **Kuis 1**  **(15%)** | **Kuis 2**  **(15%)** | **Tugas Besar**  **(20%)** | **UAS**  **(25%)** |
| Mampu menganalisis permasalahan dalam kerangka berpikir obyek. | **✔** | **✔** | **✔** |  | **✔** | **✔** |
| Mampu merancang dan mengimplementasikan solusi masalah dalam pemrograman berorientasi obyek Java. | **✔** | **✔** | **✔** |  | **✔** | **✔** |
| Memahami prinsip *software engineering*, seperti *decoupling* dan abstraksi, serta *error-handling*. |  | **✔** | **✔** |  |  | **✔** |
| Mampu menganalisis dan meningkatkan kualitas perangkat lunak dengan menerapkan *coding style*, *refactoring*, dan *object-oriented design*. |  | **✔** |  | **✔** | **✔** | **✔** |

**KEHADIRAN**

Minimal 80% sebagai syarat diprosesnya nilai.

**KUIS**

Kuis dilaksanakan tanpa pemberitahuan. Kuis direncanakan akan dilaksanakan setiap pertemuan, di awal atau di akhir pertemuan. Materi kuis adalah materi yang dipelajari di pertemuan sebelumnya atau pada pertemuan tersebut. Jika ada mahasiswa yang berhalangan hadir, tidak akan diadakan kuis susulan.

**TUGAS**

Tugas mingguan ini berupa latihan soal di kelas atau PR. Tugas dapat diberikan setiap pertemuan dan diselesaikan pada saat itu juga. Tugas tersebut ada yang dikerjakan perorangan atau berkelompok.

**PROYEK**

Merancang aplikasi dengan menggunakan Java dengan mempraktekkan *good design, documentation, unit testing*.

**PRAKTIKUM**

Mata kuliah ini disertai dengan kegiatan praktikum di laboratorium. Pelaksanaan praktikum diatur terpisah dan didampingi oleh asisten yang telah ditunjuk.

**JADWAL PERKULIAHAN**

| **MINGGU KE-** | **TOPIK** | **TUJUAN** | **PERSIAPAN**  *(bahan yang harus dibaca mahasiswa sebelum kuliah)* |
| --- | --- | --- | --- |
| 1 | Kelas dan obyek | * Menjelaskan definisi kelas dan obyek. * Mencontohkan kelas dan obyek * Menjelaskan konstruktor dan instansiasi obyek * Menjelaskan *object reference* | Ref 1. Ch 2 |
| 2 | Implementasi kelas dan obyek | * Membangun program Java sederhana * Menggunakan bahasa Java untuk mengimplementasikan kelas * Mengaplikasikan level akses: *public*, *private*, *package* * Mengaplikasikan instansiasi untuk membuat obyek * Menjelaskan penggunaan *local variable* dan *this reference* * Menelusuri masalah dengan *unit testing* | Ref 1. Ch 3,  Ref 3. Ch 2 |
| 3 | Tipe Data | * Menjelaskan tipe data: *primitive* dan *reference type* * Mengaplikasikan *problem solving* dengan *hand tracing* * Menggunakan mekanisme argumen di Java * Mengaplikasikan penggunaan kelas *string* di Java * Menggunakan mekanisme *casting* di tipe data Java | Ref 1. Ch 4 |
| 4 | *Control Statements* | * Menggunakan if-*statement* untuk *problem solving* * Mengaplikasikan mekanisme *comparison*: *value* atau *reference* * Mendeskripsikan *problem solving* dengan *flowchart* * Menerapkan *problem solving* dengan *selecting test cases*: *switch* * Menjelaskan variabel Boolean * Mengoperasikan operasi Boolean | Ref 1. Ch 5. |
| 5 | Perulangan | * Membedakan penggunaan *looping* dengan while, for, atau do * Mengaplikasikan *problem solving* dengan *hand tracing* * Mendeskripsikan *problem solving* dengan *storyboards* * Menyusun algoritma *loop* yang umum digunakan | Ref 1. Ch 6, 7 |
| 6 | Array dan List | * Menggunakan *arrays* * Mengaplikasikan *enhanced loop* pada *array* * Menyusun algoritma *array* yang umum digunakan * Menerapkan *searching* dan *sorting array* * Menerapkan *list* dengan *array list* | Ref 1. Ch 8,  Ref 2. Ch 3,  Ref 3. Ch 3 |
| 7 | *Input/ Output* | * Menggunakan Java I/O untuk baca tulis *file* * Menggunakan *input* dan *output stream* * Menggunakan *command line arguments* |  |
| 8 | Perancangan Kelas | * Merancang kelas dari problem domain * Merancang *method* yang baik * Menjelaskan *abstraction* dan *encapsulation* * Melakukan *unit testing* * Menjelaskan *patterns for object data* * Menjelaskan *package* * Menjelaskan *unit test framework* | Ref 1. Ch 9 |
| 9 | *Inheritance* | * Menjelaskan *inheritance hierarchies* * Mengaplikasikan *subclass* * Menggunakan *method overriding* * Menjelaskan *polymorphism* * Menjelaskan kelas obyek | Ref 1. Ch 10,  Ref 3. Ch 5 |
| 10 | *Interface* | * Menjelaskan *interface* * Menggunakan *interface* dan *callback* * Menggunakan *inner class* * Menjelaskan *mock object* * Menggunakan *event handling* | Ref 1. Ch 11 |
| 11 | *Exception handling dan debugger* | * Menjelaskan konsep *exception* * Menerapkan *exception handling* * Menerangkan mekanisme aplikasi menangani *input error* * Menjelaskan teknik-teknik *debugging* dengan menggunakan *debugger* | Ref 1. Ch 12 |
| 12 | *Object-oriented design* | * Menjelaskan *classes* dan *responsibilities* * Menjelaskan *class relationship* * Menjelaskan *class coupling* * Menerapkan *class diagram* * Menjelaskan *object diagram* * Menjelaskan *dependency model diagram* | Ref 2, Ch 2, 3 |
| 13 | *Coding Style dan refactoring* | * Menerapkan *coding style*: *formatting, naming, documentation* * Menerapkan *refactoring*: prinsip, *code smell, catalog* |  |
| 14 | *Software project workshop* | * Menjelaskan *software project* * Mendiskusikan proposal proyek | Ref. 4, Ref. 5 |
| 15 | **UAS** |  |  |