|  |  |  |
| --- | --- | --- |
| C:\Users\norsu\Desktop\jata negara.png | | |
| **KOD DAN NAMA PROGRAM NOSS** | IT-010-3:2016 PEMBANGUNAN APLIKASI  **BAHAGIAN PENDIDIKAN DAN LATIHAN TEKNIK VOKASIONAL**  **KEMENTERIAN PENDIDIKAN MALAYSIA**  **ARAS 5 & 6, BLOK E14, KOMPLEKS E,**  **PUSAT PENTADBIRAN KERAJAAN PERSEKUTUAN**  **KERTAS PENERANGAN**  ***(INFORMATION SHEET)*** | |
| **TAHAP NOSS** | 3 | |
| **KOD, NAMA CU DAN WA NOSS** | **CU02/WA1 – INTERPRET APPLICATION MODULE DEVELOPMENT REQUIREMENT** | |
| **NAMA PROGRAM KV** | SISTEM PENGURUSAN PANGKALAN DATA DAN APLIKASI WEB | |
| **KOD DAN NAMA KURSUS KSKV** | KPD2033 FUNDAMENTAL OF PROGRAMMING | |
| **NO DAN TAJUK STANDARD KANDUNGAN KSKV** | K1 INTERPRET APPLICATION MODULE DEVELOPMENT  REQUIREMENT | |
| **NO. KOD NOSS** | IT-010-3:2016-C02/P(4/70) | Muka Surat : 1 Drp : 08 |
| **NO. KOD KSKV** | KPD2033/P(2/13) |

**TAJUK/***TITLE***:**

**PENGENALAN KEPADA INTEGRATED DEVELOPMENT ENVIRONMENT (IDE) DAN TEST DRIVEN DEVELOPMENT (TDD)**

**TUJUAN/***PURPOSE* **:**

Kertas penerangan ini adalah bertujuan menerangkan mengenai :

1. Kenalpasti persekitaran IDE.

1. Nyatakan TDD.

3. Fungsi TDD.

**PENERANGAN/***INFORMATION* **:**

1. **PENGENALAN KEPADA *INTEGRATED DEVELOPMENT ENVIRONMENT* (IDE)**

* **IDE ialah aplikasi / software yang digunakan untuk memudahkan kerja-kerja programming.**
* **Fungsi IDE** adalah sebagai:
* Text editor
* Kebolehan untuk compile dan execute program
* Debugger
* Code auto-completion

Integrated Development Environment (IDE) atau lebih dikenali sebagai Pembangunan Persekitaran Bersepadu merupakan aplikasi perisian yang menyediakan kemudahan menyeluruh kepada pengaturcara komputer untuk membuat satu program. IDE biasanya terdiri daripada source code editor build automation tools dan debugger. Kebanyakkan IDE moden mempunyai object-oriented software development(objek berorientasikan pembangunan perisian).

                Sesetengah IDE mengandungi compiler, interpreter atau kedua-duanya, seperti NetBeans dan Eclipse. Terdapat juga IDE yang tidak mempunyai compiler da interpreter seperti SharpDevelop dan Lazarus. Sempadan antara IDE adalah masih kurang jelas. Kadang kala sistem kawalan versi, atau pelbagai alat untuk memudahkan pembinaan Graphical User Interface (GUI) telah disepadukan. Kebanyakkan IDE moden juga mempunyai class browser, object browser dan gambar rajah kelas hierarki, untuk digunakan dalam object-oriented software development (objek berorientasikan pembangunan perisian).

Secara asasnya, IDE memiliki kemudahan seperti;

1. Editor  :  Untuk menulis source code.
2. Compiler  :  Untuk menganalisis sintaks kod yang dibuat dan mengubahnya ke dalam bentuk binary yang sesuai dengan bahasa mesin.
3. Debugger  :  Untuk melakukan percubaan terhadap kod yang telah dibuat.
4. Linker  :  Untuk menggabungkan data binary dari beberapa kod telah dihasilkan oleh compiler sehingga data – data binary menjadi satu dan menjadi satu program komputer yang dapat digunakan.

**CONTOH - CONTOH IDE**

* MS Visual C++ - Bahasa pengaturcaraan yang digunakan adalah C/C++.
* Borland Delphi - Bahasa pengaturcaraan yang digunakan adalah Delphi.
* Netbeans – Bahasa pengaturcaraan yang digunakan adalah Java.
* Eclipse – Bahasa pengaturcaraan yang digunakan adalah Java/PHP .
* Turbo Pascal – Bahasa pengaturcaraan yang digunakan adalah Pascal.

**SEJARAH IDE?**

Pada mulanya, IDE berkembang melalui konsol (console) dan terminal. Sistem awal, bahasa tidak boleh menyokong satu, kerana program telah disediakan dengan menggunakan flow chart (carta alir) dan coding sebelum dihantar kepada compiler. Dartmouth BASIC adalah bahasa pertama yang dibuat dengan menggunakan IDE (dan juga yang pertama direka untuk digunakan pada konsol (console) atau terminal). IDE yang merupakan sebahagian daripada Dartmouth Time Sharing System adalah berdasarkan arahan. Oleh itu, IDE ini tidak sama seperti menu-driven Graphical IDE yang popular. Walau bagaimanapun, IDE ini dilengkapi dengan penyuntingan bersepadu (integrated editing), penyusunan fail (file management), penyusunan (compilation), debugging dan pelaksanaan secara konsisten dengan IDE moden.

Maestro I adalah produk dari Softlab Munich dan dunia pertama pelopor Pembangunan Persekitaran Bersepadu pada 1975 khas untuk software (perangkat lunak). Maestro I telah dipasang (install) untuk 22,000 pengaturcara (programmer) di seluruh dunia. Sehingga tahun 1989, sebanyak 6000 pemasangan telah dibuat di Federal Rebulic of Germany. Maestro I boleh dikatakan pemimpin dunia dalam bidang ini dari tahun 1970-an hingga 1980-an. Pada era ini, Maestro I boleh didapati di Museum of Information Technology di Arlington.

Salah satu IDE pertama yang berkonsepkan plug-in adalah Softbench. Pada tahun 1995, Computerwoche memberi komen tentang penggunaan IDE yang tidak  diterima baik oleh para pembangun.

**Persekitaran Netbean ide.**

* kelebihan bahasa java yang paling utama adalah boleh digunakan hampir disemua Platform, dan juga kerana pengguna Java
* Kelebihan yang ke dua iaitu java merupakan OOP (*Object Oriented Programming) Pengaturcaraan Berorientasi Objek*).

**Persiapan Sebelum Membuat Aplikasi Java**

Untuk memulai belajar javamemerlukan beberapa software yang berguna agar komputer yang gunakan boleh menjalankan aplikasi java iaitu:

1. ***Java Development Kit (JDK)***
2. ***Java Runtime Environtment (JRE)***
3. ***Text Editor.***

Java Development Kit (JDK), berguna untuk agar komputer boleh menjalankan aplikasi dan juga boleh menulis Bahasa pengaturcaraan JAVA.



Rajah 1**:** Logo netbeans ide

* **PENGENALAN KEPADA TEST DRIVEN DEVELOPMENT (TDD)**

**Definisi TDD**

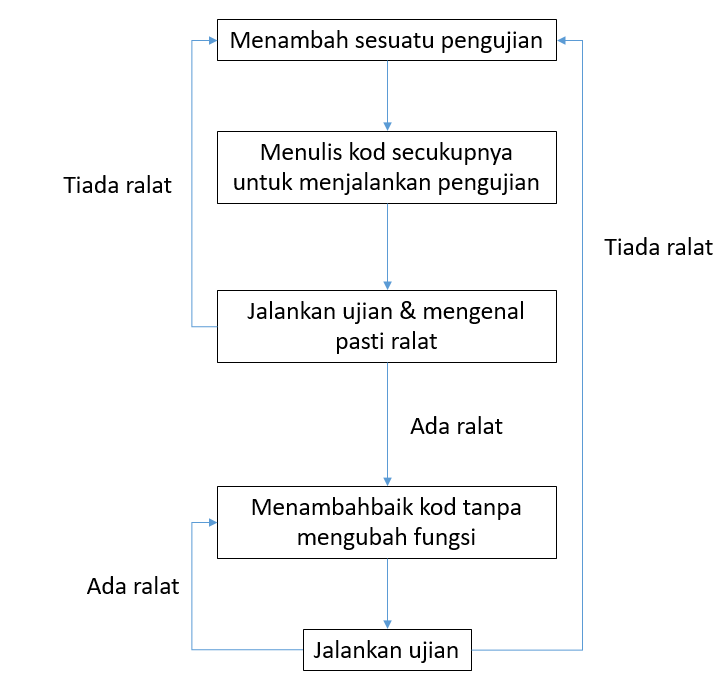
1. **TDD merupakan sebuah proses pembangunan perisian yang bersandarkan kepada pengulangan semula (proses singkat) dalam kitaran pembangunan**

* dimana setiap keperluan (*requirement*) akan ditukarkan kepada *test-case* yang spesifik.
* *test-case* yang lulus sahaja akan dibangunkan.

1. TDD bermula dengan mereka bentuk dan membangunkan *test* untuk setiap fungsi kecil sistem yang bakal di bangunkan.
2. Kebiasaannya, proses *testing* dijalankan dengan menjana kod dan kemudian menguji kod tersebut. Pengujian tersebut mungkin gagal. Dalam usaha untuk lulus test, pasukan pembangunan mempunyai untuk ruang untuk membangunkan semula dan *refactors* kod.
3. Pemfaktoran semula kod bermaksud mengubah beberapa kod tanpa menjejaskan tingkah laku (*behaviour*).
4. Tujuan 🡪TDD memastikan bahawa unit kod sumber sepenuhnya diuji dan membawa kepada modularized, fleksibel dan kod *extensible*. Ia memberi tumpuan kepada hanya menulis kod yang diperlukan untuk lulus ujian, menjadikan reka bentuk mudah dan jelas

**Aktiviti TDD:**

1. Menambah sesuatu pengujian
2. Jalankan test. Jika berjaya, tambah pengujian baharu.
3. Jika gagal pengujian, buat perubahan pada kod tanpa merubah pada fungsi 🡪 Refactoring
4. Menjalankan pengujian. Jika berjaya, tambah pengujian baharu



Rajah 2: Rajah aktiviti TDD

**FUNGSI TDD**

* Menghasilkan aplikasi berkualiti tinggi dalam masa yang kurang daripada yang mungkin dengan kaedah yang lebih tua.
* Pelaksanaan yang betul TDD memerlukan pemaju dan penguji untuk menjangka dengan tepat bagaimana aplikasi dan ciri-cirinya akan digunakan dalam dunia sebenar.
* TDD mencipta suite regresi test sebagai kesan sampingan yang boleh meminimumkan test manual manusia, sementara mencari masalah sebelum ini, yang membawa kepada perbaikan lebih cepat.
* Pemberitahuan ralat.

Pemaju menguji kod mereka tetapi dalam dunia pangkalan data, ini selalunya terdiri daripada test manual atau skrip sekali sahaja. Menggunakan TDD anda membina, sepanjang masa, satu set test automatik yang anda dan mana-mana pemaju lain boleh menjalankan semula sesuka hati.

* Direka dengan lebih baik, bersih dan kod yang lebih boleh dilanjut.

Ia membantu untuk memahami bagaimana kod akan digunakan dan bagaimana ia berinteraksi dengan modul lain. Ia menyebabkan keputusan reka bentuk yang lebih baik dan kod yang lebih dikekalkan. TDD membolehkan menulis kod yang lebih kecil yang mempunyai tanggungjawab tunggal dan bukannya prosedur monolitik dengan pelbagai tanggungjawab. Ini menjadikan kod lebih mudah untuk difahami. TDD juga memaksa untuk menulis hanya kod pengeluaran untuk lulus test berdasarkan keperluan pengguna.

**SOALAN/***QUESTION* **:**

1. Nyatakan maksud IDE.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Berikan **empat**(4) ciri – ciri kemudahan yang terdapat dalam IDE.

i.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ii.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

iii.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

iv\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Nyatakan maksud ***Test Driven Development.***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Nyatakan **tiga**(3)fungsi TDD.

i.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ii.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

iii.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**RUJUKAN/***REFERENCE* **:**

1. Keith Cooper, Linda Torczon(2011), Engineering a Compiler (2nd Edition), Morgan Kaufmann, ISBN: 978-0120884780
2. Todd Zaki Warfel (2009), Prototyping: A Practitioner's Guide, Louis Rosenfeld Media, LLC, ISBN I-933820-21-7
3. Steve McConnell (2004), Code Complete: A Practical Handbook of Software Construction, (2nd Edition),Microsoft Press, ISBN: 978-0-7356-1967-8
4. Michael L. Scot (2009) , Programming Language Pragmatics, Third Edition, Morgan Kaufmann, ISBN-13: 978-0123745149
5. Simon Marlow (2013), Parallel and Concurrent Programming in Haskell: Techniques for Multicore and Multithreaded Programming (1st Edition), O'Reilly Media, ISBN-13: 978-1449335946