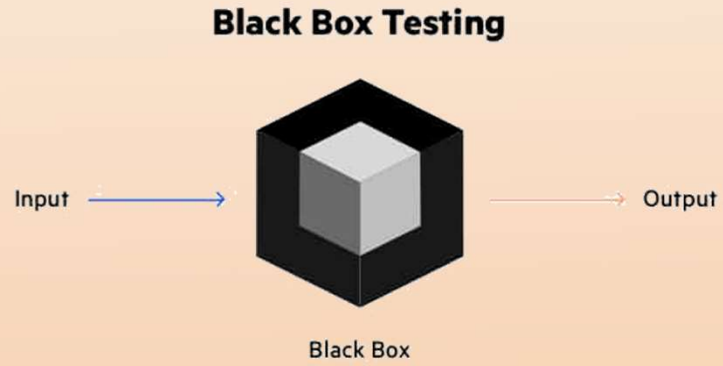
#02
Pengujian Black-Box

PAIK6602 (3 sks)

UJI PERANGKAT LUNAK

Nurdin Bahtiar, MT

Pendahuluan



- Pengujian black-box berfokus pada persyaratan fungsional perangkat lunak.

  Dengan kata lain, pengujian harus dapat ditelusuri sampai ke persyaratan pelanggan. Meskipun pelanggan hanya tahu sebatas yang ia minta saja.

- Pengujian black-box bukan alternatif lain dari white-box, tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan daripada metode white-box.

  Bisa jadi pengujian white-box dilakukan terlalu mendalam sehingga melupakan bagian yang luar / lebih umum.

Pendahuluan

Pengujian ini berusaha mengungkap kesalahan dalam kategori:

- Fungsi-fungsi yang tidak benar atau hilang

  Misalnya salah dalam penggunaan tombol **close** dan **cancel**

- Kesalahan interface

  Misalnya salah menggunakan **radio button** dan **check box**

- Kesalahan dalam struktur data atau akses database eksternal

  Misalnya salah menggunakan tipe **integer** dan **real**

- Kesalahan kinerja

  Misalnya salah menghitung jumlah, rata-rata, dan sebagainya

- Inisialisasi dan kesalahan terminasi

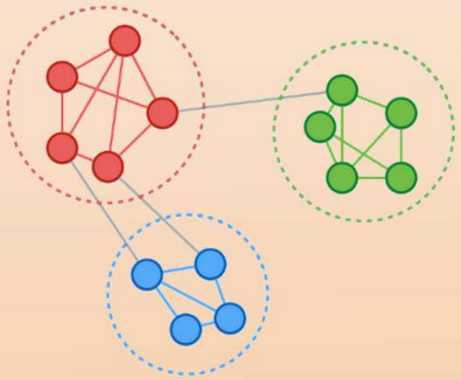
  Misalnya salah dalam memasukkan nilai awal atau mengeluarkan nilai akhir.

Pendahuluan

- Pengujian black-box menguji beberapa aspek dasar suatu sistem dengan sedikit memperlihatkan struktur logika internal perangkat lunak tersebut.
- Beberapa teknik pengujian black-box di antaranya:

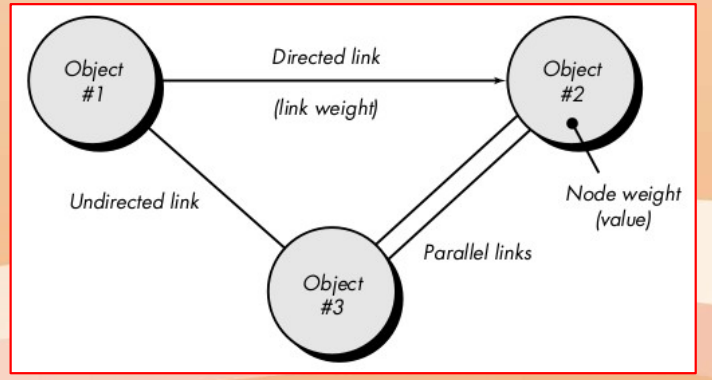
1. Metode Pengujian Graph-Based
2. Partisi Ekivalensi
3. Analisis Nilai Batas
4. Pengujian Perbandingan

1. Metode Pengujian Graph-Based



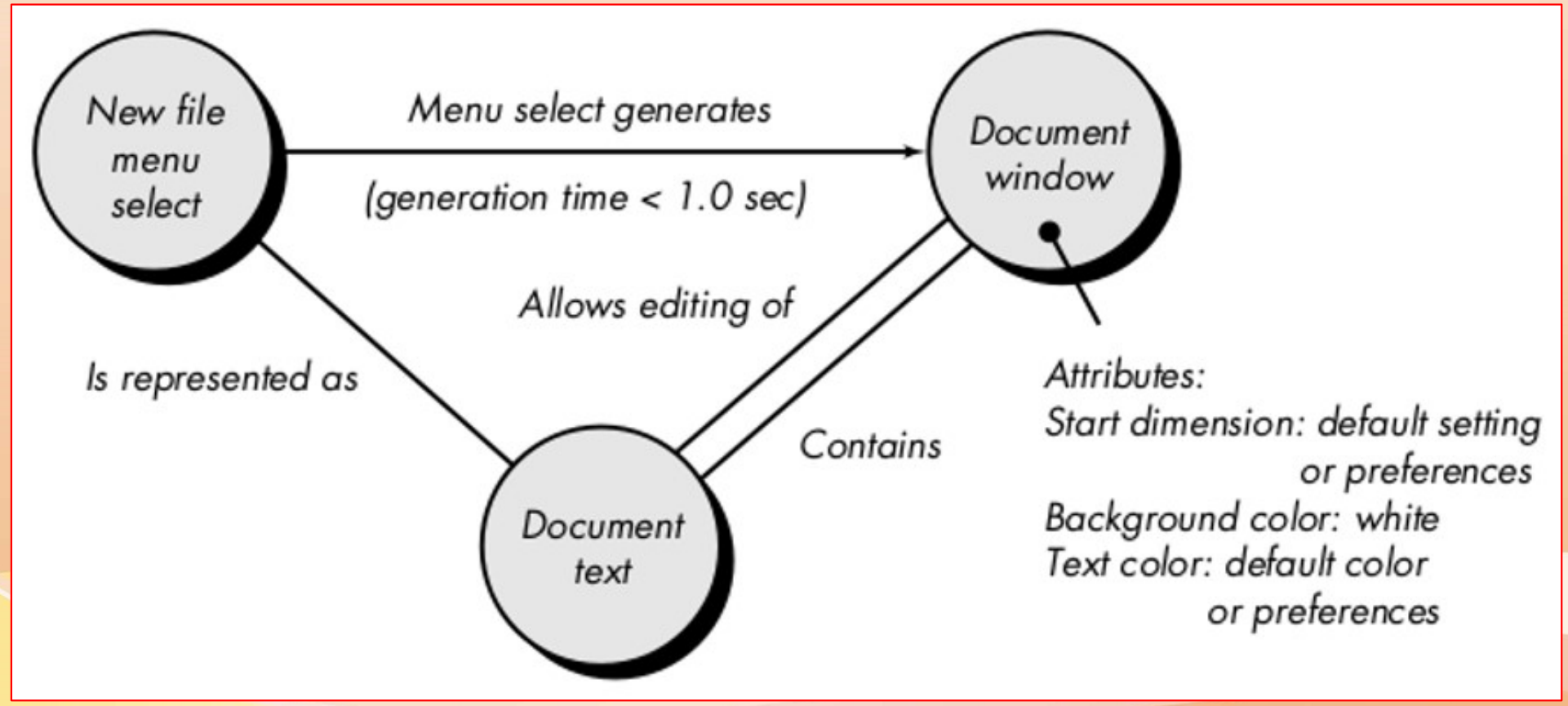
- Langkah pertama dalam pengujian black-box adalah memahami objek yang dimodelkan di dalam perangkat lunak dan hubungannya.
- Kemudian menentukan sederetan pengujian yang membuktikan bahwa “Semua objek memiliki hubungan yang diharapkan satu dengan yang lain”.

- ***Simpul*** merepresentasikan suatu objek (seperti modul atau koleksi statemen).
- ***Link*** merepresentasikan hubungan antar objek.
- ***Node weight*** menggambarkan properti simpul (atribut).
- ***Link weight*** menggambarkan beberapa karakteristik link.

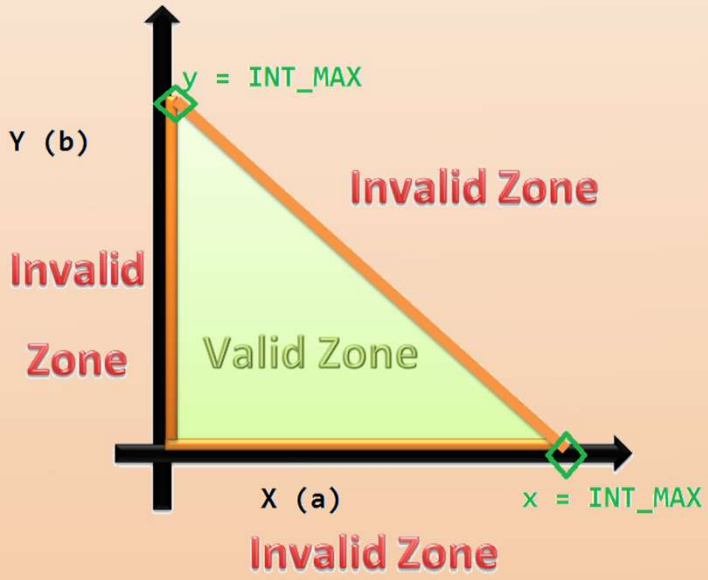


1. Metode Pengujian Graph-Based

Contoh sederhana:



2. Partisi Ekivalensi



- Merupakan teknik pengujian yang membagi domain input dari suatu program ke dalam kelas data dari mana test case dapat dilakukan.
- Kadang dinamakan *equivalence classing*.
- Kelas ekivalensi merepresentasikan serangkaian keadaan valid atau invalid untuk kondisi input. Kondisi input dapat berupa harga numeris, rentan harga, serangkaian harga khusus, atau kondisi boolean.

2. Partisi Ekivalensi

Beberapa petunjuk dalam kelas ekivalensi:

- Bila kondisi input suatu *range*, satu kelas ekivalensi valid dan dua invalid ditentukan.

  Misalnya dengan memasukkan angka -1.2, 2.0, dan 4.01 pada masukan IPK.

- Bila kondisi input suatu *harga khusus*, satu kelas ekivalensi valid dan dua invalid ditentukan.

  Misalnya dengan memasukkan huruf A, F, dan X pada masukan nilai huruf (A, B, C, D, dan E).

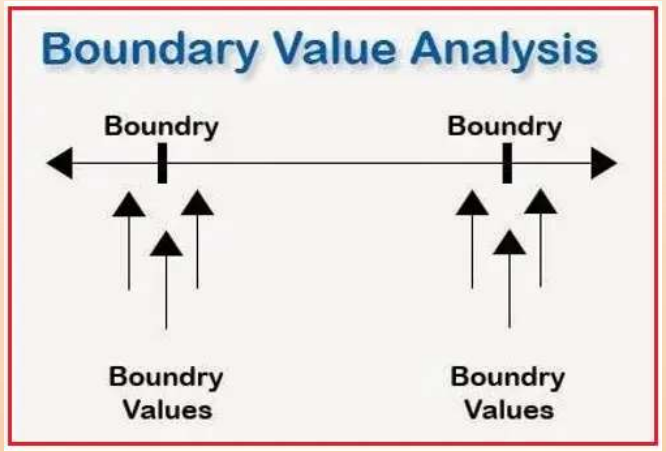
- Bila kondisi input anggota suatu *himpunan*, satu kelas ekivalensi valid dan satu invalid ditentukan.

  Misalnya dengan memasukkan input yang tidak terdaftar (pada himpunannya).

- Bila kondisi input *boolean*, satu kelas ekivalensi valid dan satu invalid ditentukan.

  Misalnya pada masukan YES atau NO, coba dengan YES atau YA.

3. Analisis Nilai Batas



- Teknik pengujian *Boundary Value Analysis* (BVA) lebih mengarah pada pemilihan test case pada "**edge**" dari kelas.
- Test case yang dilakukan berfokus pada domain output.

- Bila kondisi input suatu *range a dan b*, test case didisain persis di atas dan di bawah *a* dan *b* secara bersesuaian. Misalnya untuk memasukkan nilai IPK, dapat diuji dengan memasukkan nilai -0.01, 0.01, 3.99, atau 4.01
- Bila kondisi input suatu *harga khusus*, nilai yang tepat di atas dan di bawah minimum juga harus diuji. Bila suatu kondisi input mengkhususkan sejumlah nilai, maka test case harus dikembangkan dengan menggunakan jumlah minimum dan maksimum. Nilai tepat di atas dan di bawah minimum dan maksimum juga diuji.

3. Analisis Nilai Batas

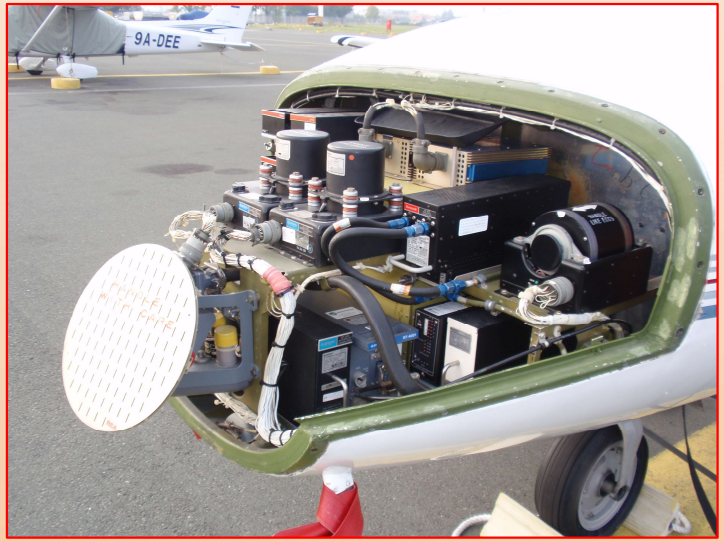
- Mengaplikasikan dua pedoman di atas ke kondisi output, output harus menghasilkan jumlah minimum (dan maksimum) dari entri tabel yang diijinkan.

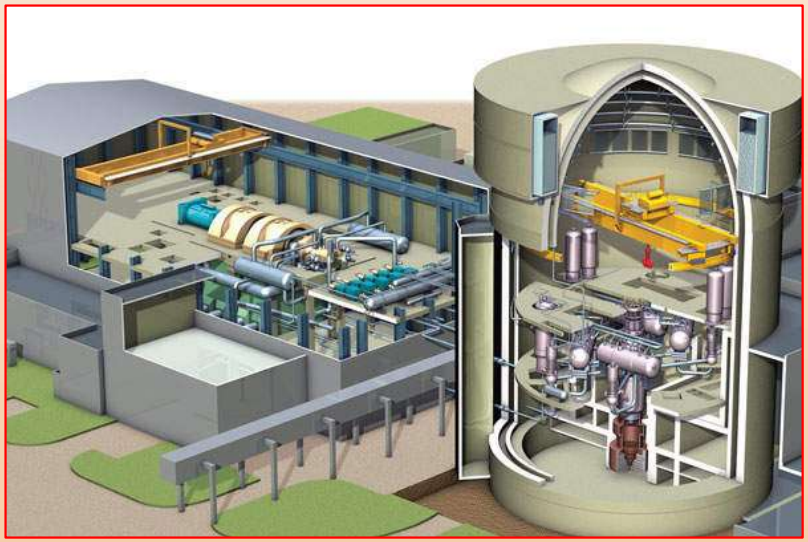
  Misalnya dengan memberi nilai E atau A semua pada transkrip nilai, sehingga IPK akan menjumpai batas maksimum dan minimumnya.

- Bila kondisi input suatu *batasan*, test case didisain menggunakan struktur pada batasannya.

  Bila struktur data program telah memesan batasan (misal, suatu array memiliki suatu batas yang ditentukan dari 100 entri), maka pastikan untuk mendesain test case yang menggunakan struktur data pada batas tersebut.

4. Pengujian Perbandingan





- Ada banyak situasi (seperti avionik pesawat udara, kontrol sumber tenaga nuklir) reliabilitas perangkat lunak sangat kritis.
- Kebutuhan pengujian harus dilakukan dengan teliti menggunakan perangkat duplikat untuk menghindari hal yang tidak diinginkan.

4. Pengujian Perbandingan

- Implementasi berupa pengembangan *hardware* dan *software* redundan untuk meminimalkan kemungkinan kesalahan.
  - ✓ Tim RPL terpisah mengembangkan versi-versi independen dari suatu aplikasi dengan menggunakan spesifikasi yang sama
  - ✓ Setiap versi diuji dengan data uji yang sama untuk memastikan bahwa semua versi memberikan output yang identik
  - ✓ Semua versi dieksekusi secara paralel dengan perbandingan real time hasil untuk memastikan konsistensi
  - ✓ Pengujian ini disebut juga *Pengujian Back-to-Back*.

Resume

Advantages of Black Box Testing:

- Penguji tidak harus memahami secara teknis.
- Pengujian dilakukan hanya seperti mencari bug, dimana pengguna biasa pun dapat melakukannya.
- Pengujian dapat membantu mengidentifikasi ketidakjelasan dan kekontradiksian pada spesifikasi fungsional.
- Test case dapat segera didisain setelah spesifikasi fungsional telah terpenuhi.

Resume

Disadvantages of Black Box Testing:

- Kemungkinan melakukan perulangan dari pengujian yang telah dilakukan programmer.
- Input test yang dibutuhkan dapat berasal dari ruang contoh yang luas.
- Susah untuk mengidentifikasi semua kemungkinan masukan jika waktunya terbatas, sehingga menentukan kasus ujinya menjadi lambat dan rumit.
- Kemungkinan menjumpai jalur yang tidak teridentifikasi selama pengujian.

End of File

Latihan

1. Sebutkan 3 (tiga) SRS dari aplikasi yang pernah Anda kembangkan!
2. Jelaskan 5 (lima) test case yang bisa diterapkan masing-masing SRS tersebut!