Black box testing

What is Black Box testing?
• Testing without having an insight into the details of underlying code

$\text{input} \rightarrow \text{Unit} \rightarrow \text{output}$

• Testing based on specification

What is Black Box testing?
• Ignores internal mechanism of a system.
• Perform output correctness tests
• Focuses on output generated (responses to selected input and execution conditions)
• Conducted to evaluate the compliance of a system or component with specified functional requirements.
• Identifies bugs only according to software malfunctions as they are revealed in its erroneous outputs.

Error Guessing
• Adhoc approach based on experience
• Approach:
  1. Make list of possible errors or error-prone situations $\rightarrow$ error model
  2. Design test-cases
• to cover error model
• Develop and maintain your own error models
### Error Guessing

- Example sorting array function
- Error model:
  - empty array
  - already sorted array
  - reverse-sorted array
  - large unsorted array
  - ...
- Generate test cases for these situations

### Equivalence Classes (EC) Partitioning

- A set of input variable values that produce the same output results or are processed identically
- Aim at increasing the efficiency of testing
- Improve coverage of potential error conditions

### Test cases and boundary values

- Defined by a single/group/range of numeric or alphabetic value. In cases where a program’s input is provided by several variables, valid and invalid ECs should be defined for each variables.
- e.g. Instead of dealing with, a set of ‘ages’ from (0-140),
- - create a set of ‘male insurance rate calculation and application age’ equivalence classes are: {young(0-12), teen(13-19), adult(20-35),mature(36-64),retired(65-140)}.

- Errors tend to occur near extreme values (boundaries)
- If EC cover a range of values (e.g. monthly income, apartment area), testing border values should be considered.
- e.g three test cases: mid range, lower boundary and upper boundary values.
### Entrance ticket price table - The Pool

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Visitor’s status</td>
<td>18.00-24.00</td>
<td>18.00-24.00</td>
<td>18.00-24.00</td>
<td>18.00-24.00</td>
<td>18.00-24.00</td>
<td>18.00-24.00</td>
<td>18.00-24.00</td>
</tr>
<tr>
<td>Entry hour</td>
<td>8.00-19.00</td>
<td>8.00-19.00</td>
<td>8.00-19.00</td>
<td>8.00-19.00</td>
<td>8.00-19.00</td>
<td>8.00-19.00</td>
<td>8.00-19.00</td>
</tr>
<tr>
<td>Age: 6.00-16.00</td>
<td>$5</td>
<td>$6</td>
<td>$2.50</td>
<td>$3</td>
<td>$7.50</td>
<td>$9</td>
<td>$3.50</td>
</tr>
<tr>
<td>Age: 16.01-18.00</td>
<td>$10</td>
<td>$17</td>
<td>$5</td>
<td>$6</td>
<td>$15</td>
<td>$18</td>
<td>$7</td>
</tr>
<tr>
<td>Age: 19.01-24.00</td>
<td>$8</td>
<td>$8</td>
<td>$4</td>
<td>$4</td>
<td>$12</td>
<td>$12</td>
<td>$5.50</td>
</tr>
</tbody>
</table>

### Test cases - The ticket price module

<table>
<thead>
<tr>
<th>Test case type</th>
<th>Test case no.</th>
<th>Day of week</th>
<th>Visitor’s status</th>
<th>Entry hour</th>
<th>Visitor’s age</th>
<th>Test case result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid ECs</td>
<td>1</td>
<td>Mon.</td>
<td>Off</td>
<td>7.30</td>
<td>9.4</td>
<td>$5</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Sat.</td>
<td>Mem</td>
<td>22.44</td>
<td>65.0</td>
<td>$5.50</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Sat.</td>
<td>Mem</td>
<td>8.30</td>
<td>0.0</td>
<td>$5.50</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Sat.</td>
<td>Mem</td>
<td>19.00</td>
<td>19.0</td>
<td>$5.50</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Sat.</td>
<td>Mem</td>
<td>19.01</td>
<td>6.01</td>
<td>$8</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>Sat.</td>
<td>Mem</td>
<td>19.01</td>
<td>6.01</td>
<td>$8</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>Sat.</td>
<td>Mem</td>
<td>19.01</td>
<td>6.01</td>
<td>$8</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>Sat.</td>
<td>Mem</td>
<td>24.00</td>
<td>93.01</td>
<td>$5.50</td>
</tr>
<tr>
<td>Invalid ECs</td>
<td>9</td>
<td>Sat.</td>
<td>Mem</td>
<td>24.00</td>
<td>125.0</td>
<td>$5.50</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>Mon.</td>
<td>Off</td>
<td>7.30</td>
<td>9.4</td>
<td>Invalid day</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Mon.</td>
<td>Off</td>
<td>7.30</td>
<td>9.4</td>
<td>Invalid visitor’s status</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Mon.</td>
<td>Off</td>
<td>4.46</td>
<td>9.4</td>
<td>Invalid entry hour</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>Mon.</td>
<td>Off</td>
<td>4.46</td>
<td>9.4</td>
<td>Invalid entry hour</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>Mon.</td>
<td>Off</td>
<td>7.30</td>
<td>115.1</td>
<td>Invalid visitor’s status</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>Mon.</td>
<td>Off</td>
<td>7.30</td>
<td>115.1</td>
<td>Invalid visitor’s status</td>
</tr>
</tbody>
</table>

### Operation factor testing classes

<table>
<thead>
<tr>
<th>Quality requirements factor</th>
<th>Test class</th>
</tr>
</thead>
</table>
| Correctness                 | 1. Documentation tests  
                            | 2. Availability (reaction time) tests |
| Reliability                 | Reliability tests |
| Efficiency                  | Stress tests (load and durability tests) |
| Integrity                   | Software system security tests |
| Usability                   | 1. Training usability tests  
                            | 2. Operational Usability tests |

### Correctness - Documentation Tests

- Erroneous user manual/programmer manual can lead to mistakes during program operation and maintenance.
- Completeness check
- Correctness tests
- Document style and editing inspection
Correctness - Availability tests

- Reaction time - Time required to obtain requested information or for firmware installed in computerized equipment to react.

Reliability Tests

- Deals with features that can be translated as events occurring over time.
- e.g – average time for recovery after system failure (15 minutes).
  - average downtime per month (30 minutes per month)
- Constraint for reliability test is the scope of resources required.

Efficiency - Stress Test

- **Load test** – functional performance of system under maximal operational load. e.g: maximal transaction per minute, hits per minute to an Internet site.
- **Durability test** – carried out in physically extreme operating conditions. e.g: high temperatures, humidity.

Integrity – Security tests

- Aim at preventing unauthorized access, detection of unauthorized access, recovery of damages caused by unauthorized penetration cases.
- **Issues:**
  - access control (password, firewall systems)
  - backup of database and software files
  - logging of transactions, system usage, access trials
Usability – Training tests

• Resources needed to train new employee. e.g: how many hours needed to train new employee to achieve a defined level of acquaintance with the system.
• Result – a sophisticated plan of training courses and improved directions for software system operation.

Usability – Operational usability tests

• Focus on operator’s productivity.
• Can be performed manually by means of time studies.
• Provide insight into performance levels and initiate ideas for improvements.

Revision factor testing classes

• To assure software package successful, long service and successful sales to larger user populations.
  - Maintainability tests
  - Flexibility tests
  - Testability tests

Maintainability tests

• System structure abides by standards and development procedures.
• Programmers manual prepared according to approved documentation standards
• The internal documentation incorporated in the software code is prepared according to coding procedures and conventions.
Flexibility tests

• Efforts required to adapt the software to the variety of customer needs, changes initiated by customers and maintenance teams
• System capabilities based on structural and programming characteristics.
• E.g: adequate modular structure.

Testability tests

• Relates to the addition features in the program that help testers in their work.
• e.g: possibility of obtaining intermediate results for certain checkpoints and predefined log files.

Transition factor testing classes

- Portability tests — specify the environment in which the system has to be operable.
- Reusability tests — defines which part of program are to be developed for future reuse.
- Tests for interoperability requirements:
  - software interfacing tests — deals with software capabilities of interfacing equipment and other software packages.
  - equipment interfacing tests — deals with equipment’s firmware interfacing other software packages.

Advantages of Black Box Testing

• Allows us to carry out the majority of testing classes, most of which can be implemented solely by black box tests, i.e. load tests and availability tests.
• For testing classes that can be carried out by both white and black box tests, black box testing requires fewer resources.
Disadvantages of Black Box Testing

- Possibility that coincidental aggregation of several errors will produce the correct response for a test case, and prevent error detection.
- Absence of control of line coverage. There is no easy way to specify the parameters of the test cases required to improve coverage.
- Impossibility of testing the quality of coding and its strict adherence to the coding standards.