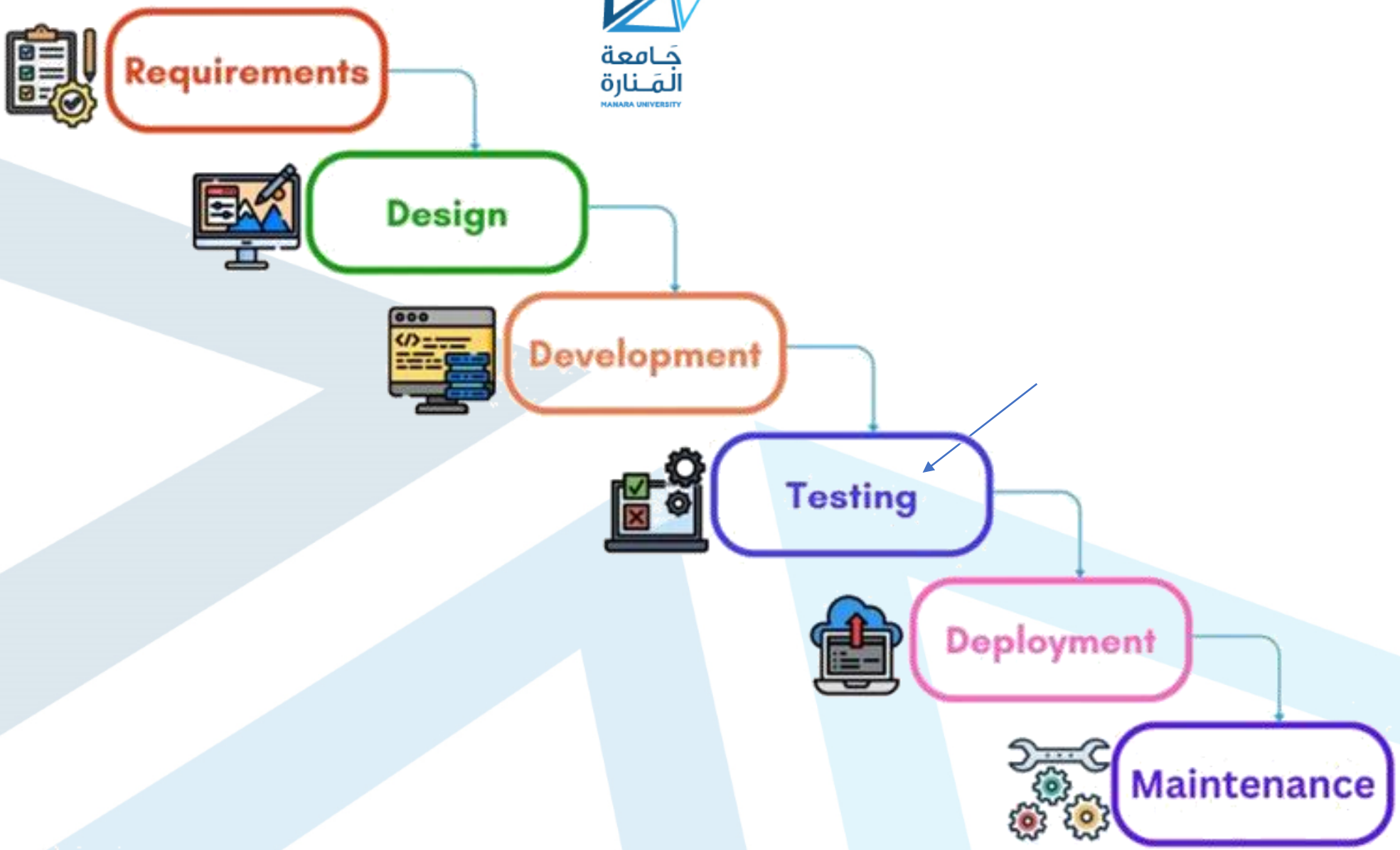
Software Engineering -2-

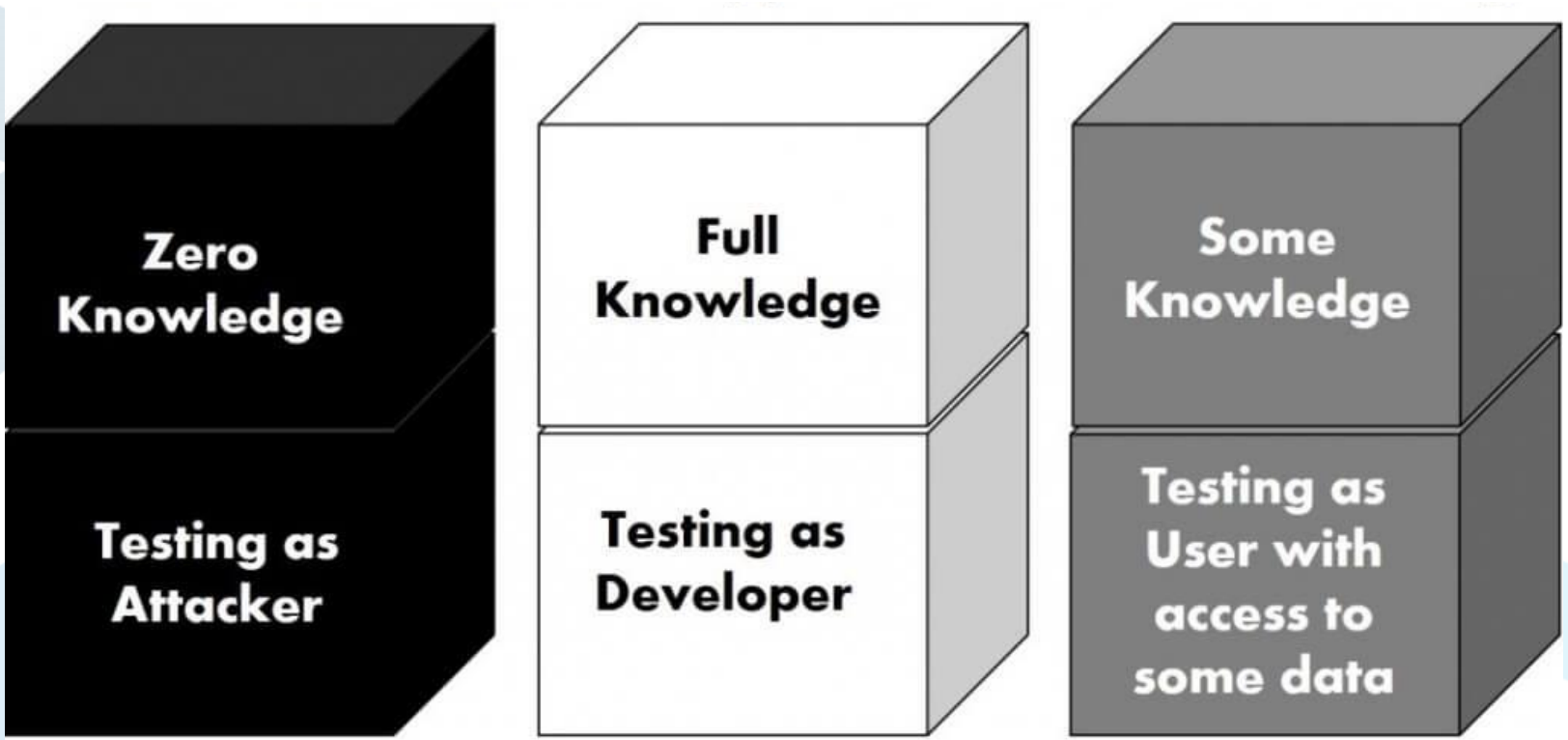
Lecture -6-

Dr. Inas Laila


Requirements
Design
Development
Testing
Deployment
Maintenance

Test Technique

- White Box
- Black Box
- Gray Box



White Box

- Testing technique based on knowledge of the internal logic of an application's code and includes tests like coverage of code statements, branches, paths, conditions.
- also known as structural or glass box testing or structure based testing
- typically undertaken at Component Test phases by development teams

اختبار يتم مع معرفة كاملة بالكود الداخلي للنظام

المختبِر يعرف الكود البرمجي والخوارزميات والمسارات الشرطية

يقوم به المبرمج أو مهندس اختبار لديه وصول للكود

ماذا يتم اختباره؟

- شروط if / else
- الحلقات (loops)
- مسارات التنفيذ
- تغطية الكود (Code Coverage)

- **Black Box**
  - A method of software testing that verifies the functionality of an application without having specific knowledge of the application's code/internal structure
  - Tests are based on requirements and functionality.
  - WHAT a system does, rather than HOW it does it
  - also known as specification based testing
  - applies for Functional and Non-Functional testing

اختبار يتم **دون** أي **معرفة بالكود البرمجي او البنية الداخلية**
ويتم التركيز هنا فقط على:
- المدخلات
- المخرجات
- المتطلبات

يقوم به مختبر النظام -المستخدم النهائي - QA Engineer

المزايا:

✓ لا يحتاج معرفة بالكود

✓ يشبه استخدام المستخدم الحقيقي

✓ جيد لاكتشاف أخطاء المتطلبات

العيوب:

لا يضمن تغطية كل الكود

يصعب معرفة سبب الخطأ

مثال:

نظام تسجيل دخول:

- إدخال اسم مستخدم وكلمة مرور صحيحة تؤدي الى دخول
- ادخال كلمة مرور خاطئة يؤدي رسالة خطأ

Gray Box

Gray Box Testing (اختبار الصندوق الرمادي)

هو مزيج بين **White Box** و **Black Box**

المختبر لديه **معرفة جزئية** عن النظام فهو يعرف

- مخططات
- قواعد بيانات
- هيكل عام للنظام

من يقوم بهذا الاختبار؟

•مختبر لديه اطلاع جزئي على التصميم

◆ **ماذا يتم اختباره؟**

•التكامل بين الوحدات

•التعامل مع قواعد البيانات واختبار الثغرات الأمنية

مثال: اذا كان المختبر يعرف أن النظام يستخدم قاعدة بيانات SQL يمكنه اجراء اختبار SQL Injection دون رؤية الكود

الخاصية	White Box	Black Box	Gray Box
معرفة الكود	كاملة	لا يوجد	جزئية
من ينفذه	المبرمج	المختبر / المستخدم	مختبر متقدم
التركيز	المنطق الداخلي	الوظائف	التكامل والأمن
تغطية الكود	عالية جدًا	منخفضة	متوسطة
شائع في	Unit Testing	System Testing	Security / Integration

- **White Box** →اختبار الوحدات Unit Testing
- **Black Box** → اختبار النظام واختبارالقبول System / Acceptance
- **Gray Box**→اختبار الأمان والتكامل

Black Box Techniques

- Based on requirements(From the requirements, tests are created)

- **Techniques**
  - Equivalence Partitioning
  - Boundary Value Analysis
  - Decision Tables
  - State Transition Testing
  - Use Case Testing

في **اختبارات الصندوق الأسود** تُستخدم عدة **تقنيات** لتصميم حالات الاختبار بشكل فعّال.
فيما يلي شرح واضح ومختصر لكل تقنية من التقنيات ولكن لنبدأ بداية بتعريف حالة الاختبار

تعريف حالة الاختبار (Test Case) في هندسة البرمجيات:

حالة الاختبار هي **مجموعة من الشروط والخطوات والمدخلات والنتائج المتوقعة** تُستخدم للتحقق من أن جزءًا معيّنًا من النظام البرمجي يعمل كما هو مطلوب وفق المتطلبات المحددة.

بعبارة أبسط: حالة الاختبار تصف *كيف نختبر وظيفة معيّنة وما النتيجة الصحيحة المتوقعة* عند تنفيذ هذا الاختبار.

it helps the tester to detect defects in the application

عناصر حالة الاختبار الأساسية

عادةً تتكوّن حالة الاختبار من العناصر التالية:

١. **معرّف حالة الاختبار (Test Case ID)** وهو رقم أو رمز فريد لتمييز حالة الاختبار.
٢. **وصف حالة الاختبار** شرح مختصر لما يتم اختباره.
٣. **الشروط المسبقة Preconditions** الحالة التي يجب أن يكون عليها النظام قبل تنفيذ الاختبار.
٤. **المدخلات (Test Data / Inputs)** البيانات المستخدمة أثناء الاختبار.
٥. **خطوات التنفيذ (Test Steps)** الخطوات التفصيلية لتنفيذ الاختبار.
٦. **النتيجة المتوقعة (Expected Result)** السلوك الصحيح المتوقع من النظام.
٧. **النتيجة الفعلية (Actual Result)** ما حدث فعليًا بعد تنفيذ الاختبار.
٨. **حالة الاختبار (Pass / Fail)** هل نجح الاختبار أم فشل.

Test Case Example1 (simple test)

Test Case #: 2.2 **Test Case Name:** Change PIN **Page:** 1 of 1

System: ATM **Subsystem:** PIN

Designed by: ABC **Design Date:** 28/11/2004

Executed by: **Execution Date:**

Short Description: Test the ATM Change PIN service

Pre-conditions

The user has a valid ATM card - The user has accessed the ATM by placing his ATM card in the machine
The current PIN is 1234
The system displays the main menu

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Click the 'Change PIN' button	The system displays a message asking the user to enter the new PIN		
2	Enter '5555'	The system displays a message asking the user to confirm (re-enter) the new PIN		
3	Re-enter '5555'	The system displays a message of successful operation The system asks the user if he wants to perform other operations		
4	Click 'YES' button	The system displays the main menu		
5	**Check post-condition 1**			

Post-conditions

1. The new PIN '5555' is saved in the database

Test Case (Login example)

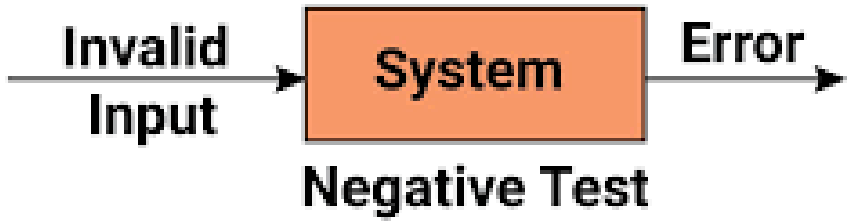
العنصر	الوصف
Test Case ID	TC_Login_01
Title	تسجيل دخول ببيانات صحيحة
Preconditions	المستخدم مسجل بالنظام
Test Steps	1. فتح صفحة الدخول 2. إدخال اسم مستخدم صحيح 3. إدخال كلمة مرور صحيحة 4. الضغط على Login
Test Data	username = inas password = 1234
Expected Result	يتم الدخول إلى الصفحة الرئيسية
Status	Passed

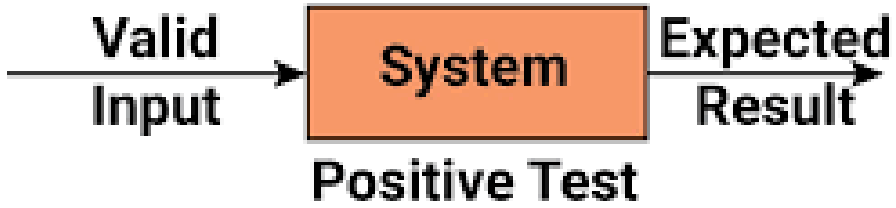
Negative testing

الاختبار السلبي (Negative Testing) هو أحد أساليب **اختبار البرمجيات** ويهدف إلى **التحقق من سلوك النظام عند إدخال بيانات غير صحيحة أو غير متوقعة**، للتأكد من أن النظام **لا ينهار** ويتعامل مع الأخطاء بشكل صحيح.

ويمكن تعريفه بانه عملية اختبار يتم فيها إدخال **قيم خاطئة، أو غير صالحة، أو خارج الحدود المسموحة** للتأكد من أن النظام:

- يكتشف الخطأ
- يرفض الإدخال غير الصحيح
- يعرض رسالة خطأ مناسبة
- يحافظ على استقراره دون توقف أو انهيار





مثال على الاختبار السلبي

نموذج تسجيل دخول

- إدخال اسم مستخدم غير موجود
- إدخال كلمة مرور خاطئة
- ترك الحقول فارغة
- إدخال رموز خاصة في حقل غير مخصص لها

النتيجة المتوقعة:

رفض تسجيل الدخول مع رسالة خطأ مناسبة.

المقارنة	الاختبار السلبي	الاختبار الإيجابي
البيانات المدخلة	غير صحيحة	صحيحة
الهدف	التأكد أن النظام يكتشف الادخال الخاطئ ويتعامل معه بطريقة صحيحة ولا يفشل في عمله	التأكد أن النظام يعمل
النتيجة المتوقعة	رفض مع رسالة خطأ	تنفيذ ناجح

Enter Only Numbers

abcdef

Negative Testing

Enter Only Numbers

99999

Positive Testing

Test Suite حزمة الاختبار هي **مجموعة منظَّمة من حالات الاختبار (Test Cases)** تُستخدم لاختبار جزء معيّن من النظام أو النظام كاملًا، بهدف التحقق من أن البرنامج يعمل كما هو متوقَّع.

Test Suite: Login Module

Test Case ID	Description	Input	Expected Result
TC01	تسجيل دخول صحيح	user/pass صحيح	دخول ناجح
TC02	كلمة مرور خاطئة	user/pass خطأ	رسالة خطأ
TC03	حقل فارغ	username فارغ	تنبيه للمستخدم

Black Box Techniques

فيما يلي شرح واضح ومختصر لكل تقنية من التقنيات المستخدمة لتصميم حالات الاختبار بشكل فعّال في اختبارات الصندوق الأسود

1- Equivalence Partitioning(EP)

التقسيم التكافؤي: تقسيم بيانات الإدخال إلى مجموعات (Partitions) يُفترض أن يتصرف النظام معها بنفس الطريقة، ثم اختبار قيمة واحدة فقط من كل مجموعة.

يستخدم لتقليل عدد حالات الاختبار TCs مع المحافظة على التغطية أي دون التأثير على جودة الاختبار حيث تستخدم هذه التقنية لتتوافق مع مبدأ الجودة اختبار كل شيء مستحيل .

مثال

اذا كان حقل العمر يقبل القيم من ١٨ إلى ٦٠ فان لدينا ثلاث مجموعات

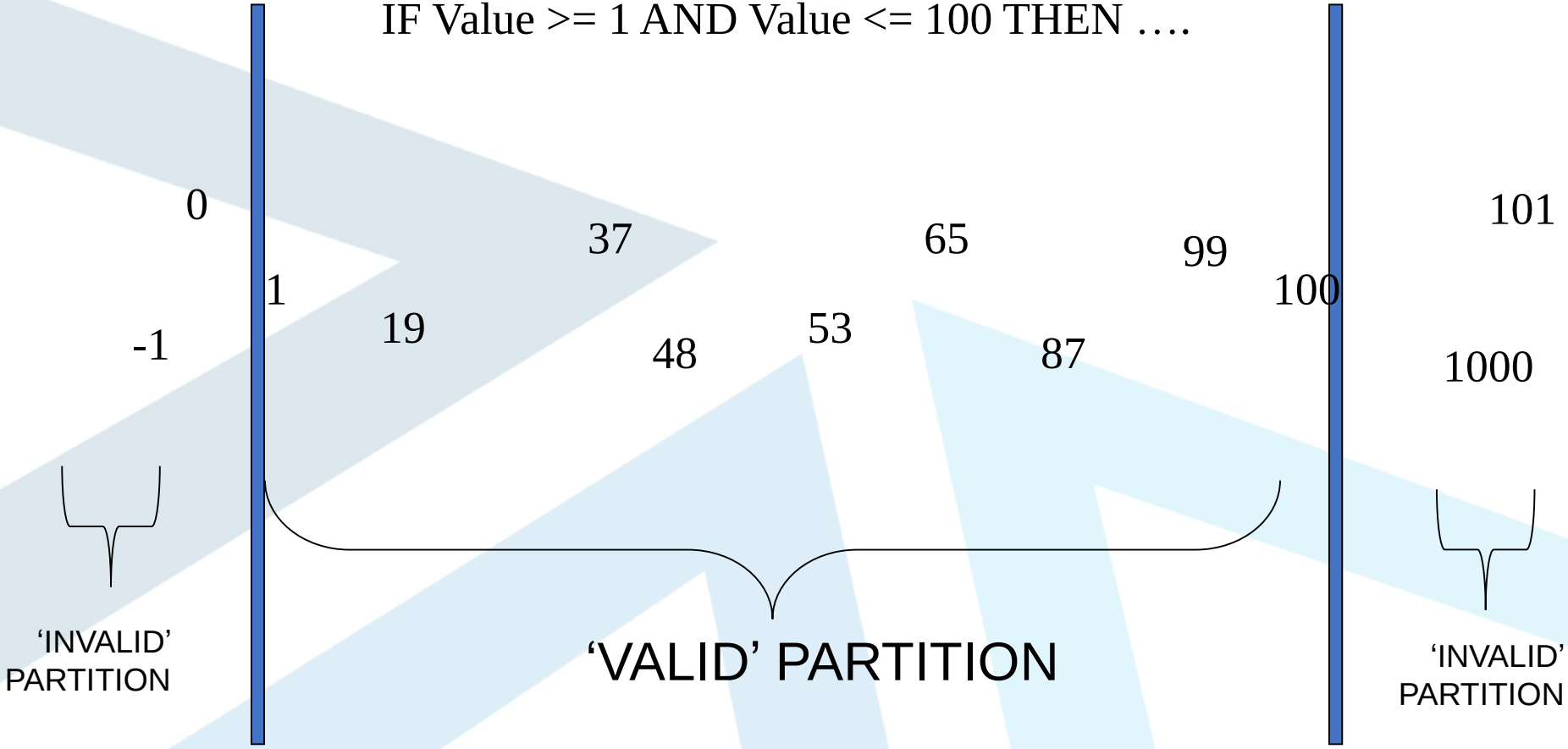
- مجموعة صحيحة: ١٨–٦٠
- مجموعة خاطئة: أقل من ١٨

مثلا نختبر:مجموعة خاطئة: أكثر من ٦٠

٢٥ (صحيحة) ١٠ (خاطئة) ٧٠ (خاطئة)

Equivalence Partitioning example

IF Value >= 1 AND Value <= 100 THEN ….



مثال:

بفرض لدينا برنامج يأخذ عدد صحيح بين ال ١ و ١٠٠ ويعيد مربعه

والمطلوب: إيجاد Test Suite لتحقيق تغطية ١٠٠% وفقاً للتقسيم التكافؤي Equivalence Partitioning

الحل:

Test cases for program 'Square' based on Input domain

Test Case	Input x	Expected Output
I_1	0	Invalid Input
I_2	50	2500
I_3	101	Invalid Input

مثال:

بفرض لدينا مجموعة حسومات على شراء القطع الالكترونية، حيث يقوم البرنامج بحساب الحسم اعتمادا على قيمة الشراء وفق التالي:

Price	Discount
100$ or less	10%
Between 100$ and 500$	30%
500$ or more	50%

والمطلوب: إيجاد Test Suite لتحقيق تغطية ١٠٠% وفق Equivalence Partitioning

الحل:

Test Case	Input	Expected Output	Discount
I_1	-30	Error Message	invalid class
I_2	50	45	10%
I_3	300	210	30%
I_4	800	400	50%

2-Boundary Value Analysis (BVA) تحليل القيم الحدّية

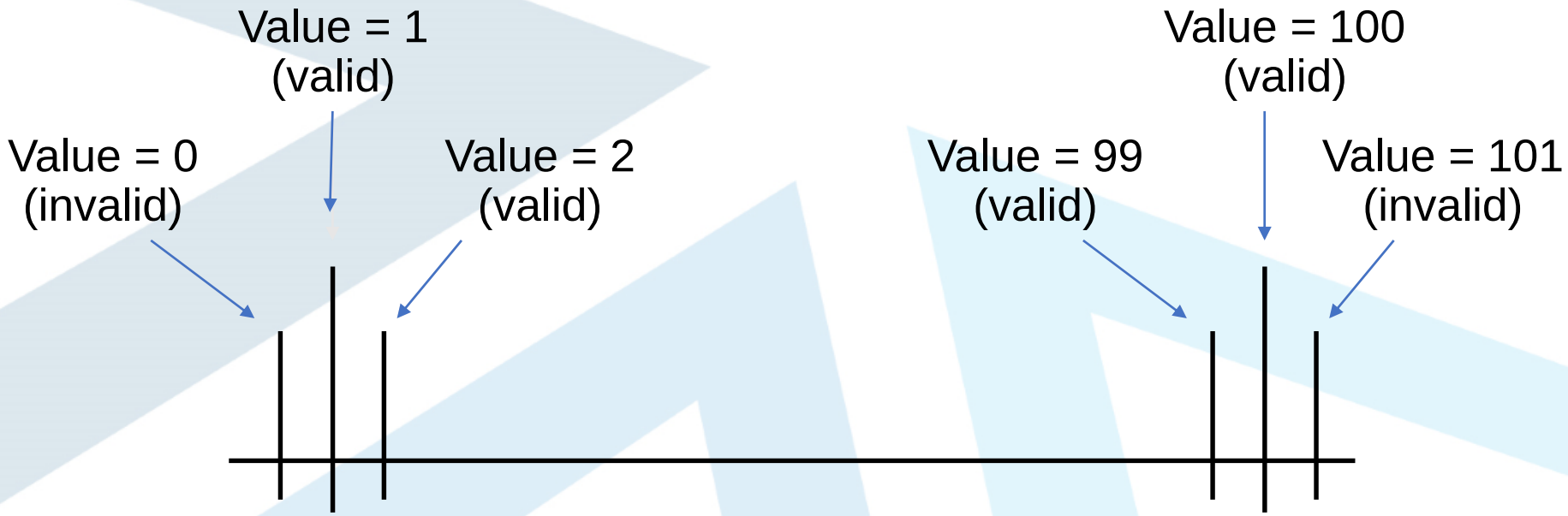
من الملاحظ أن الحدود بين المجالات تتحمل أو قد تقع فيها نسبة كبيرة من الاخطاء وذلك بسبب عدم الدقة أثناء تحقيق الشروط (مثل كتابة =< بدلا من <) أو عدم الدقة في فهم المتطلبات **ولذلك من المهم اختبار سلوك البرمجيات عند قيم هذه الحدود** بالإضافة إلى **السلوك في النقاط الطبيعية داخل المجال**.

- تعتمد BVA على أساس أن التجارب السابقة أظهرت أن الكثير من الأخطاء ظهرت عند الحدود بين المجالات، وعليه يجب دمج مجموعة من الاختبارات السلبية **negative testing into the test design** عند تصميم الاختبارات.

- يُستخدم تحليل القيم الحدية BVA عادةً بالتزامن مع EP في تصميم حالات الاختبار وهو قابل للتطبيق غالبا فقط في الحقول الرقمية والتواريخ

Boundary Value Analysis

IF Value >= 1 AND Value <= 100 THEN ….



Black Box Techniques

- Equivalence Partitioning
- Boundary Value Analysis
- Decision Tables
- State Transition Testing
- Use Case Testing

متى نستخدم جداول القرار؟

- ❑ عند وجود **منطق أعمال معقد**
- ❑ عندما تعتمد النتيجة على **عدة شروط**
- ❑ تقلل من نسيان حالات مهمة

3-Decision Tables

جداول القرار

جداول القرار (Decision Tables) هي إحدى **تقنيات الاختبار في هندسة البرمجيات** تُستخدم لتمثيل **القواعد المنطقية المعقدة** التي تعتمد على مجموعة من **الشروط والنتائج**، وتساعد في اشتقاق حالات اختبار دقيقة ومتكاملة.

يعد جدول القرار من الأساليب المعتمدة لاختبار سلوك النظام يتفاعل وفقًا لمزيج من المدخلات اعتمادًا على مجموعات معينة من شروط الإدخال، حيث يتم صياغة المدخلات في جدول اتخاذ القرار مما يساعد في كتشف ثغرات في النظام.

جدول القرار هو تمثيل جدولي يوضح:

- **الشروط (Conditions)** التي تؤثر على القرار
- **القواعد (Rules)** وهي جميع التراكيب الممكنة للشروط
- **الإجراءات أو النتائج (Actions)** التي يجب تنفيذها لكل قاعدة

تُستخدم عندما يعتمد سلوك النظام على **أكثر من شرط في نفس الوقت**.

مثال:

نظام يمنح خصمًا إذا:

كان المستخدم عضوا

وكانت قيمة الشراء > ١٠٠

مثال بسيط على جدول قرار نظام منح خصم لزبون

الشروط:

- C1 هل الزبون عضو؟
- C2 هل قيمة الشراء ≤ ١٠٠؟

الإجراءات:

- A1 منح خصم ١٠%
- A2 لا يوجد خصم

عدد حالات الاختبار = عدد القواعد = ٤

الشروط / القواعد	R1	R2	R3	R4
الزبون عضو؟	نعم	نعم	لا	لا
الشراء ≤ ١٠٠؟	نعم	لا	نعم	لا
منح خصم ١٠%	✓	✗	✗	✗
لا يوجد خصم	✗	✓	✓	✓

يحوي جدول القرار k عمود حيث k هو (عدد حالات الشرط الأول * عدد حالات الشرط الثاني * عدد حالات الشرط الثالث * ....) حيث يمكن اختزال عدد الاعمدة لاحقا

اما عدد الأسطر في جداول القرار فيعتمد على عدد بارامترات الإدخال والمخرجات المتعلقة بها

كل عمود في جدول القرار يمثل حالة محتملة للنظام، وبالتالي يتم تحويله إلى حالة اختبار.

يمكن بسهولة تحويل أي تدفق أعمال معقد إلى سيناريوهات الاختبار وحالات الاختبار باستخدام جداول القرار،

تضمن هذه الجداول مراعاة كل مجموعة ممكنة من قيم الشروط، هذا هو المعروف باسم "**completeness property**" خاصية الاكتمال

Conditions - *(Condition stub)*	Condition Alternatives – *(Condition Entry)*
Actions – *(Action Stub)*	Action Entries

Condition entry

	Rule1	Rule2	Rule3	Rule4
Condition1	Yes	Yes	No	No
Condition2	Yes	No	Yes	No
Action1	Yes	Yes	No	No
Action2	No	No	Yes	No
Action3	No	No	No	Yes

Condition stub

Action stub

Action Entry

Conditions	TC02	TC08	TC09
UserID	Valid	Invalid	Valid
Password	Blank	Invalid	Valid
Actions			
Login Successfully			Execute
Error showing 'Invalid Credentials'	Execute	Execute	

Decision Table Testing example

	Test 1	Test 2	Test 3
> 55 yrs old	F	T	T
Smoker	F	T	F
Exercises 3 times a week +	T	F	T
History of Heart Attacks	F	T	F
Insure	Y	N	Y
Offer 10% Discount	N	N	Y
Offer 30% Discount	Y	N	N

What will be the out come of the following Scenarios?

Joe is a 22 year old non smoker who goes to the gym 4 times / week and has no history of heart attacks in his family

Kevin is 62 year old non smoker who swims twice a week and plays tennis. He has no history of heart attacks in his family

Example

شركة برمجية تقوم باعطاء رواتب لموظفيها وفق التالي:

غير متخرج Ungraduated بدوام جزئي Part Time راتب ٣٠٠$

غير متخرج Ungraduated بدوام كامل Full Time راتب ٨٠٠$

متخرج Graduated بدوام جزئي Part Time راتب ٦٠٠$

متخرج Graduated بدوام كامل Full Time راتب ١٥٠٠$

والمطلوب: ايجاد جدول القرار وايجاد حالات الاختبار بنسبة ١٠٠%.

جدول القرار

Conditions	1	2	3	4
متخرج	F	F	T	T
دوام كامل	F	T	F	T
Actions				
الراتب	300	800	600	1500

اختزال جداول القرار (Decision Table Reduction / Simplification) هو عملية **تقليل عدد القواعد (Rules)** في جدول القرار دون التأثير على **صحة المنطق أو تغطية حالات النظام**، بهدف جعل الجدول **أبسط وأسهل في الفهم والاختبار**.

لماذا نختزل جداول القرار؟

- ❑ تقليل عدد **حالات الاختبار**
- ❑ إزالة القواعد **المكررة أو غير الضرورية**
- ❑ تبسيط منطق الأعمال

دمج القواعد المتشابهة (Rule Merging)

إذا كانت عدة قواعد تؤدي إلى **نفس الإجراءات**، يمكن دمجها باستخدام رمز – (Don't Care)

استخدام رمز (–) يعني أن الشرط **لا يؤثر على النتيجة**

مثال ٢ نظام يقوم بمنح قروض بدون كفيل اذا كان الشخص موظف وعمره أقل من ٥٠ سنة، وذلك اذا كان دخله أكثر من ٢٠٠٠٠ ليرة سورية جديدة اما اذا حقق الشروط السابقة ولكن كان دخله اقل فانه يمنح قرض ولكنه يحتاج لكفيل اما في الحالات الاخرى فان النظام لا يمنح قرضاً للشخص.

والمطلوب: ايجاد جدول القرار واختصاره وايجاد حالات الاختبار بنسبة ١٠٠%

Conditions	1	2	3	4	5	6	7	8
موظف	F	F	F	F	T	T	T	T
العمر< ٥٠	F	F	T	T	F	F	T	T
الدخل< ٢٠٠٠٠	F	T	F	T	F	T	F	T
Actions								
يمنح القرض	F	F	F	F	F	F	**T**	**T**
وجود كفيل	F	F	F	F	F	F	T	F
لا يمنح القرض	T	T	T	T	T	T	F	F

بعد الاختزال

Conditions	1	2	3	4
موظف	F	T	T	T
العمر< ٥٠	-	F	T	T
الدخل< ٢٠٠٠٠	-	-	F	T
Actions				
يمنح القرض	F	F	T	T
وجود كفيل	F	F	T	F
لا يمنح القرض	T	T	F	F

Test-Suite Table

No	Input	Expected output
1	الشخص غير موظف عمره ٣٠ راتبه ١٥٠٠٠	لا يمنح القرض
2	الشخص موظف عمره ٥٥ راتبه ٣٠٠٠٠	لا يمنح القرض
3	الشخص موظف عمره ٢٥ راتبه ١٥٠٠٠	يمنح القرض مع كفيل
4	الشخص موظف عمره ٤٥ راتبه ٣٠٠٠٠	يمنح القرض بدون كفيل

مثال ٣: يقبل الطالب في المنحة إذا كان خريج هندسة معلوماتية، معدله يزيد عن ٨٠، وعمره لا يزيد عن ٣٠ سنة، وإذا كان الطالب حاصل على التوفل يبدأ دراسة المواد مباشرة أو يبدأ بسنة لغة.

والمطلوب: ايجاد جدول القرار واختصاره وايجاد حالات الاختبار بنسبة ١٠٠%.

Conditions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
خريج هندسة معلوماتية	F	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T
معدله أكبر من ٨٠	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T
العمر أصغر من ٣٠	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T
حاصل على التوفل	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T
Actions																
يقبل بالمنحة ويبدأ المواد	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	T
يقبل بالمنحة ويبدأ سنة لغة	F	F	F	F	F	F	F	F	F	F	F	F	F	F	T	F
لا يقبل بالمنحة	T	T	T	T	T	T	T	T	T	T	T	T	T	T	F	F

بعد الاختزال

Conditions	1	2	3	4	5
خريج هندسة معلوماتية	F	T	T	T	T
معدله أكبر من ٨٠	-	F	T	T	T
العمر أصغر من ٣٠	-	-	F	T	T
حاصل على التوفل	-	-	-	F	T
Actions					
يقبل بالمنحة ويبدأ المواد	F	F	F	F	T
يقبل بالمنحة ويبدأ سنة لغة	F	F	F	T	F
لا يقبل بالمنحة	T	T	T	F	F

Test-Suite Table

No	Input	Expected output
1	المتقدم ليس خريج هندسة معلوماتية معدله ٨١ وعمره ٢٥	لا يقبل بالمنحة
2	المتقدم خريج هندسة معلوماتية معدله ٧٥ وعمره ٢٥	لا يقبل بالمنحة
3	المتقدم خريج هندسة معلوماتية معدله ٨٥ وعمره ٣١	لا يقبل بالمنحة
4	المتقدم خريج هندسة معلوماتية معدله ٨٥ وعمره ٢٥ وليس حاصل على التوفل	يقبل بالمنحة ويبدأ سنة لغة
5	المتقدم خريج هندسة معلوماتية معدله ٨٥ وعمره ٢٥ وحاصل على التوفل	يقبل بالمنحة ويبدأ المواد


HAVE ANY
QUESTIONS AND DOUBTS