

# Advanced Programming

\* الـ class عبارة عن مجموعه من methods و attributes يبني على class inheriting و overriding.

\* class هوا object ينتمي إلى class student والـ attribute هما name و grade التي تكونا part of student.

\* methods هي functions و procedures و functions يكون inside class body only.

\* classes  $\leftarrow$  super و sub class Inheritance.  
\* overwrite  $\leftarrow$  + use + add.  
\* super class exists before sub class exists and sub class inherits from super class.  
\* super class exists before sub class exists and sub class has its own methods.

\* method parameters and return type are the same as super method but you can change them.

\* extends class means that sub class inherits methods and variables from super class.

\* extends class means that sub class inherits methods and variables from super class.

\* full compile for gdk ji IDE is not available.

\* class extends java single in inheritance.

\* You can add + use + overwrite.

\* private attribute is encapsulated and can be accessed only through getters and setters.

\* getter and setter use private attribute.

\* الـ attribute هو المكان الذي يصوّر فيه Variables و ما يصرّر

\* أثبت منه كود زبادي وما بال setter والـ getter يقرّر أين زبادي يـ

\* بالعادة يعطى المـ attribute رجـ فعل attribute و يصل الفـ private getter و setter

\* الـ set هي التي تخطـ الـ value

مـ

\* الـ abstraction هو عـ بـرهـةـ methodـue بدون كـ وـ وـ تـ خـرـ منـهاـ انهـ هـايـ مـكـنـهـ يـعـلـمـهـ اـخـرـ خـصـائـصـ وـ يـعـدـواـ كـ وـ

\* different implementation ofـ خـصـائـصـ عـلـيـهاـ وـ هـيـ عـبـرهـ

the same method

\* مـ ١ـ الـ skـ يـتـزـمـ بـ مـ بـ نـ اـخـاـ طـبـيـتـهـ

\* هـيـ كـائـنـ بـعـيـ وـيـتـزـ مـ بـ اـخـاـ طـبـيـتـهـ (ـبـحـرـهـ)

\* abstract methodـisـ classـ لـ extendـ classـ لـ

فـ الـ classـ يـعـنـ دـيـنـهـ كـ وـ

\* الـ abstraction وـ encapsulation وـ inheritance

عـبـرهـ مـفـاهـيمـ الـ OOP

(ـرـاجـعـ فـيـ الـ رـابـعـ)

\* Polymorphismـ هوـ كـيـوـيـ عـلـيـ الـ methodـ overloadingـ وـ الـ inheritanceـ

وـ يـعـنـ انـ الـ codeـ يـكونـ الـ o~o~

\* overloadingـ اـنـ عـيـ اـسـ نـ methodـ وـ الـ signatureـ بـ مـ

وـ ظـائـفـ مـخـتـفـ (ـرـفـ الـ a~s~ دـيـنـ الـ signatureـ بـ مـخـتـفـ)

\* الـ overrideـ هـيـ اـنـ الـ superـ تكونـ مـنـ مـ

مـهـيـةـ وـ اـنـ اـبـرـجـ بـعـدـ مـنـهاـ لـ كـلـ مـاـ يـبـالـ subـ classـ

الـ Access Modifiers Scope  
protected default private public

الـ package بعده علىـ method هو default

sub class هو protected اذا كان في package و اذا كان في same package

-> naming convention

capital + noun ← class | small ← package

camel

case + noun ← Attribute | verb + small ← method  
(variable)

method هو الاشيء الذي يكتب ويكون فيه مجموعه من الـ class

object هو عبارة عن البناد او التغيرات المخاطة الى البرنامج class

و object هو المدخلات والoutput هو البناد

Class101 = new Class();  
new ایسی object بخیرم کہے

class هو الجامعه والـ object دین الکیان الي صنعا

integers او capital هی class نہیں String  
وala double primitive  
capital double class type  
علوہا class و خواہا

method دیکھیں وہیں بتکوہ data type void  
وانا كانت procedure  
نوع تابن بتکوہ function بترجمہ میتھ

القيمة التي يدخلها آخرها

```
public void setId(int id) {
    this.id = id;
}
```

\* الـ `this` الذي يدخله هو الـ `id` الذي يوصي به دفع المقدار المدروزة

```
public int getId() {
    return this.id;
}
```

\* خطأ نوع المقدار `int` ← method  
والـ `this` يدخل نوع الـ `id` وما يكتب عليه `return`  
ـ تقبل مثلاً بـ `return` قيمة وكتبه `return` لها في العينة

\* متى ندخل الكود لـ `main` أعمل `call` حالاً من خلال  
أنه أعمل `object` لـ `class` التي علامة مثل

\* حول ما هي `private` أو `set` `get` `private` ← attribute  
ـ أخاد `them` `private` → `main class` `class` ← و يستخدم الـ `set`  
ـ `method`

ـ أخطأ العينة و كملة العينة آخر أخطاء بـ `get`

\* يغير ربط `super` لـ `default value` `class`  
ـ كما عرفناهم

ـ ببيان `object` يقدر اسقاطه على `e1.setId(1);`  
ـ يخدم الـ `super` لـ `default value` وربط فيه `class`  
ـ `( set value )` ← `method` ← يدون ( `value` )  
ـ تفاصيل الـ `attribute` لـ `class` و زبون في `signature`  
ـ وزبون منه لتعريف المقدار

`this.id=id;`  
`this.names=name`

overload  
constructor

Five Apple

لما أحجَّ أعرِفُ الـ object حوزَ إِنْ فِي جِنادي لـ  
super class

default value هي الـ ~~value~~ class وبيَّنَ method المُحَاكَةِ

public Employee()  
id = 99;  
name = "Ahmed";  
}

default value

إِنِّي أَعْلَمُ الـ class إِنِّي أَعْلَمُ method المُحَاكَةِ  
وَالـ constructor

object  $\rightarrow$  new automatically  
constructor

إِنِّي أَعْلَمُ الـ attributes default  
complex class method

إِذَا اعْتَدَتْ constructor  
فَمُتَّسِّرٌ هُوَ لِلرِّزْمِ أَعْلَمُ الـ  
set method

غيرِ الـ constructor خاصَّ

إِذَا أَسْتَعِدَتْ client إِذَا يَرْتَدُ مُهْلِعَاتِ الـ  
default constructor وَيَبْلُغُ

إِنِّي أَعْلَمُ method المُحَاكَةِ وَصَرِيقَةً معِ الـ object تكونُ ذِيَّة  
منِّي وَبِرَكَلَاسِ تَانِيَاتِ علىِ الـ IDE وبِكُونِيَّاتِ

object extend هُوَ عَاملِ الـ بِرَكَلَاسِ بالـ class

إِذَا طَبَقَتْ الـ physical add object يَطْبَقُ الـ  
وَعَلَى إِنِّي أَعْلَمُ behavior

override وَيَجْعَلُ e1.toString() يَكتُبُ

class مُسْتَحْدِي مُسْتَحْدِي

Five Apple

attributes

↑

• ~~بِعْدَ الـ toString() يُعْطَى الـ source~~  
 Employee class  
 generate toString ← source ← right click  
 ويرجع لكتبة فتحة

object لـ attributes كل الـ object  
 el.toString() ومثله ما أطبع

2Attributes في المقدمة  
 address رجوع إلى المقدمة  
 equal لـ override

ما سبقه أول ماضي بين بالكاره

\* ما يصرّح حول المقدمة Employee.toString() أو id.equals()  
 ونقر زر تفاصيل (F5) بالعبارة وحيث بدون ما أخفى  
 نظم يتألّف إذا الموزر دخل في يصرّح حوله المقدمة Employee أو  
 id.equals() فيكتب بحوال

@Override

public boolean equals(Object obj) {

if (obj instanceof Employee) {

Employee that = (Employee) obj;

return this.id == that.getId();

}

return false;

}

هذا يعني أنّه يدخل إلى دالة

Employee

obj

Five Apple  
 function

\* كل الـ ~~public~~ (final) ~~circle~~ كلاس final وهي قيمة ثابتة  
لها قيمة وما يغيرها، ليس فقط في متغيره بل في المضف عليه ~~public~~

\* متغير ~~String~~ الذي أراد أن يكون عنوان class مع `name` ويرجع بـ `name`

`name` →

بعد إدخال أمر إحداث له مثلاً

\* الـ `String` وـ `methods` والـ `String`

ـ `String` نتائج رقم الجافا الدالة

ـ هو عبارة عن التحويل `(Type conversion)` Wrapper \*  
بين الأنواع والارتفاعات `(primitive types)` بـ `name`

`byte, short, int, long, float, double, char, boolean.`

\* أي اثنين غير تحول الأطوال يمكن أن يكونوا مختلفين

ـ `long` هي عبارة عن `int` مع مساحتها أكبر والـ `short`  
وـ `byte` من أقل تلهم يوحظوا في عدد بعدهم مساحات مختلفة  
وفترتب ~~أكعراً~~ الأبعاد تنازلياً بدءاً بالـ `byte`

\* `primitive types` ~~classes~~ الـ `capital` يكون أول حرف

\* `toString()`/`parseInt()` الـ `method` الـ `classes`  
 ↓ ~~converts~~ `int`   
 بـ `strings` / `ParseDouble()`  
 ↓ `Float`  
`Long`

\* من حالة sum بمحضها تكون صفر فإذا كانت حزبة بخط فـ ~~أو~~ وآخر  
الـ ~~args~~ ~~array~~ ~~to~~ ~~main~~ ~~in~~ ~~the~~ ~~args~~ ~~is~~ ~~an~~ ~~array~~ ~~of~~ ~~String~~ ~~and~~ ~~we~~ ~~pass~~ ~~it~~ ~~to~~ ~~the~~ ~~main~~ ~~method~~ ~~as~~ ~~a~~ ~~parameter~~  
\* ~~or~~ ~~return~~ ~~the~~ ~~sum~~ ~~value~~ ~~to~~ ~~the~~ ~~user~~

Right click on class ~~to~~ ~~choose~~ ~~Run/Debug~~ ~~Run~~ ~~Choose~~ ~~arguments~~ ~~javaapp~~ ~~and~~ ~~edit~~ ~~give~~ ~~new~~ ~~values~~  
\* ~~choose~~ ~~the~~ ~~arguments~~ ~~space~~ ~~between~~ ~~the~~ ~~values~~

ويمكننا ~~arg~~ ~~values~~ ~~to~~ ~~be~~ ~~separated~~ ~~by~~ ~~space~~ ~~between~~ ~~the~~ ~~values~~

\* ~~we~~ ~~can~~ ~~also~~ ~~deal~~ ~~with~~ ~~object~~ ~~and~~ ~~print~~ ~~it~~ ~~on~~ ~~the~~ ~~screen~~

Integer or ~~Object~~ ~~class~~ ~~has~~ ~~no~~ ~~instance~~ ~~variables~~ ~~but~~ ~~it~~ ~~has~~ ~~methods~~ ~~like~~ ~~parseInt~~ ~~and~~ ~~toString~~

Integer ~~x~~ = new Integer(0); ~~variable~~ ~~object~~

sum = sum + x.parseInt(args[i]);

\* ~~or~~ ~~attributes~~ ~~class~~ ~~has~~ ~~no~~ ~~instance~~ ~~variables~~ ~~but~~ ~~it~~ ~~has~~ ~~methods~~  
object ~~class~~ ~~has~~ ~~no~~ ~~instance~~ ~~variables~~ ~~but~~ ~~it~~ ~~has~~ ~~methods~~  
ومعرفة على ~~the~~ ~~object~~ ~~we~~ ~~call~~ ~~the~~ ~~method~~ ~~parseInt~~ ~~on~~ ~~the~~ ~~object~~  
المعرفة على ~~the~~ ~~object~~ ~~we~~ ~~call~~ ~~the~~ ~~method~~ ~~toString~~ ~~on~~ ~~the~~ ~~object~~  
بـ ~~the~~ ~~object~~ ~~we~~ ~~call~~ ~~the~~ ~~method~~ ~~parseInt~~ ~~on~~ ~~the~~ ~~object~~

\* ~~we~~ ~~can~~ ~~use~~ ~~the~~ ~~method~~ ~~parseInt~~ ~~of~~ ~~Integer~~ ~~class~~ ~~in~~ ~~the~~ ~~main~~ ~~method~~  
main ~~class~~ ~~we~~ ~~call~~ ~~the~~ ~~method~~ ~~parseInt~~ ~~on~~ ~~the~~ ~~object~~ ~~we~~ ~~call~~ ~~the~~ ~~method~~ ~~toString~~ ~~on~~ ~~the~~ ~~object~~

\* ~~the~~ ~~method~~ ~~parseInt~~ ~~is~~ ~~static~~ ~~so~~ ~~we~~ ~~call~~ ~~it~~ ~~without~~ ~~an~~ ~~object~~  
we ~~call~~ ~~the~~ ~~method~~ ~~parseInt~~ ~~on~~ ~~the~~ ~~class~~ ~~Integer~~ ~~and~~ ~~not~~ ~~an~~ ~~object~~

static attributes

Static methods  $\leftrightarrow$  static مع التعامل مع static methods

array / map / hashmap / set / vector : (containers) collections

int a[] = new int[5]; : array التعريف \*

يقرر أخر فار class لـ array أو الممتلكات الخاص بال class  
Employee e[] = new Employee[5];

ويمكن أن يُعرف بال 100 موظف لارئهم مثلاً واحداً آخر  
وأعمل now على كل دالة دالة loop.

for(int i=0; i < e.length; i++) {

e[i] = new Employee();

}  
e[0].setName("Ali");

الـ Vector . الـ size . الـ add بالعمره تكون  
غير المائية ونوعه methods ويعبر  
أو خط منه يعني البيانات ويمكن إضافة كل الزمام  
إنه أو سختم منه في datatype .

util.  
java.util import ;

الـ object التي تجوب كل الأنواع هو الـ datatype  
هي datatype .

Vector<String> vstring = new Vector<String>();  
Vector<String> vstring = new Vector<String>();

methods هي مجموعه class

\* يفضل استخدام الـ Vector بدلاً من الـ array

\* هو ابن أبنائه كائن الـ main تابع امتهن Casting او Casting() نوته وـ parameters بـ program

19 بـ casting casting TestVector Class

\* كون أباً لـ بـ دراً وهو

\* الـ duplicates هو مجموعه collection set array collection والـ الخواص وـ الحفاظ على ترتيب معين للعناصر فعاليتـ casting يرتبطوا بذلك مرتبة زمانية فيها

\* إذاً دلـ casting class abstract class قبل كـ casting ذـ casting  
ـ casting class وـ casting main وـ casting main

\* دلـ casting main هـ casting main دلـ casting main  
ـ casting main دلـ casting main دلـ casting main دلـ casting main  
ـ casting main دلـ casting main دلـ casting main دلـ casting main

\* collection دلـ hashmap دلـ collection دلـ hashmap  
ـ key دلـ value دلـ key دلـ value دلـ key دلـ value دلـ key

\* يـ collection دلـ collection دلـ collection دلـ collection دلـ collection

full abstract classes هم وحدة interface \*  
التي لها مетодات تكونوا بدون كود ويعرفن ما أنت  
abstract

implements ~~is~~ keyword \*

abstract و public ~~يكون في~~ methods () by default \*

interface classes ~~هي~~ هي ~~جزء من~~ package \*

value () يعبر عن ~~أمثلة~~ attributes ~~بأنه~~ final ~~لأنها~~

upper case all تكون ~~بأنها~~ naming convention \*

HELLO\_WORLD under score ~~ويحصل بينهم~~

override () يعمر المثود final \*

extends () يعمر class final \*

exception ~~غير~~ عام هو الـ throw ~~يطلب~~  
الغور و لازم احتفظ ~~تحفظ~~ على هاذه الـ throw ~~الغور~~  
او ~~تحفظ~~ ~~الغور~~

throws ~~ما~~ exception ~~لـ~~ ~~الغور~~ معال ~~لـ~~ ~~الغور~~  
و هـ ~~الغور~~ ~~الغور~~ class ~~الـ~~ ~~الـ~~ جهـ

catch and try ~~ما~~

\* وظيفة try catch في أحد الكود الى تحيط بها try catch وتحيطها باليمين right click ~~وتحيطها~~

## try/catch Block

exception is database file والخطأ معرفة ، كل بثباتها تغير حوى (Fail exception) باتجاه اثمرتوا لها

file Readers file writer المروحة هي افعى افعى

classes 2 methods في file writer close و write

File Reader object

Buffered Reader object في filereader

وبعد ما يجيء على الـ readline (ارجع المراجع)

Test Files ← eclipse متى موجود على eclipse

لابد من وجود الكود حاليه ١١. Generic شرح

data types التي تكونوا methods لـ parameters لـ interfaces classes

data type الذي يحد نوع الـ generic

الـ super لـ object يحيط بالـ generic

public class GenericClass<myType> و مكتوب Five Apple  
private myType value;

ـ معاصرة اليوم  
Object Oriented

ـ نظرية حان

ـ هو این اخون (UML) Unified Modeling language

ـ تم على كل دساتر على مرآته وروي

ـ الاختلافات حوله ومن ثم الناحي المعاين يعرفوا بغير أور ويفهم  
ـ صن وبنوكه های اور standard ← UML ← حاصلیکها خاص معاين

ـ UML Class اول  
ـ Diagram دفعه التجزع المفهوم  
ـ classes و properties  
ـ attribute

ـ methods و attributes ای دیاگرام Class دلایل  
ـ relationships بينهم  
ـ Protected # | Public + | Private --  
ـ Package

ـ class دلایل class دلایل dependency  
ـ class دلایل dependency دلایل class دلایل

ـ Using و بنظر عده بکلمه

ـ arguments الى ای دلایل تخدم دلایل class دلایل  
ـ the class دلایل parameters

ـ diagram all attributes دلایل columns کل

ـ Bean دلایل class دلایل  
ـ مراجع الارائه

DAO هي class  $\rightarrow$  يحتوي على drop / update / select / insert

drop / update / select / insert  
or delete

Dean هو الـ DAO من الـ Super  $\rightarrow$  Sub

- ٥.٤ / ٢٤ - الـ DAO  $\rightarrow$  الـ DAO  $\rightarrow$  الـ DAO

(Inheritance) Generalization  $\rightarrow$  الـ DAO  $\rightarrow$  الـ DAO

If represents is a relationship  $\rightarrow$  is a  $\rightarrow$  relationship

يكون العلاقة بين الـ inheritance العام (super class) والـ الـ inheritance المـ (sub class)

Visual paradigm  $\rightarrow$  relationship

Realization  $\rightarrow$  implementation

implementation  $\rightarrow$  class و interface  $\rightarrow$  implementation

contract  $\downarrow$  implementation

Visual paradigm  $\rightarrow$  implementation

العلاقة الرابعة هي الـ  
ويندرج تحتها الـ Composition و Aggregation

classes والـ objects يمثلان Association والـ objects

Two classes only Binary association

three entities Ternary association

(خط ملائم فقط) solid line + ~~line~~

Multiplicity والـ Role بعدد أصنف عالم الخطاب

Role يندرج منها العددية  $\{1\}$  انحدارياً بمعنى  $1..1$  instructor و  $1..1$  student

العنوان  $\{1..1\}$  object متعدد مسؤوليًّا بين معلم واحد

الطالب الواحد يدرس  $\{1..1\}$  instructor والـ student واحد يدرس طلب واحد  $\{1..1\}$

exactly one

1

Zero or one

0..1

Zero or many

0..\*

One or many

1..\*

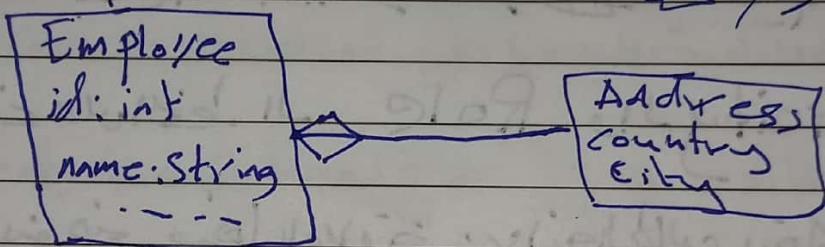
Many

\*

محاضرة ٤ / ١٢٦ / ٤ / ٢٠١٥  
مراجع ملخص

ارحللها بحري

parent child has a relationship called Aggregation.  
parent has child relationship called composition.  
add address to employee meaning that employee has address.  
address can be shared by many employees.  
the address can be shared by many employees.



آخر علاقه هى الـ composition أو aggregation  
part of

relationship

آخر علاقه هى الـ class و هو عباره عن  
يربط بين كلتين مختلفتين و يحتوي على المعلومات المترافقه  
ما يفرق ايه بين class و association  
الفرق بين them many-to-many relationship  
يختلف كلا من them many-to-many relationship  
و يعتمد فقط على

اجعل المورد

student list مقدمة خزن دلائل مفهومي  
البيانات

function dynamic ← parameter إذا جدّي ←  
البيانات ← ... وبيانات Varargs ←

## Design Patterns ← المجموعات ←

دوافع مثل مارتن فايلر ←  
أمثلة

نوع ← هو قابل للتغيير ← design patterns ←  
غير معتمدة ← OOP ← ومتغير ←  
له برمجية ←

behavioral structural creational ←

موجود على الواقع ← design patterns ←

Concrete class ← هو الـ class الذي ما فيه  
غير المطبقة ← abstract methods ←

objects ← أي objects concrete class ←

creational ← كل الـ objects ←

factory ← هو المفترض ←  
client ← client no objects ←

لهم ينوف وآخر class

ويني زبطة عام تحيط كل classes

classes() يرجع 600 2/1 client

فدون classes() factory new  
client() عاكله وينفع

client() في Pipelines

لارم 9 ابي فـ interface  
createNotification object

- ٥/٥/١٧ ماجنة \*

design patterns كرايكرو و بيعملها

implementations

بنية الور Structural

النوع الثاني في creational هو Prototype  
إنه أعاد object أول و المثل object السادس  
ما أنسى من المغير يستخدم الـ clone وليس  
Create

object no clone is in class

interface implement لارم موجود  
new() يرجع ما عدل كلـ cloneable

clone() method يرجع المـ clone();

protected Object clone() throws CloneNotSupported Exception

