الموضوع: لرمجة متطورة ٢ التاريخ: / /

سكشن ١

* الكلاس يشمل (دوال، كائنات، متغيرات).
* الكائن يمكن أن يكون بداخله كائن.
* ملف jdk يحتوي على المترجم والمفسر.

*خطوات لعمل برنامج بلغة java:

① فتح ملف Desktop من Net Beans IDE 8.2

② لفتح ملف جديد:

من File اختار new project (الأصح)

ثم اختار java ثم java application

ثم نضع اسم المشروع ومكانه ثم من اختيارك نضع اسم class والأفضل عمل اسم المشروع مثل اسم class كعمل الحجار يعمل بشكل صحيح ثم نضغط Finish.

يتم كتابة اسم class بشكل تلقائي ولكن عند تغييره يجب مراعاة شروط كتابة اسم class (أول حرف كبير).

* الدول يأتي بعدها (_).
* يوجد ٣ أنواع للكتابة أمر الطباعة:

Print / Println / Printf

Println ← يتم تأثير وظيفتها على ما بعدها

سكشن ٢

* لطباعة جملة يوجد طريقتان:

① كتابة كود: System.out.print(" ");

← علامة + لربط نص بنص أو أرقام

("" + "") ("" + 123)

* الإعلان عن متغير x :

- المتغير يمثل حاوية لثابت عند وضع قيمة له يجعله ثابت ؛ وعندما يتم وضع قيمة له يصبح متغير دائماً.

* عند الإعلان عن متغير x=123; ولطباعته :

System.out.print (x);

النص: لا يتم وضعه بين علامات تنصيص لأننا نريد طباعة قيمة x وليس طباعة متغير

* لطباعة قيمة x*2 بطريقتين :

① يتم الإعلان عن متغير y=x*x ونطبع y

② أو نعلن عن x وفي أمر الطباعة نكتب (x*2)

* لطباعة كل نص على سطر:

// Comment

① سنقدم Print ln التأثير على السطر الثاني

\n

② نضع "\n" بجانب أمر الطباعة

System.out.print (x + "\n");

* Package ← عبارة عن مكان لحفظ مكتبات اللغة، فهي عبارة عن الكلاسات

* لطباعة وقت وساعة الجهاز :

- يجب استدعاء مكتبة تسمى Date حرف كبير ويتم كتابتها في Package

import java.util.Date;

وتكتب كود

لكي نعرف استيراد مكتبة Date

ثم نكتب Date ← نوع المتغير واسم المتغير والدالة ويتم الإعلان بواسطة كائن جديد new

Date today = new Date();

↓ الاسم ↓ القيمة الوقت ← النوع

الموضوع: ______ التاريخ: / /

لطباعة المتغير today :

System.out.print("today is : " + today.toString());

لم يتم استعمال جمل الطباعة، يطبع الوقت والتاريخ كما هو ولا نقوم بجمعهم.

الفصل الرابع

* جمل التخصيص والمؤشرات:

1) جمل التخصيص Assignment Statements :

وهي نوعان

1) var_name = Expression

تخصيص قيمة واحدة لمتغير واحد

اسم المتغير

التعبير يمكن أن يكون:

1) قيمة ثابتة

2) قيمة متغيرة

لها قيمة مسبقة مرتبطة

بمؤثر نتيجة قيمة قابلة لمستعملة

* الرمز (=) ← يمثل إشارة التخصيص أو الإسناد

(تخصيص التعبير بالطرف الأيمن من هذا الرمز إلى المتغير بالطرف الأيسر)

مثال

star = '*';

pi = 3.14;

2) جملة التخصيص المتعددة

لتخصيص قيمة معينة إلى عدة متغيرات.

مثال

int A, B;

A = B = 123;

Var_name = Exp1 = Exp2 ......;

ب. المؤثرات operators : (1) أنواع من المؤثرات

أ. المؤثرات الحسابية Arithmetic operators

+ ، - ، * ، / ، %

ملحوظة * في حالة استخدام مؤثر القسمة (/) أن الناتج سيكون عدداً صحيحاً في حالة إذا كانت البيانات المستعملة أعداداً صحيحة.

* يجب أن تنفذ الأقواس إن وجدت (1) الضرب والقسمة وباقي القسمة (3) وأخيراً (2) الجمع والطرح وكلها من اليسار إلى اليمين.

ج. المؤثرات الارتباطية Relational operators

مهمتها ← إجراء عمليات على أي زوج من البيانات وينتج عنها قيمة منطقية

قد تكون False or True

== ، != ، > أقل من ، <= ، >= أكبر من

د. المؤثرات المنطقية logical operators

ينتج عنها إما True أو False

! ← not يعطي عكس القيمة ← يعني if=true إذا if=false

&& ← and يعطي True عندما يكون كل القيم True بينما

|| ← or " " عندما يكون أحد أو كل القيم True بينما

ه. مؤثرات الزيادة والنقصان Increment Decrement operators

++ ، -- ← يستعملان مع المتغيرات فقط

الشكل العام له: مؤثر الزيادة والنقصان ← متغير تم الإعلان عنه مسبقاً

(1) Var_name op ;

(2) op Var_name ;

وهناك فارق بين الشكلين حيث:

(1) Count++ ← يعني صنف الأول وبعدها احسب قيمة المتغير.

(2) ++Count ← يعني استخدم القيمة الحالية وبعدها صنف العدد 1.

و. المؤثرات المركبة

* طريقة مختصرة لعملية التخصيص التقليدية.

هي استخدام المؤثرات الحسابية والمؤثرات الخاصة بالبت مع أشارة التخصيص =.

الموضوع: _____ التاريخ: / /

الشكل العام لها: Variable op = expression

op ← يمكن أن تكون إحدى المؤثرات الحسابية والمؤثرات الخاصة بالبت.

← لهذه المؤثرات نفس الأسبقية وتنفذ من اليمين إلى اليسار.

ملحوظة * لا يكون هناك فراغ بين المؤثر وإشارة التخصيص.

+=، -=، *=، /=، %=، >>=، &=، =|، =^

X+=8 ⇒ X=X+8

ج. مؤثرات التعامل مع البت: Bitwise operators

تعمل كما يلي:

أ) مؤثرات البت المنطقية Bitwise logical operators

* مهمتها التعامل مع البت

* تتعامل هذه المؤثرات بعنصرين أو أكثر فيما عدا مؤثر (not) فهو عنصر واحد.

* أولويات تنفيذها تكون من اليسار إلى اليمين، ومن أعلى إلى أسفل حسب الآتي:

١) ~ (not) ← يعكس عكس العنصر

٢) & (and) ← قيمته تكون ١ إذا كان جميع عناصره ١ وعدا ذلك تكون صفر.

٣) ^ (Xor) ← وهو الجمع المفرد، ويعطي القيمة ١ إذا كان العناصر مختلفة و ٠ غير ذلك.

٤) | (or) ← قيمته تكون ٠ إذا كان جميع عناصره ٠ وعدا ذلك تكون ١.

ب) مؤثرات الإزاحة Shift operators

الشكل العام له:

١) Var_name = Var_name op expression;

٢) Var_name op= expression;

op ← مؤشر لاتجاه الإزاحة إلى اليسار أو اليمين

expression ← التعبير الذي يحدد عدد الخانات التي ستزاح

الموضوع: ________ التاريخ: / /

الفصل الخامس
الدوال

1. دالة الإخراج System.out.print

تقوم بطباعة المخرجات سواء كانت عددية أو حرفية.

مثال:

```
Public class Hello Egypt
{
 public static void main (String []s)
 {
  System.out.print ("Hello Egypt");
 } → method نهاية الوظيفة الدالة main
} → class نهاية الفصيلة
```

شرح البرنامج

السطر الأول Public class HelloEgypt {

الفصيلة العامة Public ← تسبق دائماً تعريف الفئة أو الفصيلة class
تعني أن هذه الفئة يمكن لأي فئة أخرى في البرنامج استخدام عناصرها.

* برنامج الجافا قد يتكون من أكثر من فئة class.

class ← هنا يتم بداية الفصيلة أو الفئة.

HelloEgypt ← اسم البرنامج (الكلاس) (المسمية زي ما انا عايز)

{ ← قوس بداية تعريف الفصيلة.

ملحوظة ← * عند حفظ ملف لجافا لابد أن يتم حفظه بنفس اسم الفصيلة وبنفس شكل الحروف والمسافات

السطر الثاني Public static Void main (String s[ ]) {

Void ← الدالة بعد تنفيذ البرنامج لن تعود بأي قيم.

main ← نقطة البداية لوظيفة الفصيلة main method.

(String [ ] s) { ← قوس البداية للدالة main.

String [] s ← مصفوفة من النوع الحرفي وتسمى s لتخزين جملة الطباعة في البرنامج.

ملحوظة * لغة الجافا هي لغة حساسة لحالة الأحرف لذلك يجب ملاحظة أن حرف S في كلمة String يجب أن يكون كبيرًا

السطر الثالث System.out.print ("Hello Egypt");
الأمر المراد طباعته ← أمر الطباعة ويجب أن يكون S كبير
ويوضع بين علامتي تنصيص حرف كبير ("") وقوسين

* خطوات كتابة وتنفيذ أي برنامج:
① كتابة البرنامج Writing
② ترجمة البرنامج Compiling
③ تنفيذ البرنامج Executing
④ مخرجات البرنامج Output

* متسلسلة الحروف هي مجموعة

ملحوظة * عند إدخال أرقام يجب إدراجها داخل جملة الطباعة فلا توضع بين علامتي تنصيص، فمن علامتي التنصيص تكتب دائمًا الحروف، وإذا تم كتابة أرقام داخل علامات التنصيص فإنها تعامل معاملة الحروف أي لا يمكن إجراء أي عمليات حسابية عليها.

③ دالة البناء Constructor :← أسلوب البناء، المشيد، البداية (تستخدم للتمهيد فقط)
- هي عبارة عن طريقة التكوين التي يتم بها إنشاء الكائن object من الكلاس فتأخذ نفس اسم الكلاس وتنفذ عند إنشاء الكلاس.
← هي عملية تمهيد للكائن وإعطاء المتغيرات الخاصة به قيم ابتدائية.
* كل كائن يوجد فيه أسلوب وظيفته هو فقط تمهيد وتحضير ووضع قيم افتراضية لخصائص الكائن.

* يوجد في الجافا قاعدة كلاسات classes جاهزة للإستخدام وتكون مجمعة داخل حزم packages.

الفصل السادس

جمل الاختيار / (جمل التحكم أو جمل اتخاذ القرار)

Selection Statements

① جملة الشرط if statement

لها صيغتان:

① الصيغة الأولى

الشرط (يجب أن يكون الشرط منطقيا)

```
if ( expression ) {
    statement 1 ;
    statement 2;
    }
```

ملحوظة: في حالة تنفيذ جملة واحدة فقط بعد جملة (if) فيمكنك الاستغناء عن الأقواس {} وفي هذه الحالة تنتهي جملة الشرط بالعبارة المنقوطة ; كالتالي:

```
if (expression)
    statement;
```

② الصيغة الثانية: if..else ...

```
if (expression) {
    Statement 1 ;   لو تحقق الشرط نفذ
    Statement 2;
    }
else {              ولو متحققش نفذ
    statement 3;
    statement 4;
    }
```

ملحوظة: يمكن حذف الأقواس الموجودة في جملة if وكذلك حذف الأقواس الموجودة بعد else عند وجود جملة ضباط واحدة بعدها مباشرة.

② جملة switch

ملحوظة: تستخدم عبارة if إذا كان جواب الشرط عبارة عن احتمالين أو ثلاثة احتمالات على الأكثر، أما إذا زاد عدد الاحتمالات على ذلك فمن الأفضل استخدام switch

```
switch ( variable)        ← يمكن أن تكون رقم أو حرف
{
 case value1:
  statement 1;
  break;  →  توقف واخرج خارج جملة switch
  ...
 default:  → جملة مبدئية يتم وضعها إذا لم يتحقق أي شرط
  statement;
}
```

ملحوظة: * جملة switch للأقواس بداية ونهاية.

* يجب دائماً أن تنتهي كل حالة case من حالات switch بالعبارة break.

الفصل السابع
جمل التكرار والدوران (الحلقات التكرارية)

for ① while ② do-while ③

① for

تكرار أمر معين أو مجموعة من الأوامر عدداً من المرات.

```
for (initialization ; condition ; increment ) {
    statement 1;  ← الجمل المراد تكرارها
    ...
}
```

initialization ← القيمة المبدئية التي يبدأ عندها العد.
condition ← الشرط.
increment ← مقدار الزيادة أو النقصان (الخطوة)

كود صحيح مهم

٢) while
هنا نحتاج إلى الشرط فقط وطالما كان هذا الشرط متحققا استمرت الحلقة في التكرار

```
While (expression) {
        statement 1;
        ...     2;
}
```

← هنا الشرط يأتي أولًا قبل تنفيذ الحلقة.
جمل يتم تكرارها

int a=0
While (الشرط)
{ الطباعة
هنا لازم وعمل الزيادة أو النقصان
لكي تتحقق من الشرط

ملحوظة
* في while لابد من وجود الأقواس لأن الجملة بصيغتها تتكون من أكثر من سطر.

٣) do-while
تختلف عن الحلقات السابقة في مكان وضع الشرط.
← يكتب الشرط بعد العبارات المطلوب تكرارها.
← تقوم بالدخول في الحلقة do ← نفذ
وتقوم بتنفيذ الأوامر.

```
int a=0
do
{
    statement 1;
    ...       2;
    ++a;
}
While (expression);
```

ملحوظة: إذا لم يتحقق الشرط فسوف
* يتم تكرار الحلقة مرة واحدة فقط.
* تقوم البرنامج بالتحقق من الشرط في نهاية الحلقة.

← يتحقق من الشرط