**المغيرات :(variables)**

هي عبارة عن أماكن يتم حجزها في الذاكرة تستخدم لتخزين قيم البيانات

في جافا هناك نوعين من المتغيرات:

1-المتغيرات البسيطة :

هي عبارة عن متغيرات معرفة مسبقا في الجافا , تبدأ بأحرف صغيرة لها حجم محدد يعتمد على نوعها تضم ثمانية أنواع يعرضها الجدول الاتي :

|  |  |  |
| --- | --- | --- |
| **الاستخدام** | **الحجم** | **النوع** |
| تستخدم في تخزين الأرقام ضمن المجال 128- الى 127 | 1byte | Byte |
| يستخدم في تخزين ارقام ضمن المجال -32786 الى 32767 | 2 byte | Short |
| تستخدم لتخزين الأرقام الصحيحة ضمن المجال من  -2,147,483,648 الى 2,147,483,647 | 4 byte | Int |
| يستخدم في تخزين ارقام صحيحة ضمن المجال من -9,223,372,036,854,775,808 الى 9,223,372,036,854,775,807 | 8 byte |  |
|  |  | Long |
| يستخدم في تخزين الأرقام العشرية كافية لتخزين 6 او 7 ارقام بعد الفاصلة العشرية | 4 byte | Float |
| يستخدم في تخزين الأرقام العشرية كافية لتخزين 15 رقم بعد الفاصلة العشرية | 8 byte | Double |
| يخزن بداخله احدى القيمتين فقط true او false | 1 bit | Boolean |
| يستخدم لتخزين محرف واحد او قيم من جدول الاسكي | 2 byte | Char |

2-البيانات المرجعية :

هي عبارة عن أي نوع اصله كائن من كلاس ونقوم بوضع كلمة new عند تعريفه ويمكن استدعاء مجموعة من الدوال (الطرق) من خلاله وتبدأ بأحرف كبيرة وجميعها لها نفس الحجم واذا لم نقم بإعطاء أي قيمة له توضع القيمة null كقيمة افتراضية .

النوع string يعد من البيانات المرجعية إلا انه لكثرة استخدامه يمكن التعامل معه بنفس الطريقة التي نتعامل معها مع الأنواع البسيطة .

ومن ضمن الأنواع المرجعية :

String -array-interface-classes-

مثال :

int [] a = new int [100];

**تعريف المتغير** (**declaring):**

لتعريف المتغير نقوم بتحديد نوعه واسمه

**مثال:**

*Int a;*

لإعطاء قيمة لمتغير موجود سابقا (Assign):

نقوم بكتابة اسم المتغير ثم إشارة الاسناد ثم القيمة

*a=5;*

هنا سنقوم بتعريف متغير جديد وإعطاءه قيمة مباشرة (Initialisation) :

int number = 5;

System.out.println(number);

كما يمكننا إنشاء متغيرات بدون اسناد قيمة لها ثم اعطاءها قيمة لاحقا

مثال :

int myNum;

myNum = 15;

System.out.println(myNum);

لاحظ اذا قمت بتعيين قيمة جديدة لمتغير موجود سابقا, فإنه سيتم الكتابة فوق القيمة السابقة :

لاحظ ستتغير قيمة myNum من 15 إلى 20 :

int myNum = 15;

myNum = 20; // myNum is now 20

System.out.println(myNum);

لتعريف أكثر من متغير لهم نفس النوع :

int x = 5, y = 6, z = 50;

**اختبر نفسك :**

\_\_\_\_\_ mynum=9;

\_\_\_\_ myfloatnum=8.99f;

\_\_\_\_\_ myletter=’a’;

\_\_\_\_\_ mybool=true;

\_\_\_\_\_ mytext=”hello”;

**تسمية المتغيرات :**

عند تعريف أي متغير يجب ان نقوم بتحديد اسم فريد له

تدعى هذه الأسماء identifiers

يمكن ان تكون هذه الأسماء قصيرة مثل x ,y او أسماء تصف شيء ما(age-name-sum)

يوصى دائما باستخدام أسماء تدل على الشيء حتى تكون جميع التعليمات البرمجية التي قمنا بكتابتها مفهومة .

**القواعد العامة لتسمية المتغيرات :**

-يمكن ان تحوي هذه الأسماء على حروف او ارقام او رمز \_او يرمز الدولار $

-يجب ان يبدأ الاسم بحرف صغير

-لا يمكن للمتغيرات ان تبدأ برقم

-الجافا حساسة لحالة الاحرف فمثلا total تختلف عن Total

-لا يمكن استخدام الكلمات المحجوزة كمعرفات مثل (int-boolean-…)

**اختبر نفسك :**

قم بإنشاء متغير اسمه carname واسند له القيمة volove

\_\_\_\_\_ \_\_\_\_\_ = \_\_\_\_\_\_;

**الثوابت :**

يمكننا استخدام الكلمة final اذا كنا لا نريد ان يقوم أي شخص بتغير قيمة المتغير الحالية

مثال:

final int myNum = 15;

**التحويل بين أنواع البيانات :**

عندما يتم اسناد قيمة من بيانات معين إلى متحول من نوع بيانات أخر .

يوجد نوعين :

1**-التحويل من النوع الضيق الى العريض**

هي عملية بسيطة لا تحتاج إلى تحويل صريح لأن النوع الواسع يستطيع تخزين القيمة من النوع الضيق دون أي ضياع للقيمة .

byte -> short -> char -> int -> long -> float -> double

public class Main {

public static void main(String[] args) {

int myInt = 9;

double myDouble = myInt; // Automatic casting: int to double

System.out.println(myInt); // Outputs 9

System.out.println(myDouble); // Outputs 9.0

}

}

2-**التحويل من النوع الواسع إلى النوع الضيق :**

يرافق هذا التحويل ضياع للقيمة بالتالي تشترط لغة الجافا القيام بهذا التحويل بشكل صريح إلا سيصدر المترجم خطأ

double -> float -> long -> int -> char -> short -> byte

public class Main {

public static void main(String[] args) {

double myDouble = 9.78d;

int myInt = (int) myDouble; // Manual casting: double to int

System.out.println(myDouble); // Outputs 9.78

System.out.println(myInt); // Outputs 9

}

}