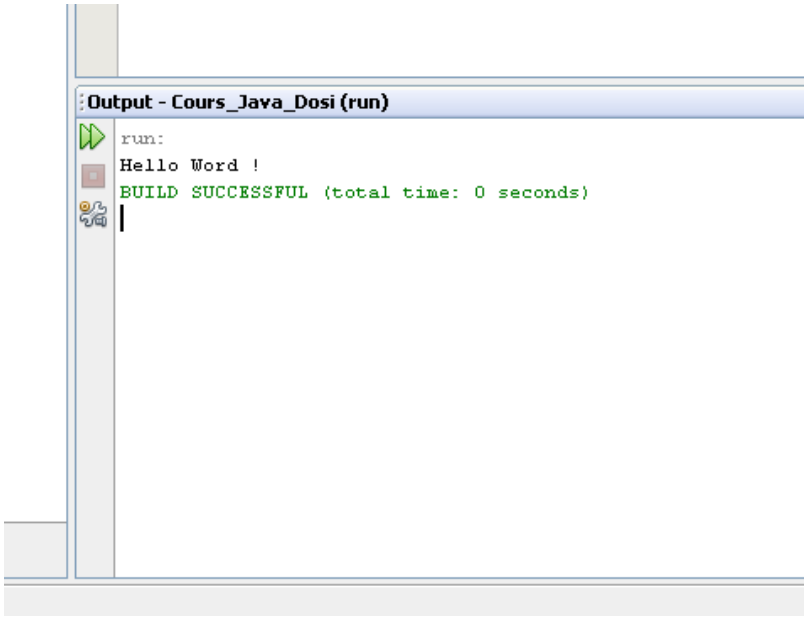
Java
Généralités

Ahcène Bounceur

université de bretagne occidentale

Premier exemple

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello Word !");
    }
}
```



Structure générale du programme

```
public class Main {



}
```

Structure générale du programme

```
public class Main {
    public static void main(String[] args) {

    }
}
```

Structure générale du programme

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello Word !");
    }
}
```

Exécution d'un programme Java

- Nom du fichier contenant le code java
  - Doit impérativement se nommer par le nom de la classe
  - Extension : **.java**
- Pour l'exemple, le nom du fichier doit être :

Main.java

Exécution d'un programme Java

- Compilation :
  - **Javac**
  - Générera un fichier **bytecode**
  - Extension : **.class**
- Pour l'exemple :
  - Compilation :

    javac Main.java
  - Fichier généré : **Main.class**

Exécution d'un programme Java

- Ces dernières notions sont vraies dans le cas où on utilise l'environnement de développement JDK (Java Development Kit ou Kit de Développement Java) de SUN

Exécution d'un programme Java

- Execution :

java Main

- **Attention :**
  - Il existe une différence entre Main (M majuscule) et main (m minuscule)
  - On pourrait appeler la classe Main différemment, par exemple, MonPrg
  - Par contre on ne peux pas changer les nom de la méthode main

Exécution d'un programme Java

- Les fichiers **JAR** (**J**ava **AR**chive)
  - C'est un fichier compressé (exemple : un zip)
  - Extension : .jar
  - Il contient :
    - Les fichier .class de l'application
    - un sous dossier META-INF
  - Le dossier META-INF contient un fichier MANIFEST.MF

Exécution d'un programme Java

- Le contenu du fichier MANIFEST.MF

```
Main-Class: nom_classe
Class-Path: \
```

- nom_classe : est le nom de la classe contenant le main
- \ : le chemin classpath (exemple : un repertoire ou un autre jar)

Instructions de base

Variables
Affectation
Affichage

```
public class Exemple {
   public static void main(String[] args) {
         int n;
         double x;
         n = 5;
         x = 2*n+1.5;
         System.out.println("n = "+n);
         System.out.println("x = "+x);
         double y;
         y = n * x + 12;
         System.out.println("Valeur de y : "+y);
   }
}
```

- Exécution :

```
n = 5
x = 11.5
Vleur de y : 69.5
```

Instructions de base

- Remarques
  - Aucun objet n'apparait dans ce programme
  - Attentino aux conversion de types
  - Attention à la concaténation "+"
    - Soit a=**3** et b=**5**
    - `System.out.println("y = "+a+b);`
      - `Execution : y = 35`
    - `System.out.println("y = "+(a+b));`
      - `Execution : y = 8`

Instructions de base

Lecture d'informations au clavier

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Test1 {
    public static void main(String [] args) throws IOException {
        int a;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        String s = br.readLine();
        a = Integer.valueOf(s);
        System.out.println("Le double de " + a + " est " + (a*2));
    }
}
```

Instructions de base

Boucles
et
Choix

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Table_Multiplication {
    public static void main(String [] args) throws IOException {
        int a;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        System.out.print("Table de : ");
        String s = br.readLine();
        a = Integer.valueOf(s);
        for (int i = 1; i <= 10; i++) {
            System.out.println(a + " x " + i + " = " + (a*i));
        }
    }
}
```

Instructions de base

Boucles
et
Choix

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Impair_Verif {
    public static void main(String [] args) throws IOException {
        int a;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        System.out.println("Donner un nombre impair : ");
        a = Integer.valueOf(br.readLine());
        while (a%2 != 0) {
            System.out.println(a + " est un nombre impair (Bravo !)");
            System.out.println("Donner un nombre impair : ");
            a = Integer.valueOf(br.readLine());
        }
        System.out.println("Non, " + a + " est un nombre pair (perdu).");
        System.out.println("A titre d'information, " + a + " = "+ (a/2) + " x 2");
    }
}
```

Instructions de base

Boucles
et
Choix

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class Impair_Verif2 {
    public static void main(String [] args) throws IOException {
        int a;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        boolean continuer = true;
        do{
            System.out.println("Donner un nombre impair : ");
            a = Integer.valueOf(br.readLine());
            if (a%2 != 0)
                System.out.println(a + " est un nombre impair (Bravo !)");
            else
                continuer = false;
        } while(continuer);
        System.out.println("Non, " + a + " est un nombre pair (perdu).");
        System.out.println("A titre d'information, " + a + " = "+ (a/2) + " x 2");
    }
}
```

Instructions de base

Boucles
et
Choix

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Impair_Verif2 {
    public static void main(String [] args) throws IOException {
        int a;
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        do{
            System.out.println("Donner un nombre impair : ");
            a = Integer.valueOf(br.readLine());
            if (a%2 != 0)
                System.out.println(a + " est un nombre impair (Bravo !)");
            else
                break;
        } while(true);
        System.out.println("Non, " + a + " est un nombre pair (perdu).");
        System.out.println("A titre d'information, " + a + " = "+ (a/2) + " x 2");
    }
}
```

Les différentes sortes d'instructions

- Instructions de déclaration
- Instructions exécutables
  - Simples (terminées par des ;)
  - De structuration (telles que if, for, etc.)
  - Blocs (délimité par { et })

Règles générales d'écriture

- Les identificateurs
  - les différentes entités manipulées par un programme :
    - Variables
    - Fonctions
    - Classes
    - Objets
    - …

Règles générales d'écriture

- Les identificateurs
  - Formé de lettres et/ou de chiffres
    - A-Z, a-z, 0-9, et le souligné (_)
  - On peut utiliser le $ mais il est déconseillé
  - Pas de limitation sur le nombre de caractères
  - On distingue les majuscules des minuscules
- Exemples :
  - Ligne, clavier, valeur_5, _total, _578, valeurMin
  - Ces identificateurs ne désignent pas la même varibale : **ligne**, **Ligne** et **liGne**

Règles générales d'écriture

- Les identificateurs : Conseil
  - Utiliser les minuscules pour identifier une variable : x, valeur, total, …
  - Mettez le premier caractère en majuscule pour définir les classes : Voiture, Personne, Porte, …
  - Si l'identificateur est composé de plusieurs mots juxtaposés, utiliser une lettre majuscule pour chaque début de mot sauf le premier : valeurMin, teauDeDef, maClasse, algoDeTri_2, …
  - Utiliser que des majuscules pour les constantes : PI, TAU, MU, ROUGE, DROITE, GAUCHE, …

Règles générales d'écriture

- Petit exercice :
  - Que définissent ces identificateurs :
    - Numero
    - lien_7
    - souris
    - LargeurMax
    - LONGUEUR_Max
    - CATA
    - CONS_TXT
    - VariableST_6

Règles générales d'écriture

- Les mots clés

abstract	**default**	**if**	**private**	**throw**
assert	**do**	**implements**	**protected**	**throws**
boolean	**double**	**import**	**public**	**transient**
break	**else**	**instanceof**	**return**	**true**
byte	**extends**	**int**	**short**	**try**
case	**false**	**interface**	**static**	**void**
catch	**final**	**long**	**strictfp**	**volatile**
char	**finally**	**native**	**super**	**while**
class	**float**	**new**	**switch**	
const	**for**	**null**	**synchronized**	
continue	**goto**	**package**	**this**	

Règles générales d'écriture

- Les séparateurs
  - **Espace**, **virgule** et le **point virgule**
  - Exemples :
    - `int x,y`
    - `int a,b,x,compte,p,valeur,total;`
    - `int a, b, x, compte, p, valeur, total ;`
    - ```
      int    a, b,
             x,
             compte, p,
             valeur, total ;
      ```
      ```

Règles générales d'écriture

- Le format libre
  - But : rendre le code visible
  - Attention : risque de rendre le code non visible
  - Une instruction peut être écrite sur plusieurs lignes
  - Un saut de ligne (ou fin de ligne) ne représente pas une fin d'instruction
  - Une fin d'instruction est marquée par un ' ; '
  - Un identificateur ne pouvant être coupé en deux par une fin de ligne

Règles générales d'écriture

- Les commentaires
  - Les commentaires usuels
  - Les commentaires de fin de ligne
  - *Les commentaires de documentation*

Règles générales d'écriture

- Les commentaires usuels
  - Placés entre : /* et */
  - Exemples :

```
/* Programme de tri */
/*  Ce programme utilise la méthode
    classique pour trier
    un tableau d'entiers
*/
/*-----------------------------------------------*
 *            Declaration des variables          *
 *-----------------------------------------------*/
int i ;     /* compteur de boucles */
float x ;   /* le resultat         */
```

Règles générales d'écriture

- Les commentaires de fin de ligne
  - Placés après : //
  - Exemples :

```
System.out.println("x = ");   // afficher la valeur de x
```

Règles générales d'écriture

- Les commentaires de documentation
  - Placés entre : **/**** et ***/**
  - Leur intérêt est de pouvoir être extraits automatiquement par des programmes utilitaires de création de documentation tels que **Javadoc**

Règles générales d'écriture

- Emploi du code UNICODE
  - Le but est d'utiliser un ensemble large de caractères. Ainsi, l'utilisation des lettres à accent est possible.
  - Exemple :
    - windows utilise le code ACII/ANSI Latin 1
      - Limitation : caractères codés sur 7 bits
    - Le codage Unicode est basé sur 2 octets
      - C'est-à-dire, 65536 combinaisons possibles
      - Donc, il est possible de représenter la plupart des symboles utilisés dans le monde

Règles générales d'écriture

- Emploi du code UNICODE
  - Pour finir :
    - int têteListe ; // déclaration correcte
  - **Attention :**
    - Bien qu'il est possible d'utiliser comme identifiant têteListe, c'est-à-dire, il n'yaura aucun problème de compilation,
      - Par de problème au niveau java
    - Mais, il peut y avoir de problème dans les éditeurs utilisés
    - *L'utilisation des accents est déconseillée (pour le moment)*

- Merci de votre attention