

– POO1 – Travaux Pratiques – Série N°1

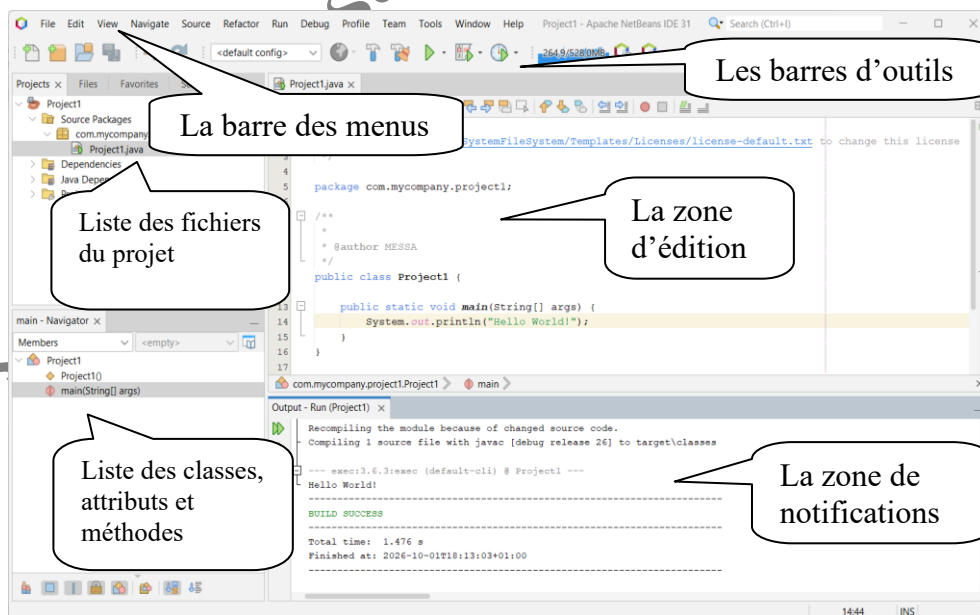
Introduction

- ✓ Java est un langage de programmation **orienté objet**, **fortement typé** et **multiplateforme**. Son code (.java) est compilé en **bytecode** (.class), exécuté par la machine virtuelle Java (**JVM**), ce qui permet de développer des applications indépendantes du système d'exploitation.
- ✓ Pour programmer en langage Java, il existe plusieurs **IDE** (**E**nvironnement de **D**éveloppement **I**ntégré). On peut citer **Netbeans**, **Eclipse**, **IntelliJ IDEA**, **Visual Studio Code**, ...
- ✓ Un **IDE** est un ensemble d'outils qui inclut un **éditeur** de texte, un **compilateur** et un **débogueur**. L'éditeur de texte permet d'écrire le programme source. Le compilateur vérifie si le programme écrit respecte la syntaxe du langage de programmation utilisé puis génère le programme exécutable. Le débogueur facilite la recherche d'erreurs de logique dans le programme. Il permet notamment l'exécution du programme pas-à-pas (instruction par instruction), l'affichage des valeurs des variables à tout moment,
- ✓ Nous choisissons de travailler avec Netbeans, les programmes écrits marchent parfaitement avec les autres IDEs.

Exercice 1 : Mon premier programme en langage Java

Écrire un programme qui affiche « Bonjour tout le monde ! ».

ETAPE 1 – Lancer le logiciel Netbeans en cliquant sur l'icône affichée sur le bureau de Windows. Vous apercevez la fenêtre de l'IDE suivante :



ETAPE 2 – Pour créer un programme Java avec Netbeans (Version 31), on peut procéder ainsi :

1. Cliquer sur le **menu File → New Project...** (ou utiliser le raccourcis **Ctrl+Shift+N**).
2. Dans **Categories**, choisir **Java with Maven**

3. Dans **Projects**, choisir **Java Application**
4. Cliquer sur **Next >**
5. Choisir un nom de projet (**Project Name**), par exemple **Projet1**.
6. Choisir le répertoire (**Project Location**) où le projet sera créé, par exemple un répertoire JAVA sur le Bureau de Windows.
7. Préciser (**Group Id**) le nom de l'entreprise ou le groupe de travail.
8. Préciser (**Version**) le numéro de la version du programme.
9. Préciser (**Package**) le nom du package.
10. Cliquer sur **Finish**. Un projet sera créé automatiquement avec la méthode **public static void main(String[] args)** qui est le point d'entrée du programme principal.
11. Compiler le projet : Cliquer sur le menu **Run → Build Project** ou cliquer sur l'icône correspondante dans la barre d'outils ou utiliser le raccourcis **F11**.
12. Exécuter le projet : Cliquer sur le menu **Run → Run Project** ou cliquer sur l'icône correspondante dans la barre d'outils ou utiliser le raccourcis **F6**.
13. Modifier le programme pour qu'il affiche, sur 3 lignes, votre nom, votre prénom et votre groupe.
14. Exécuter le projet de nouveau.

Exercice 2 : quelques opérations arithmétiques

Q1) Écrire un programme qui lit deux nombres entiers **a** et **b** et calcule leur somme.

Q2) Modifier le programme précédent pour qu'il calcule aussi :

- 1) La différence entre **a** et **b**.
- 2) La multiplication de **a** par **b**.
- 3) La division de **a** par **b**.
- 4) Le reste de division de **a** par **b** (c'est-à-dire **a % b**).

Q3) Pour le calcul de la division et du reste de la division, que se passe-t-il si la valeur de **b** est nulle ? Compléter votre programme en affichant un message d'erreur s'il y a besoin.

Exercice 3 : Min, Max

Q1) Écrire un programme java qui lit deux nombres réels **x1** et **x2** et calcule leur minimum.

Q2) Transformer ce programme en une méthode (fonction) **public static float min2(float x1, float x2)** et appeler la dans le programme principal (méthode **main()**).

Q3) Écrire une autre fonction **min3** qui permet de calculer le minimum de 3 nombres réels. Adapter votre programme.

Q4 - Optionnel) Adapter les fonctions précédentes ainsi que le programme principal pour afficher le **maximum** de trois réels.

Exercice 4 : SUITE if, else if, else if, ..., else

Écrire un programme Java qui lit la moyenne générale ($0 \leq \text{moyenne} \leq 20$) obtenue par un(e) étudiant(e) et affiche la mention correspondante :

- Si $\text{moyenne} < 10$, mention : **Insuffisant**.

- Si $10 \leq \text{moyenne} < 12$, mention : **Passable**.
- Si $12 \leq \text{moyenne} < 14$, mention : **Assez Bien**.
- Si $14 \leq \text{moyenne} < 16$, mention : **Bien**.
- Si $16 \leq \text{moyenne} < 18$, mention : **Très Bien**.
- Si $18 \leq \text{moyenne} \leq 20$, mention : **Excellent**.

Exercice 5 : switch

Une méthode pour trouver le jour (J) de la semaine d'une date (jj/mm/aaaa) est décrite ci-dessous :

Si $mm \geq 3$, $J = \{ [(23mm)/9] + jj + 4 + aaaa + [z/4] - [z/100] + [z/400] - 2 \} \bmod 7$

Si $mm < 3$, $J = \{ [(23mm)/9] + jj + 4 + aaaa + [z/4] - [z/100] + [z/400] \} \bmod 7$

J = Jour de semaine (J = 0 à 6; 0 = Dimanche, 1 = Lundi, ... , 6 = Samedi)

mm = Mois (mm = 1 à 12; 1 = Janvier, 2 = Février, 3 = Mars, ... , 12 = Décembre)

Si $mm < 3$ alors $z = aaaa - 1$

Si $mm \geq 3$ alors $z = aaaa$

/ est l'opérateur de division entière

jj = Jour (jj = 1 à 31)

aaaa = année

$\bmod 7$ = Modulo 7 = Reste de la division entière par 7

Exemple : Quel jour de la semaine tombait le 01 novembre 1954 ? Réponse : c'est un **Lundi**.

jj = 1, mm = 11, aaaa = 1954 et $z = aaaa = 1954$ (parce que $mm \geq 3$); d'où :

$J = \{ [(23 \times 11)/9] + 1 + 4 + 1954 + [1954/4] - [1954/100] + [1954/400] - 2 \} \bmod 7 \rightarrow J = 1 \rightarrow$ un **Lundi**.

Q1) Écrire une méthode (fonction) qui permet de trouver le jour de la semaine d'une date.

Q2) Compléter le programme précédent pour qu'il affiche le nom du jour correspondant.

C'est-à-dire 0 $\rightarrow$ "Dimanche", 1 $\rightarrow$ "Lundi", ... , 6 $\rightarrow$ "Samedi".

Exercice 6 : Nombres Parfaits

Q1) Ecrire un programme Java qui permet d'afficher tous les diviseurs d'un entier N.

Un nombre est dit parfait s'il est égal à la somme de tous ses diviseurs excepté lui-même.

Exemples : • 6 est parfait car $6 = 1 + 2 + 3$. Les diviseurs de 6 sont : 1, 2, 3 et 6 (exclu).

• 28 est parfait car $28 = 1 + 2 + 4 + 7 + 14$. Les diviseurs de 28 sont : 1, 2, 4, 7, 14 et 28 (exclu).

Q2) Ecrire une méthode (fonction) **public static boolean perfect(int n)** qui permet de vérifier si un entier N est parfait ou pas. Adapter le programme principal.

Q3) Généraliser le programme précédent pour afficher tous les nombres parfaits $\leq N_{\text{Max}}$.

Exercice 7 : Nombres Premiers

Un nombre est dit premier s'il n'admet que deux diviseurs : 1 et lui-même.

Q1) Ecrire une méthode (fonction) **public static boolean prime(int n)** qui permet de vérifier si un entier n est premier ou pas. **Note :** la racine carrée est obtenue en utilisant la méthode **Math.sqrt()**.

Q2) Ecrire une autre méthode (procédure) qui permet d'afficher les m petits nombres premiers.

Exemple : Si $m = 8$, la procédure doit afficher : 2 3 5 7 11 13 17 19.

Q3) Ecrire un programme Java qui permet de lire un entier m et d'afficher les m petits nombres premiers.

Exercice 8 : Racine carrée

Pour calculer une valeur approchée de la racine carrée de n'importe quel réel positif a , on construit la

$$\text{suite } (x_n) \text{ définie comme suit : } \begin{cases} x_0 = 1 & n = 0 \\ x_n = \frac{(x_{n-1})^2 + a}{2x_{n-1}} & n = 1, 2, \dots \end{cases}$$

On arrête le calcul des termes de la suite à l'étape n (n est inconnu) tel que $|a - (x_n)^2| \leq e$, où e est la tolérance de l'erreur (on prendra $e = 10^{-6}$). Cette valeur x_n est la valeur approchée de la racine carrée de a .

Note : Une constante est définie par le mot clé **final** (ex : ***final static double PRECISION = 1.0E-6;***)

Q1) Ecrire une méthode ***uSqrt(double a)*** qui calcule la racine carrée d'un réel positif a par cette méthode.

Note : la valeur absolue est obtenue en utilisant la méthode ***Math.abs()***.

Q2) Ecrire un programme Java qui permet de :

- ①- Lire un nombre réel positif x ,
- ②- Calculer la racine carrée par la méthode créée en Q1
- ③- Comparer les résultats de ***uSqrt()*** et ***Math.sqrt()***.

Exercice 9 : Nombres Symétriques

Soit N un nombre entier positif.

Q1) Ecrire un programme Java qui permet d'afficher les chiffres qui composent le nombre N ainsi que sa longueur. **Exemples** : - Si $N = 17 \rightarrow$ on affiche les chiffres 7 puis 1 et La longueur = 2.

- Si $N = 695 \rightarrow$ on affiche les chiffres 5 puis 9 puis 6 et La longueur = 3.

Q2) Ecrire une méthode ***public static int mirror(int n)*** qui permet de calculer l'image miroir de N .

Exemple : Si $N = 695 \rightarrow$ son image miroir = 596.

Un nombre N est dit **symétrique** s'il est égal à son image miroir.

Exemples : Les nombres suivants sont symétriques : 1, 2, 3, 44, 55, 161, 717, 8228, 94549.

Q3) Ecrire une méthode ***symetric(int n)*** qui permet de vérifier si un entier n est symétrique ou pas.

Q4) Modifier le programme précédent pour afficher un message indiquant si n est symétrique ou pas.

Q5) Ecrire une méthode ***display(int lg)*** qui permet d'afficher tous les nombres symétriques d'une longueur lg donnée.

Q6) Modifier le programme principal en conséquence.