3. TP 1 : Introduction à Java

Objectifs

- Comprendre les types de données en Java.
- Manipuler les fonctions d'affichage (`print` et `println`).
- Découvrir les structures de contrôle (`if/else`, `switch`).
- Utiliser les boucles (`for`, `while`).

Partie 1 : Questions guidées avec réponses

3.0.1 Types de données et affichage

Question 1 :

Comment déclarer une variable de type entier (`int`), et afficher sa valeur dans la console ?

Réponse :

```
public class Main {
    public static void main(String[] args) {
        int nombre = 42;
        System.out.println("La valeur de nombre est : " +
    nombre);
    }
}
```

Question 2 :

Déclarez une variable de type `double` et affichez-la avec une description.

Réponse :

```java
public class Main {
    public static void main(String[] args) {
        double pi = 3.14159;
        System.out.println("La valeur de pi est : " + pi);
    }
}
```

Question 3 :

Comment concaténer du texte avec des variables pour les afficher ?

Réponse :

```java
public class Main {
    public static void main(String[] args) {
        String prenom = "Amina";
        int age = 25;
        System.out.println("Bonjour, je m'appelle " + prenom +
    " et j'ai " + age + " ans.");
    }
}
```

3.0.2 Constantes

Question 1 :

Comment déclarer une constante en Java avec le mot-clé `final` ?

Réponse :

```java
public class Constantes {
    public static void main(String[] args) {
        // Déclaration d'une constante
        final double PI = 3.14159;

        // Utilisation de la constante
        double rayon = 5.0;
        double circonference = 2 * PI * rayon;
        System.out.println("La circonférence est : " +
   circonference);

        // PI = 3.14; // Cette ligne provoquerait une erreur
   de compilation
    }
}
```

Question 2 :

Quelle est l'utilité du mot-clé `final` en Java ?

Réponse :

— Le mot-clé `final` rend une variable constante, c'est-à-dire que sa valeur ne peut être assignée qu'une seule fois.
— Cela permet d'éviter les modifications accidentelles des valeurs qui doivent rester constantes tout au long du programme.
— Les constantes améliorent la lisibilité et la maintenabilité du code.

3.0.3 Structures de contrôle (`if/else`, `switch`)

Question 1 :

Comment utiliser une condition simple avec `if` pour vérifier si un nombre est positif ou négatif ?

Réponse :

```
public class Main {
    public static void main(String[] args) {
        int nombre = -5;
        if (nombre > 0) {
            System.out.println("Le nombre est positif.");
        } else {
            System.out.println("Le nombre est négatif.");
        }
    }
}
```

Question 2 :

Utilisez une structure `switch` pour afficher un message selon une note (A, B, C, D, F).

Réponse :

```
public class Main {
    public static void main(String[] args) {
        char grade = 'B';
        switch (grade) {
            case 'A':
                System.out.println("Excellent !");
                break;
            case 'B':
                System.out.println("Très bien !");
                break;
            case 'C':
                System.out.println("Bien.");
                break;
            case 'D':
                System.out.println("Passable.");
                break;
            case 'F':
                System.out.println("Échec.");
                break;
            default:
```

```
                System.out.println("Note inconnue.");
                break;
            }
        }
}
```

3.0.4 Boucles (`for`, `while`) et `do-while`

Question 1 :

Comment afficher les nombres de 1 à 5 avec une boucle `for` ?

Réponse :

```
public class Main {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++) {
            System.out.println(i);
        }
    }
}
```

Question 2 :

Utilisez une boucle `while` pour afficher les nombres de 10 à 1 en ordre décroissant.

Réponse :

```
public class Main {
    public static void main(String[] args) {
        int i = 10;
        while (i > 0) {
            System.out.println(i);
            i--;
        }
    }
}
```

Question 3 :

Comment utiliser une boucle `for` pour afficher les nombres pairs de 0 à 10 ?

Réponse :

```
public class BoucleFor {
    public static void main(String[] args) {
        for (int i = 0; i <= 10; i += 2) {
            System.out.println(i);
        }
    }
}
```

Question 4 :

Comment fonctionne une boucle `do-while` et quelle est la différence avec une boucle `while` ?

Réponse :

— La boucle `do-while` exécute le bloc de code au moins une fois avant de vérifier la condition.

— Syntaxe :

```
    do {
        // Instructions
    } while (condition);

```

— Exemple :

```
    public class BoucleDoWhile {
        public static void main(String[] args) {
            int i = 0;
            do {
                System.out.println("i = " + i);
                i++;
            } while (i < 5);
        }
    }

```

— Contrairement à la boucle `while`, où la condition est vérifiée avant l'exécution du bloc de code, la boucle `do-while` garantit au moins une exécution du bloc.

Question 5 :

Écrivez un programme qui utilise une boucle `do-while` pour demander à l'utilisateur de deviner un nombre secret jusqu'à ce qu'il le trouve.

Réponse :

```
import java.util.Scanner;

public class DevinerNombre {
    public static void main(String[] args) {
        int nombreSecret = 7;
        int nombreUtilisateur;
        Scanner scanner = new Scanner(System.in);

        do {
            System.out.print("Devinez le nombre entre 1 et 10 : ");
            nombreUtilisateur = scanner.nextInt();
        } while (nombreUtilisateur != nombreSecret);

        System.out.println("Bravo ! Vous avez trouvé le nombre secret.");
```

```
        scanner.close();
    }
}
```

3.0.5 Instructions de Contrôle de Boucle

Question 1 :

Comment utiliser l'instruction `break` pour sortir prématurément d'une boucle ?

Réponse :

```
public class UtilisationBreak {
    public static void main(String[] args) {
        for (int i = 1; i <= 10; i++) {
            if (i == 5) {
                break; // Sort de la boucle lorsque i vaut 5
            }
            System.out.println("i = " + i);
        }
        System.out.println("Boucle terminée.");
    }
}
```

Question 2 :

Comment l'instruction `continue` affecte-t-elle le flux d'une boucle ?

Réponse :

— L'instruction `continue` passe immédiatement à l'itération suivante de la boucle, en sautant le code restant dans le corps de la boucle pour l'itération en cours.

— Exemple :

```
    public class UtilisationContinue {
        public static void main(String[] args) {
            for (int i = 1; i <= 5; i++) {
                if (i == 3) {
                    continue; // Saute l'affichage lorsque
    i vaut 3
                }
                System.out.println("i = " + i);
            }
        }
    }
```

Question 3 :

Donnez un exemple où l'utilisation de `break` et `continue` est appropriée.

Réponse :

— **Utilisation de** `break` : Recherche d'un élément dans un tableau. Une fois l'élément trouvé, on peut sortir de la boucle.

```
    int[] nombres = {2, 4, 6, 8, 10};
    int aChercher = 6;
    boolean trouve = false;

    for (int i = 0; i < nombres.length; i++) {
        if (nombres[i] == aChercher) {
            trouve = true;
            break; // Sort de la boucle une fois l'élément trouvé
        }
    }

    if (trouve) {
        System.out.println(aChercher + " a été trouvé dans le tableau.");
    } else {
        System.out.println(aChercher + " n'est pas dans le tableau.");
    }

```

— **Utilisation de** `continue` : Parcourir les nombres de 1 à 10 et afficher uniquement les nombres impairs.

```
    for (int i = 1; i <= 10; i++) {
        if (i % 2 == 0) {
            continue; // Saute les nombres pairs
        }
        System.out.println("Nombre impair : " + i);
    }

```

Partie 2 : Exercices sans réponses

3.0.6 Types de données et affichage

Exercice 1 :

Déclarez une variable `float` pour stocker une température, puis affichez-la avec un message expliquant s'il fait chaud ou froid (si la température est supérieure à 25°C, il fait chaud).

Exercice 2 :

Déclarez une variable `boolean` pour indiquer si une porte est ouverte (`true`) ou fermée (`false`), puis affichez le message correspondant.

3.0.7 Constantes

Exercice 3 :

Déclarez une constante pour le taux de TVA (par exemple, `final double TVA = 0.2;`) et écrivez un programme qui calcule le prix TTC d'un produit en fonction de son prix HT.

Exercice 4 :

Créez une constante représentant le nombre maximum d'utilisateurs autorisés dans une application. Écrivez un programme qui compare le nombre actuel d'utilisateurs avec cette constante et affiche un message approprié.

3.0.8 Structures de contrôle

Exercice 5 :

Écrivez un programme qui demande à l'utilisateur d'entrer un nombre, et affiche un message si le nombre est pair ou impair.

Exercice 6 :

Utilisez une structure `switch` pour afficher un message en fonction du jour de la semaine (Lundi, Mardi, Mercredi, etc.).

3.0.9 Boucles

Exercice 7 :

Écrivez un programme qui affiche la table de multiplication de 5 (de 1 à 10).

Exercice 8 :

Utilisez une boucle `while` pour calculer et afficher la somme des nombres de 1 à 50.

Exercice 9 :

Créez une boucle qui demande à l'utilisateur d'entrer un nombre, et continue jusqu'à ce que l'utilisateur entre un nombre négatif.

Exercice 10 :

Écrivez un programme qui utilise une boucle `for` pour calculer et afficher la factorielle d'un nombre entier positif saisi par l'utilisateur.

Exercice 11 :

Utilisez une boucle `do-while` pour demander à l'utilisateur de saisir un mot de passe jusqu'à ce qu'il entre le mot de passe correct (par exemple, "Java123").

Exercice 12 :

Créez un programme qui affiche les multiples de 3 entre 1 et 30 en utilisant une boucle `for`.

3.0.10 Instructions de Contrôle de Boucle

Exercice 13 :

Écrivez un programme qui parcourt les nombres de 1 à 20 et utilise `continue` pour sauter l'affichage des multiples de 5.

Exercice 14 :

Créez un programme qui lit une série de nombres entrés par l'utilisateur et s'arrête lorsque l'utilisateur entre un nombre négatif, en utilisant `break`.

Exercice 15 :

Élaborez un programme qui parcourt les lettres de l'alphabet et utilise `continue` pour sauter les voyelles, n'affichant que les consonnes.