```java
/*
 *  Chapitre 04
 *  Sujet : Les opérations sur les variables en Java
 */

public class Operations {
    public static void main(String[] args) {
        /*
            Les opérations que l'on peut faire sur les variables sont :
            - Les opérations de concaténation sur les chaînes de caractères (String).
            - Les opérations mathématiques sur les nombres;
            - L'inversion booléenne
         */

        // La concaténation de chaînes de caractères
        String chaineConcatenee = "Bonjour" + " tout le monde";
        chaineConcatenee = chaineConcatenee + '!';

        // Comme les String sont immuables, la valeur de la chaîne n'est pas modifiée
        // Une nouvelle chaîne est créée et elle correspond à la valeur attendue

        // Les opérations mathématiques permises sont :
        // 1. L'addition
        int valeurCalcul = 1 + 1;
        valeurCalcul = valeurCalcul + 2;
        valeurCalcul = 3 + valeurCalcul;
        valeurCalcul = valeurCalcul + valeurCalcul;
        valeurCalcul += 5; // Il s'agit d'un raccourci correspondant à -> valeurCalcul = valeurCalcul + 5;

        // 2. La soustraction
        valeurCalcul = 5 - 2;
        valeurCalcul = valeurCalcul - 1;
        valeurCalcul = 7 - valeurCalcul;
        valeurCalcul = - valeurCalcul; // Opération pour changer de signe
        valeurCalcul -= 3; // Il s'agit d'un raccourci correspondant à -> valeurCalcul = valeurCalcul - 3;

        // 3. La multiplication
        valeurCalcul = 2 * 3;
        valeurCalcul = valeurCalcul * 2;
        valeurCalcul = 3 * valeurCalcul;
        valeurCalcul = valeurCalcul * valeurCalcul;
        valeurCalcul = valeurCalcul * -1; // Opération pour changer de signe
        valeurCalcul *= 10; // Il s'agit d'un raccourci correspondant à -> valeurCalcul = valeurCalcul * 10;

        // 4. La division
        valeurCalcul = 20 / 2;
        valeurCalcul = valeurCalcul / 5;
        valeurCalcul = 20 / valeurCalcul;
        valeurCalcul = valeurCalcul / valeurCalcul;
        valeurCalcul = valeurCalcul / -1; // Opération pour changer de signe
        valeurCalcul /= 3; // Il s'agit d'un raccourci correspondant à -> valeurCalcul = valeurCalcul / 3;
        // valeurCalcul = 5 / 0; // Attention la division par zéro engendrera une erreur

        // En ce qui concerne la division, un nombre entier divisé par un nombre entier donnera un nombre entier
        System.out.println("5 / 3 = " + 5/3); // La réponse est 1

        // 5. Le modulo => reste de la division entière
        valeurCalcul = 10 % 4; // valeurCalcul est maintenant de 2
        valeurCalcul %= 3; // Il s'agit d'un raccourci correspondant à -> valeurCalcul = valeurCalcul % 3;
        // valeurCalcul = 11 % 0; // Attention la division par zéro engendrera une erreur

        // 6. L'inversion binaire ~ (NOT) (inversion de tous les bits)
```

```
63          byte valeurByte = ~1; // 1 = 00000001 ; ~1 = 11111110
64
65          // 7. Le ET binaire & (AND)
66          valeurByte = 3 & 10; // 00000011 & 00001010 = 00000010 (2)
67
68          // 8. Le OU binaire | (OR)
69          valeurByte = 3 | 10; // 00000011 | 00001010 = 00001011 (11)
70
71          // 9. Le OU Exclusif binaire ^ (XOR)
72          valeurByte = 10 ^ 3; // 00001010 ^ 00000011 = 00001001 (9)
73
74          // 10. Le décalage à gauche (la valeur des bits décale vers la gauche et les
    bits de droite sont mis à 0)
75          valeurCalcul = 0b00000000_00000000_00000000_00001101;
76          // a) 0b indique que ce qui suit est en représentation binaire
77          // b) le souligné _ est utilisé dans les nombres pour la simplification de
    lecture (grand nombre, numéro de carte bancaire,...)
78
79          valeurCalcul = valeurCalcul << 8; // valeurCalcul =
    0b00000000_00000000_00001101_0000
80          valeurCalcul <<= 8; // Il s'agit d'un raccourci équivalant à la ligne précédente
81
82          // Attention le bit tout à gauche est le bit de signe, si il change de valeur,
    le signe change pour le nombre
83
84          // 11. Le décalage à droite avec conservation de signe : la valeur des bits
    décalent vers la droite et les bits de gauche
85          // sont mis à zéro (sauf le bit de signe qui conserve sa valeur)
86
87          valeurCalcul = valeurCalcul >> 8;
88          valeurCalcul >>= 8; // Il s'agit d'un raccourci équivalant à la ligne précédente
89
90          // 12. Le décalage à droite sans conservation de signe : la valeur des bits
    décalent vers la droite et les bits de gauche
91          // sont mis à zéro (même le bit de signe)
92
93          valeurCalcul = valeurCalcul >>> 8;
94          valeurCalcul >>>= 8; // Il s'agit d'un raccourci équivalant à la ligne
    précédente
95
96          // L'inversion booléenne consiste à donner la valeur inverse du couple true/
    false
97          boolean valeurBoolean = true;
98          boolean valeurInversion = !valeurBoolean; // valeurInversion = false
99          valeurInversion = !true; // valeurInversion = false
100         valeurBoolean = !valeurBoolean; // Inversion de la valeur de valeurBoolean
101
102         // Il est également possible d'écrire un nombre hexadécimal
103         valeurCalcul = 0x1010_AC0F;
104         // a) 0x indique que ce qui suit est en représentation hexadécimale
105         // b) le souligné _ est utilisé dans les nombres pour la simplification de
    lecture (grand nombre, numéro de carte bancaire,...)
106     }
107 }
108
```