Základní konstrukce Javy

Aneb základy psaní programů

Celočíselné operace

Napište metodu, které sečte, odečte, vynásobí, vydělí a zjistí zbytek po dělení dvou čísel.
Použijte čísla:

- 25 a 10
- 13 a 0

Celočíselné operace - řešení


```
static void celociselnaArtimetika() {
    System.out.println("Celociselne operace.");
    int a = 25;
    //int a = 13;
    int b = 10;
    // int b = 0;
    System.out.printf("a + b = %1$d; a - b = %2$d; a * b = %3$d; "
            + "a / b = %4$d; a %% b = %5$d%n", a + b, a - b, a * b, a / b, a % b);
    System.out.println("");
}
```

Datové typy

Napište metodu, která sečte 2 čísla typu byte a součet (také typu byte) vypíše na obrazovku. Metodu spusťte pro čísla:

- 12 a 35
- 100 a 101
- -95 a -90

Čísla k sečtení budou argumenty metody.

Nápověda:
Výchozí typ součtu je int, proveďte přetypování příkazem (byte) (a + b)

Datové typy - řešení


```
 static void soucet(byte a, byte b) {
      System.out.println("Soucet 2 cisel.");
      byte c;
      c = (byte) (a + b);
      System.out.println(a + " + " + b +" = " + c);
      System.out.println("");
 }

public static void main(String[] args) {
     soucet((byte) 12, (byte) 35);
     soucet((byte) 100, (byte) 101);
     soucet((byte) -95, (byte) -90);
}
```

Pole

Napište metodu, která vytvoří pole o 10 prvcích, naplní je hodnotami od 0 do 9 a následně vypíše obsah pole na std. výstup. Pro výpis použijte příkaz for(int cislo:pole)

Tip: Po deklaraci pole napište fori a stiskněte tabulátor. Netbeans za vás vyplní správně syntaxi cyklu for.

Pole - řešení


```
static void cyklus() {
    System.out.println("Prace s polem, jeho naplneni a prochazeni.");
    int[] pole = new int[10];
    for (int i = 0; i < pole.length; i++) {
        pole[i] = i;
    }
    for (int cislo : pole) {
        System.out.print(cislo + " ");
    }
    System.out.println("\n");
}
```

Minimum maximum

Napište metodu, která vytvoří pole, vloží do něj hodnoty a pak nalezne maximum a minimum a zjistí jejich pozice.

Tip: Pro generování hodnot využijte náhodná čísla z knihovny Math. Př.

```
int a = (int) (Math.random() * 10);
```

uloží do proměnné a čísla v intervalu <0,10).

Minimum maximum - řešení


```
static void minMax() {
    System.out.println("Prace s poli - hledani minima a maxima.");
    int[] pole = new int[10];
    for (int i = 0; i < pole.length; i++) {
        pole[i] = (int) (Math.random() * 10);
    }
    int min = 11, max = -1;
    int pozMin = -1, pozMax = -1;
    for (int i = 0; i < pole.length; i++) {
        int j = pole[i];
        System.out.println("pole[" + i + "] = " + j);
        if (j < min) {
            min = j;
            pozMin = i;
        }
        if (j > max) {
            max = j;
            pozMax = i;
        }
    }
    System.out.println("Minimum bylo pole[" + pozMin + "] s hodnotou " + min);
    System.out.println("Maximum bylo pole[" + pozMax + "] s hodnotou " + max);
    System.out.println("");
}
```

Binární operace

Napište metodu, která provede tyto operace:

- 13 << 1 (shift left)
- 13 & 26 (and)
- 13 | 26 (or)
- 13 ^ 26 (xor)

Výsledky vypište jak decimálně tak i v binární podobě.

Nápověda:
Integer.toBinaryString(9) vrací 1001.

Binární operace - řešení


```
static void binary() {
    int a = 13;
    System.out.println("a = " + a + " =  " + Integer.toBinaryString(a));
    // operace posunuti vlevo o 1 bit
    int b = a << 1;
    System.out.println("b = " + b + " = " + Integer.toBinaryString(b));
    // operace binarni AND
    int c = a & b;
    System.out.println("& = " + c + " =   " + Integer.toBinaryString(c));
    // operace binarni OR
    c = a | b;
    System.out.println("| = " + c + " = " + Integer.toBinaryString(c));
    // operace binarni XOR
    c = a ^ b;
    System.out.println("^ = " + c + " = " + Integer.toBinaryString(c));

}
```

Otázky

....?