

TÖL101G - Tölvunarfræði 1

Vikublað 1

Lausn

Æfingar

1.2.10

- a `System.out.println(a)` ; prentar út 2147483647, gildið á a
- b `System.out.println(a+1)` ; prentar út -2147483648, því heiltölurnar fara í hring, einu hærra en hæsta gildið verður lægsta gildið á `int`.
- c `System.out.println(2-a)` ; prentar út -2147483645, “rétt” svar
- d `System.out.println(-2-a)` ; prentar út 2147483647,
- e `System.out.println(2*a)` ; prentar út -2
- f `System.out.println(4*a)` ; prentar út -4

Öll svörin eru rétt ef við reiknum svörin $\text{mod } 2^{31}$ með formerkjum. T.d. er $a = 2^{31} - 1$ svo

$$2 * a = 2 * (2^{31} - 1) = 2^{32} - 2 = -2$$

1.2.11

- a `System.out.println(a)` ; prentar 3.14159
- b `System.out.println(a+1)` ; prentar 4.14159
- c `System.out.println(8/(int) a)` ; prentar 2, `(int) a` skilar gildinu 3 (af taginu `int`) og $8/3$ er 2.
- d `System.out.println(8/a)` ; prentar 2.5464812403910124, heiltalan 8 er kastað yfir í fleytitöluna 8.0
- e `System.out.println((int) (8/a))` ; prentar 2, útkomunni í lið d er kastað yfir í heiltölu og námundað í átt að 0

Cloudcoder æfingar

1.2.15

```
public class TriangleTest {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
```

```

    int c = Integer.parseInt(args[2]);

    boolean triangle = (a >= b + c);
    triangle = triangle || (b >= a + c);
    triangle = triangle || (c >= a + b);

    System.out.println(triangle);
}
}

```

1.2.28

```

public class OrderTest {
    public static void main(String[] args) {
        int x = Integer.parseInt(args[0]);
        int y = Integer.parseInt(args[1]);
        int z = Integer.parseInt(args[2]);

        boolean t = (x < y && y < z) || (x > y && y > z);

        System.out.println(t);
    }
}

```

Verkefni

Lausn 1

Þessi lausn byggir á því að bera saman gildin tvö og tvö og raða í rétta röð. Ef að tölurnar eru a, b, c þá er a, b raðað fyrst í rétta röð og útkomurnar bornar saman við c . Fyrir fleiri en þrjár tölur verður þetta flóknara og þá þarf að nota svokölluð Sorting Networks.

```

public class ThreeSorted_v1 {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
        int c = Integer.parseInt(args[2]);

        // min_ab and max_ab put a and b in sorted order
        int min_ab = Math.min(a, b);
        int max_ab = Math.max(a, b);

        // max_abc is the maximum, d is either the middle or the min
        int max_abc = Math.max(max_ab, c);
        int d = Math.min(max_ab, c);

        // now d and min_ab are the minimum and mid numbers,
        // put them in sorted order
    }
}

```

```

    int min_abc = Math.min(min_ab, d);
    int mid_abc = Math.max(min_ab, d);

    System.out.println(min_abc + " " + mid_abc + " " + max_abc);
}
}

```

Lausn 2

Þessi lausn virkar aðeins fyrir þrjár tölur og einungis fyrir heiltölur. Stærsta og minnsta talan er fundin eins og áður og miðtalan er reiknuð út frá summunni $a + b + c$, þegar max og min eru dregnar frá stendur miðtalan eftir. Athugið að það má reikna max út í einu skrefi með $\text{Math.max}(\text{Math.max}(a, b), c)$ þá er útkoman úr fyrsta $\text{Math.max}(a, b)$ kallinu notuð sem viðfang (parameter) í næsta kalli á Math.max

```

public class ThreeSorted_v2 {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
        int c = Integer.parseInt(args[2]);

        int sum = a + b + c;

        int max = Math.max(Math.max(a, b), c);
        int min = Math.min(Math.min(a, b), c);

        int mid = sum - max - min;

        System.out.println(min + " " + mid + " " + max);
    }
}

```