

OPMS 2018 – Exercise 2

Task 1 (Warm-Up)

Determine the type and result of the following expressions!

1. `false && true`
2. `!true`
3. `false && false`
4. `false && true || true`
5. `true && false == false`
6. `20 / 3`
7. `20 % 3`
8. `20.0 / 3.0`

Task 2 (Warm-Up)

What do the following programs print on the console? **You are not allowed to use Eclipse during this task!**

a)

```
public class Task2a {
    public static void main(String[] args) {
        int first = 2;
        int second = 5;

        for (int i = 0; i < 2; i++) {
            first = second / first;
            second++;
            first = second * first;
        }
        System.out.println("first: " + first);
    }
}
```

b)

```
public class Task2b {  
    public static void main(String[] args) {  
        int a = 4;  
        int b = 3;  
        for (int i = 0; i <= a; i++){  
            int n = 0;  
            if (i == 2)  
                break;  
            b = i + b;  
        }  
        System.out.println(b);  
    }  
}
```

Task 3 (Basics)

In the following task, you have to answer the given questions. You can note your answers on a piece of paper or in a text document. **You are not allowed to use Eclipse during this task!**

- a) What is the return value of the following function given parameters a=9 and b=2?

```
int function(int a, int b) {  
    if (a < b)  
        return a * b;  
    else  
        return a / b;  
}
```

- b) What is the return value of the following function given parameters a=9.0 and b=3?

```
double function_2(double a, int b) {  
    while (b > 1) {  
        a = (int) (a / b);  
        b--;  
    }  
    return a;  
}
```

- c) What is the return value of the following function given parameters a=3 and b=2?

```
int function_3(int a, int b) {  
    switch (a + b) {  
        case 10:  
            a = b;  
        case 5:  
            a = a * b;  
        case 3:  
            a = a / b;  
            break;  
        default:  
            a = 0;  
    }  
    return a;  
}
```

- d) The following method (which checks if a given number is prime) contains one syntax error. What is the error and in which line is it?

```
1    boolean is_primenumber(int n) {  
2        if (n < 2)  
3            return false;  
4        for (int i = n / 2; i > 1; i--) {  
5            if (n % i == 0)  
6                return false;  
7        }  
8        return "Yes!";  
9    }
```

- e) What does the following method compute?

```
int function(int[] x) {  
    int y = x[0];  
    for (int i = 1; i < x.length; i++) {  
        if (x[i] < y) {  
            y = x[i];  
        }  
    }  
    return y;  
}
```

- f) In what way do you have to augment the following method at the blank position so that the method computes if the given number is even?

```
boolean is_even(int x) {  
    return ____ == 0;  
}
```

Task 4 (Basics)

1. Create a new project called “Exercise2”.
2. In the new project, create a new class called “Task4”.
3. Implement an application that prints all odd numbers between 0 and 1000 to the console. Thereby, each number has to be separated by a comma. After each ten printed numbers, a new line should be printed to the console. Your output should look like as depicted in Picture 1.

Hints: Use the modulo operator % (to determine the remainder of a division of two integers). Use a for-loop to do the counting (see slide 56) and branches to control your flow (see slide 49).

Extension: Extend your code, so that no empty line is printed at first to the console.

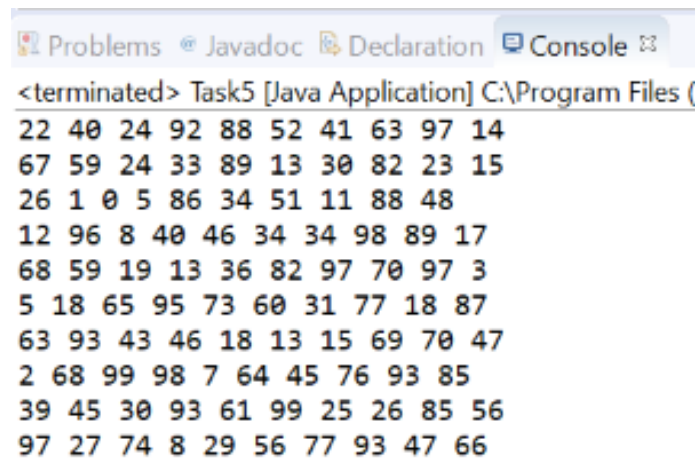
```
1, 3, 5, 7, 9, 11, 13, 15, 17, 19,
21, 23, 25, 27, 29, 31, 33, 35, 37, 39,
41, 43, 45, 47, 49, 51, 53, 55, 57, 59,
61, 63, 65, 67, 69, 71, 73, 75, 77, 79,
81, 83, 85, 87, 89, 91, 93, 95, 97, 99,
101, 103, 105, 107, 109, 111, 113, 115, 117, 119,
121, 123, 125, 127, 129, 131, 133, 135, 137, 139,
141, 143, 145, 147, 149, 151, 153, 155, 157, 159,
161, 163, 165, 167, 169, 171, 173, 175, 177, 179,
181, 183, 185, 187, 189, 191, 193, 195, 197, 199,
201, 203, 205, 207, 209, 211, 213, 215, 217, 219,
221, 223, 225, 227, 229, 231, 233, 235, 237, 239,
241, 243, 245, 247, 249, 251, 253, 255, 257, 259,
261, 263, 265, 267, 269, 271, 273, 275, 277, 279,
281, 283, 285, 287, 289, 291, 293, 295, 297, 299,
301, 303, 305, 307, 309, 311, 313, 315, 317, 319,
321, 323, 325, 327, 329, 331, 333, 335, 337, 339,
341, 343, 345, 347, 349, 351, 353, 355, 357, 359,
361, 363, 365, 367, 369, 371, 373, 375, 377, 379,
381, 383, 385, 387, 389, 391, 393, 395, 397, 399,
401, 403, 405, 407, 409, 411, 413, 415, 417, 419,
421, 423, 425, 427, 429, 431, 433, 435, 437, 439,
441, 443, 445, 447, 449, 451, 453, 455, 457, 459,
461, 463, 465, 467, 469, 471, 473, 475, 477, 479,
481, 483, 485, 487, 489, 491, 493, 495, 497, 499,
501, 503, 505, 507, 509, 511, 513, 515, 517, 519,
521, 523, 525, 527, 529, 531, 533, 535, 537, 539,
541, 543, 545, 547, 549, 551, 553, 555, 557, 559,
561, 563, 565, 567, 569, 571, 573, 575, 577, 579,
581, 583, 585, 587, 589, 591, 593, 595, 597, 599,
601, 603, 605, 607, 609, 611, 613, 615, 617, 619,
621, 623, 625, 627, 629, 631, 633, 635, 637, 639,
641, 643, 645, 647, 649, 651, 653, 655, 657, 659,
661, 663, 665, 667, 669, 671, 673, 675, 677, 679,
681, 683, 685, 687, 689, 691, 693, 695, 697, 699,
701, 703, 705, 707, 709, 711, 713, 715, 717, 719,
721, 723, 725, 727, 729, 731, 733, 735, 737, 739,
741, 743, 745, 747, 749, 751, 753, 755, 757, 759,
761, 763, 765, 767, 769, 771, 773, 775, 777, 779,
781, 783, 785, 787, 789, 791, 793, 795, 797, 799,
801, 803, 805, 807, 809, 811, 813, 815, 817, 819,
821, 823, 825, 827, 829, 831, 833, 835, 837, 839,
841, 843, 845, 847, 849, 851, 853, 855, 857, 859,
861, 863, 865, 867, 869, 871, 873, 875, 877, 879,
881, 883, 885, 887, 889, 891, 893, 895, 897, 899,
901, 903, 905, 907, 909, 911, 913, 915, 917, 919,
921, 923, 925, 927, 929, 931, 933, 935, 937, 939,
941, 943, 945, 947, 949, 951, 953, 955, 957, 959,
961, 963, 965, 967, 969, 971, 973, 975, 977, 979,
981, 983, 985, 987, 989, 991, 993, 995, 997, 999,
```

Picture 1: Expected output of Task 4

Task 5 (Basics)

1. Create a new class called “Task5” in your project “Exercise2”.
2. Implement an application that stores a 10x10 matrix of integer values in an array. First, create an array of the needed size. Next, fill the matrix with random numbers between zero and 100 (see hint for information how to do that). Use loops to do the filling. Afterwards print the generated matrix to the console (see Picture 2).

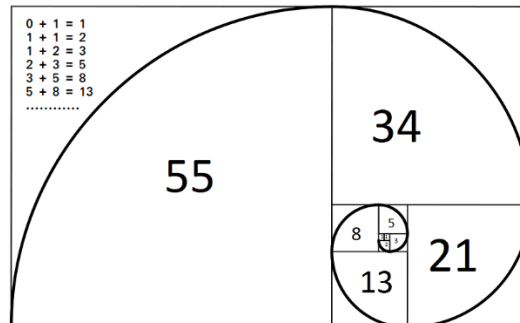
Hints: You can use *Math.random()*; to generate a random floating number between 0.0 and 1.0. By multiplying this value, you can generate random values within an interval. Further, remember to cast your final calculation into an integer value.



```
<terminated> Task5 [Java Application] C:\Program Files (
22 40 24 92 88 52 41 63 97 14
67 59 24 33 89 13 30 82 23 15
26 1 0 5 86 34 51 11 88 48
12 96 8 40 46 34 34 98 89 17
68 59 19 13 36 82 97 70 97 3
5 18 65 95 73 60 31 77 18 87
63 93 43 46 18 13 15 69 70 47
2 68 99 98 7 64 45 76 93 85
39 45 30 93 61 99 25 26 85 56
97 27 74 8 29 56 77 93 47 66
```

Picture 2: Expected output of Task 5

Task 6 (Recursion)



Picture 3: Visualization of the Fibonacci sequence

Fibonacci numbers are an integer sequence, where the n th is defined as $F_n = F_{n-1} + F_{n-2}$ with the starting values $F_0 = 0$ and $F_1 = 1$. To calculate the n th Fibonacci number we will apply the concept of recursion, which describes the ability of a function to call itself.

1. Create a new class called “Task6” in your project “Exercise2”.
2. Implement a public and static function `fibonacci(int n)` which returns the n th Fibonacci number. For all $n \geq 2$ use the function itself to calculate F_{n-1} and F_{n-2} .
3. Test your function by calling your function from the `main()`-method.
4. Use a for-loop in your `main()`-method to print the first 10 Fibonacci numbers (see Picture 4).
5. Extend your implementation by an `int[]` array `fib_numbers` in which the first 10 Fibonacci numbers are stored.
6. Use the debugger to inspect your code. At each step try to predict what will happen next in your program.

```
Problems Javadoc Declaration Console
<terminated> Task6 [Java Application] C:\Program Files (x86)\Java\jre1.8.0_181\bin\javaw.exe
0 0
1 1
2 1
3 2
4 3
5 5
6 8
7 13
8 21
9 34
```

Picture 4: Visualization of the Fibonacci sequence