Use CodeJudge to check your programs.
There are no blanks (mellemrum) symbols before or after the outputs.

Exercise 1: Solve exercise 3.1 on CodeJudge.

End of Exercise 1

Exercise 2: Solve exercise 3.2 on CodeJudge.

End of Exercise 2

Exercise 3: Tilføj en metode ved navn `isEven`, og få følgende program til at virke.

```
import java.util.*;
public class FillInTheBlankI {
    public static void main(String[] arg) {
        Scanner console = new Scanner(System.in);
        System.out.println("Indtast et heltal:");
        int x = console.nextInt();
        if (isEven(x)) {
            System.out.println(x+" er et lige tal.");
        }else {
            System.out.println(x+" er et ulige tal.");
        }
    }
    //Her skal der komme en metode. Men hvilken?
}
```

End of Exercise 3

Exercise 4: Lav en metode ved navn `printNumbers` som tager et naturligt tal, maximum, som argument og skriver tallene fra 1 til og med maximum i firkantede paranteser. For eksempel skal metoden håndtere disse kald:

```
printNumbers(15);
printNumbers(5);
```

og producere følgende output:

```
[1][2][3][4][5][6][7][8][9][10][11][12][13][14][15]
[1][2][3][4][5]
```

Du må gerne antage, at input er ≥ 1.
Lav i programmets `main`-metode en scanner, og brug den til at læse et heltal fra `System.in` og kald dernæst `printNumbers` med dette heltal som parameter.

End of Exercise 4

Exercise 5: Write a class `Factorial` which contains a method

```
public static int factorial(int n)
```

This method computes factorial function $n!$. The defintion was given in the lecture (or can be found on the web).
Remark: When you use CodeJudge, class and method names have to be precisely as shown above and you have to remove the `package` declaration before uploading.

End of Exercise 5

Exercise 6: Dette program skal tage 3 ord og printe dem på hver sin linie med foranstillede prikker, således at alle linier fylder 30 tegn.
F.eks. input:

```
kort
laaaaaaaaaaaang
mellem
```

output:

```
.........................kort
..............laaaaaaaaaaaang
........................mellem
```

Programmet har dog to grelle fejl:

- de indtastede ord gemmes ikke i strengarrayet ved navn treOrd,
- metoden repeatChar findes ikke.

Hvilke typer har metoden `repeatChar` som input og output?
Ret begge fejl og få programmet til at virke. (Man må gerne antage, at inputordene aldrig er over 30 tegn lange.)

```
import java.util.*;
public class FillInTheBlankII {
    private static int LineLength = 30;
    public static void main(String[] arg) {
        String[] treOrd = new String[3];
        Scanner console = new Scanner(System.in);
        System.out.println("Indtast tre ord:");
        String ordEt  = console.next(); //gemmes i arrayet treOrd.
        String ordTo  = console.next(); //samme.
        String ordTre = console.next(); //samme.
        for(int i = 0; i<treOrd.length; i++) {
            System.out.println(
                repeatChar('.',LineLength - treOrd[i].length())+
                treOrd[i]);
        }
    }
    //Her kommer metoden repeatChar.
}
```

End of Exercise 6

Exercise 7: Refers to last week's exercises. Improve the the program `WriteNumbers_Bad` by using parametrized methods.

End of Exercise 7

Exercise 8: Write Java code that prompts the user to enter his or her full name, then prints the name in reverse order (i.e., last name, first name). Here is an example dialogue with the user:

```
Please enter your full name: Sammy Jankis
Your name in reverse order is Jankis, Sammy
```

End of Exercise 8

Exercise 9: Write a class `Cosine` which contains a method

```
public static double cosine(double x,int k)
```

This method computes the cosine-function $\cos(x)$ using the Taylor-expansion of order $2k$ at 0.0. The Taylor-expansion of order $2k$ at 0.0 is given by

$$\sum_{i=0}^{k} \frac{(-1)^i x^{2i}}{(2i)!}$$

Compare the results with those of the library function `Math.cos` on some arguments $x \in [-\pi, \pi]$ and $k \in \{0, 1, \ldots, 10\}$.
Taylor-expansions is how a computer computes trigonometric functions.
Remark: When you use CodeJudge, class and method names have to be precisely as shown above and you have to remove the `package` declaration before uploading.

End of Exercise 9

Exercise 10:
Write a method called `swapPoints` that takes two `Points` as arguments and swaps their values with each other. Consider the following example code that calls `swapPoints`:

```
Point p1 = new Point(5, 2);
Point p2 = new Point(-3, 6);
swapPoints(p1, p2);
System.out.println("(" + p1.x + ", " + p1.y + ")");
System.out.println("(" + p2.x + ", " + p2.y + ")");
```

The output produced from the above code should be:

```
(-3, 6)
(5, 2)
```

End of Exercise 10

Exercise 11: (∗) Løs nedenstående opgave. Det er *Programming Project 2* fra bogens kapitel 3.

A certain bank offers 6.5% interest on savings accounts, compounded annually. Create a table that shows how much money a person will accumulate over a period of 25 years, assuming an initial investment of $1000 and assuming that $100 is deposited each year after the first. Your table should indicate for each year the current balance, the interest, the new deposit, and the new balance.
Use tabulators (`\t`) to structure the table.

End of Exercise 11

Exercise 12: [*]
Write a class `BinCoeff` for computing the factorial and binomial coefficients. Use `int` as data type. For the factorial you might consult the the lecture or reuse a previous exercise.
Recall that the factorial $n!$ of a non-negative integer n is defined as

$$\begin{aligned} n! &= \prod_{i=1}^{n} i = 1 \cdot 2 \cdot 3 \cdots (n-1) \cdot n \quad \text{,for } n \geq 1 \\ 0! &= 1 \end{aligned}$$

and that the binomial coefficient $\binom{n}{k}$ for non-negative integers n, k, $n \geq k$ is defined as

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} \tag{1}$$

a) Implement the method

```
public static int binomial(int n, int k);
```

Test you program against the example given on CodeJudge.

b) Tricky, you might want to skip it) Enable your program to compute $\binom{20}{10} = 184756$ while still using `int` as data type and avoiding overflow. Call the resulting method `binomialSmart` and put it together with `binomial` into the class.

Hint: A mathematical definition rarely leads to a good and efficient implementation, so tweak it.

Remark: The factorial $n!$ is the number of ways in which n distinct objects can be arranged in a row. The binomial coefficient $\binom{n}{k}$ is the number of ways to select k objects from a set of n objects.

End of Exercise 12

Exercise 13: [*] Write a class `Gates` which realizes logic gates. The class should have methods for `AND`, `OR`, `NAND`, `EXOR`, and `NOT`. These methods have two formal boolean parameters (`NOT` has only one) and return boolean.
Test your implementation.

End of Exercise 13