Handout 2
C++ Programming
Deadline is
October 5

1.
1) Write a function that computes the value of the binomial coefficient $\mathrm{C}(\mathrm{n},\mathrm{r})\ or\ \binom{n}{r}$

$$\mathrm{C}(\mathrm{n},\mathrm{r}) = \frac{n!}{(n-r)!\,r!}$$

$$\mathrm{n}! = \mathrm{n} * (\mathrm{n}-1) * (\mathrm{n}-2) * \cdots * 1$$

2) Embed your function into a little program that reads two integers **`n`** and **`r`** from **`std::cin`** and writes the value of the binomial coefficient to **`std::cout`**

2.
Write a function **`permutNumbers`** that prints all n! many permutations of the numbers 1 to n on std::out.

Example: the output for **`permutNumbers (3)`** shall be:
123, 132, 213, 231, 312, 321

3. Given the following function definition:

```
int sum_down(int x)
{
    if (x >= 0)
    {
        x = x - 1;
        int y = x + sum_down(x);
        return y + sum_down(x);
    }
    else
    {
        return 1;
    }
}
```

a) What is this smallest integer value of the parameter **x**, so that the returned value is greater than 1.000.000 ?

THE ANSWER IS 19. I found it by just computing the first values.

b) Rewrite the function, so that it is free of recursion. I.e. give an iterative definition on the foundation of a loop-construct.
TIP: First transform the recursive definition, so that you have just a single recursive call.

I didn't succeed to do this one.

c) Is it OK to switch the type of the parameter **x** to **double**?
Discuss your decision / give an argumentation.

There is no problem I guess to change an integer to a double because a double is more precise than a integer.
The only point we can talk about is the memory it will take because an integer is 4 bytes and a double is 8 bytes : It's double the memory.
I don't think it will impact the complexity of the algorithm.
The problem appears just if you do it in the other way because we would have to approximate the value.

d) Is it OK to switch the type of the parameter **x** to **unsigned int**?
Discuss your decision / give an argumentation.

A normal integer can take values from -2147483648 to 2147483647 and an unsigned integer take values from 0 to 4294967295. In our case we are using only positive integers that can fill in the borders of both data types so we can use an unsigned integer in this case.
Considering the memory there is no change because both are 4 bytes.

e) Is it OK to switch the function head to **int sum_down(const int x)**?
Discuss your decision / give an argumentation.

It is not okay because in the function we are changing the value of the integer x many times. We can’t set it up on a constant because a constant can only have one value and we can’t assign it to something else like x = x-1.