Programming 1 – Tutorial Exercises 2

1. Write a function *power(x, y)* that returns a value of expression *x* (of type double) to the power *y* (of type int).
2. Write a function *power(x, y)* that returns a value of expression *x* (of type double) to the power *y* (of type int) using recursion.
3. Write a function *isPrime(n)* checking whether the integer number taken as an argument is prime. Write also a function *printPrimeNumbers(a, b)* print to the screen all prime between *a* and b.
4. Write a function *calculate(a, b, s)* calculating operation "*a s b*", where *"s"* is +, -, * or /.
5. Write a function *isArmstrong(n)* checking whether the integer number taken as an argument is Armstrong[1] number. Write also separate function *read_data()* for input data handling.
6. Write a function *mean(data,n)* with two parameters: an array of double values *data* and an integer *n*, which is the length of the array. Function should calculate and return a mean of values in the array.
7. Write a function *standard_deviation(data,n)* with two parameters: an array of double values *data* and an integer *n*, which is the length of the array. Function should calculate and return a standard deviation of values in the array.

$$s=\sqrt{\frac{\sum\left(x_i-\overline{x}\right)^2}{n}}$$

[1] Armstrong number is a number which is equal to sum of digits raise to the power total number of digits in the number

Programming 1 – Tutorial Exercises 2

1. Write a function *power(x, y)* that returns a value of expression *x* (of type double) to the power *y* (of type int).
2. Write a function *power(x, y)* that returns a value of expression *x* (of type double) to the power *y* (of type int) using recursion.
3. Write a function *isPrime(n)* checking whether the integer number taken as an argument is prime. Write also a function *printPrimeNumbers(a, b)* print to the screen all prime between *a* and b.
4. Write a function *calculate(a, b, s)* calculating operation "*a s b*", where *"s"* is +, -, * or /.
5. Write a function *isArmstrong(n)* checking whether the integer number taken as an argument is Armstrong[2] number. Write also separate function *read_data()* for input data handling.
6. Write a function *mean(data,n)* with two parameters: an array of double values *data* and an integer *n*, which is the length of the array. Function should calculate and return a mean of values in the array.
7. Write a function *standard_deviation(data,n)* with two parameters: an array of double values *data* and an integer *n*, which is the length of the array. Function should calculate and return a standard deviation of values in the array.

$$s=\sqrt{\frac{\sum\left(x_i-\overline{x}\right)^2}{n}}$$

[2] Armstrong number is a number which is equal to sum of digits raise to the power total number of digits in the number