0. Fundamental Algorithms – Introductory Session

First off, make sure you have read the guide to the laboratory sessions (available at http://users.utcluj.ro/~cameliav/fa/LabGuideline.pdf). In this introductory session, you will get used to working with *Visual Studio 2008* by writing a more complex `Hello World` C/C++ application. Also, you will see how to generate the data to evaluate your algorithms and how to generate the required charts (either with *MS Excel* or by using a framework written in C++).

Introduction to *Visual C++*

To get the free version of the development environment, go to (only if you don't have a C/C++ environment installed already at home):

http://www.microsoft.com/express/Downloads/#2008-Visual-CPP

To create a new C/C++ project, using the wizard:

- *File – New – Project... – Win32 –* ***Win32 Console Application*** *– Name: HelloWorld – Location:* ….choose… *– OK – Next –* ***Empty project*** *– Finish*;
- *Solution Explorer – Source Files – Add – New Item – C++ File (.cpp) – Name: HelloWorld – Add*;
- Include `stdio.h` and `conio.h`, write your main function in which you print „Hello, world!", on the screen (use `getch()`) to keep the console from closing until you hit a key.
- Compile and run your application

Working with files

Now, to extend your application, do the following (use Google or MSDN library for help, or ask the teaching assistant):

- Declare an array of integers of size `MAX_SIZE` – constant defined by you
- Read a sequence of *n* numbers from the keyboard, and store them in the array
- Print the *n* numbers in the array
- Create and open a file, write the numbers from the array in the file, and close the file (check the file to see it worked)
- Now open the previous file, read the contents and print them on the screen (don't forget to close the file at the end)

Generating test cases for the algorithms (best, worst, average)

In order to test your algorithms, you have to generate a series of input sequences, such as: a sorted array of integers (of given size), a random array of integers (of given size), etc.

Since generating an ordered sequence is straightforward, let us focus on generating a random sequence of integers. We suggest two alternatives:

1. Using the random number generator available in C/C++
2. Using the Profiler Framework (available at http://users.utcluj.ro/~cameliav/fa/profiler.zip)

1. How to use the random number generator available in C/C++:
    - Read about `rand(,)` `srand()` functions and `RAND_MAX` constant:
        - http://www.cplusplus.com/reference/cstdlib/rand/
        - http://www.cplusplus.com/reference/cstdlib/srand/
        - http://www.cplusplus.com/reference/cstdlib/RAND_MAX/
    - Write a sequence of code/function which:
        - Generates *n* random numbers, using the `rand()` function alone, stores them in an array, then prints them on the screen; what happens when you run your program the second time?
        - Change your sequence of code such that the sequence of *n* random numbers differs between runs

2. How to use the Profiler Framework: check http://users.utcluj.ro/~cameliav/fa/profiler_guide.pdf

Exercises:

1. Write a function which generates an array of *n* random integers between *Low* and *High*, and returns the array; print the contents of the array in a file
2. Write a function which generates a sorted array of <u>random</u> integers; print the contents of the array in a file

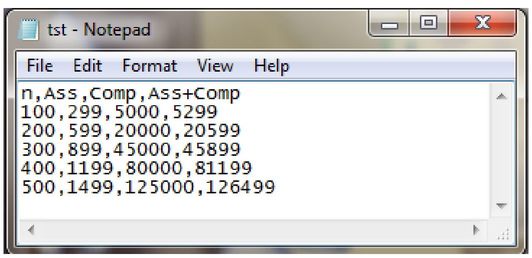
Generating charts for the analysis of algorithms

Again, you have two options for generating the evaluation charts:

1. Use *MS Excel*
2. Using the Profiler Framework, same as before: http://users.utcluj.ro/~cameliav/fa/profiler.zip

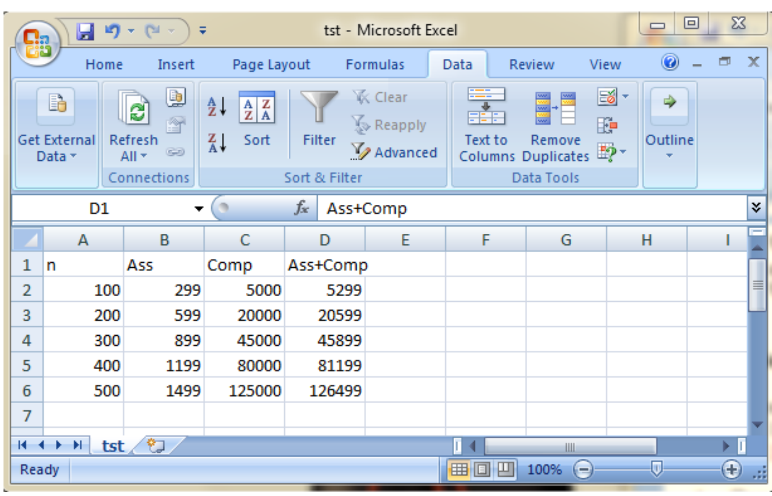
1. How to use *MS Excel*:
   - First, from your program, you have to save your analysis data in a *.csv* (**c**omma-**s**eparated **v**alues) file. You are free to use your own format for the file. However, it is a good idea to use the following format:

     Size_of_problem, No_assignments, No_comparisons, No_assignments+No_comparisons



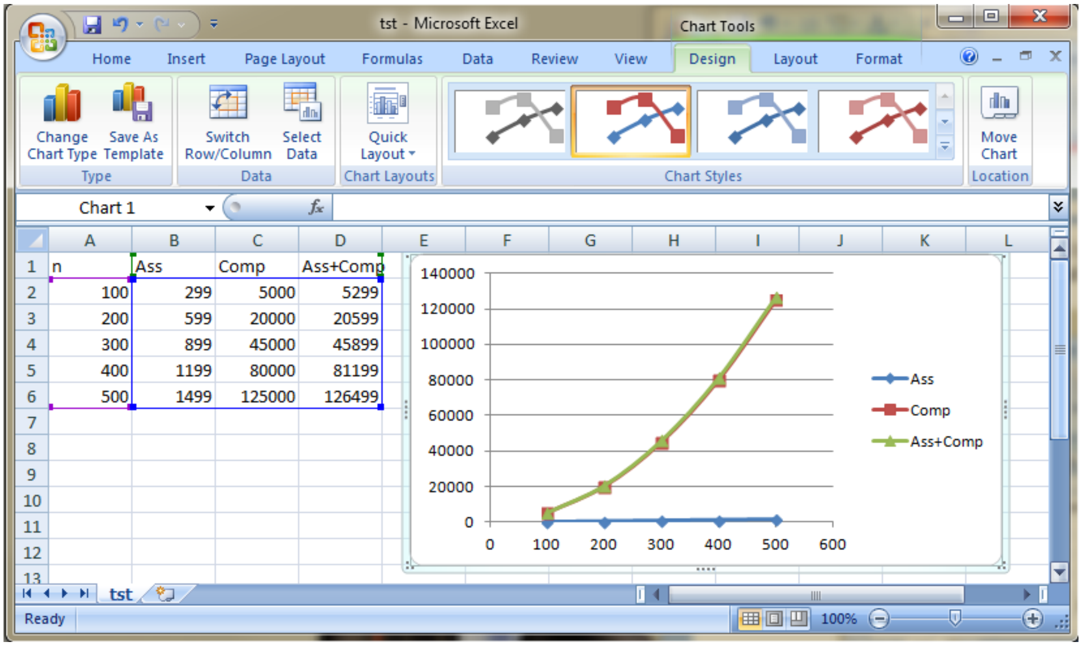
The figure above represents an example of how a *.csv* file might look like for one analysis case – input size 100 to 500. You can choose to use the same file for all cases of an algorithm (best, average, and worst). How many columns will your *.csv* require then?

   - Importing data to *MS Excel* (version 2010): if your data is properly formatted and the extension is .csv, *Excel* should recognize it and open it correctly:



However, if *Excel* places your values in the same column (probably you used a different column separator than the one set in *Excel*), use the *Data->Text to Columns* wizard to correct this (ask the teaching assistant for help). Also, you may import your data in Excel by using the *Data->Get External Data* wizard (again, ask the teaching assistant)

- Building the chart: select the data rows and columns; then go to *Insert->Charts->Scatter* and select the second type (connected points). For the above data, what you get should look like:



Note that the number of assignments, although linear, looks constant when placed on the same chart with the number of comparisons or with the sum (both quadratic). As a rule, whenever one curve cannot be visualized correctly because of the difference in growth rate with the other curves, it is best to place it also on a separate chart, by itself (try to do this by yourself).

- Additionally, you can name your chart, label the axes, scale the axes – you may need to perform scaling when comparing algorithms – on small inputs, for example. Try to identify how these operations are performed in *Excel* (ask the teaching assistant for help whenever you need guidance).

- ! Don't forget you also have to interpret the charts, and place your comments in comments at the beginning of your source code file

2. How to use the Profiler Framework: http://users.utcluj.ro/~cameliav/fa/profiler_guide.pdf

Exercise: Write a C/C++ program which writes in a file, for *n* starting from 100 to 10.000 (with a 100 increment), the following values (for each value of *n* use a separate line):

$n,\ 100*\log(n),\ 10*n,\ 0.5*n^2\ ,\ 2^n\ ,\ 2*n!$

Use the values in the file to build scatter plots for these functions, either by using *MS Excel* or the Profiler Framework.