Algorithms Design
Homework Assignment

Costin Badică, Alexandra Albişi, Mihaela Colhon,
Cristinel Ungureanu, Ionuţ Murareţu

Abstract

This document introduces the goals and methodology for developing the Algorithms Design assignment. The document contains also a description of the lab homework deliverables. The document targets 1^{st} year students in their study of algorithms.

1 Introduction

The goal of your assignment is to develop a software for experimenting with algorithms. The lab homework is focused on the development of skills for good programming of basic algorithms, covering coding, design, documentation and presentation of results. At the end of your lab homework you must produce a set of deliverables including:

- Technical Report, .doc/.docx or typeset using LaTeX [1] (Overleaf tutorial).
- Source Code in C language (mandatory) and Python language (OPTIONAL for bonus points)
- Experimental non-trivial[1].

2 General aspects

2.1 Homework assignment problem

Suppose you are an investment agent and you have capital of value C. There are n offers available for investment to choose from. For each offer i there are a_i available shares of value v_i, which can bring you an estimated profit p_i for each share acquired. You are required to determine the investment that can bring you the maximum estimated return. Two different algorithms will be implemented.

[1] large and very large data sets randomly generated

2.2 Grading

The grading of your laboratory assignment will take into account all the elements presented in this document:

- The structure and content of the technical report should follow the requirements stated in section 4.
- The structure and content of the source code should follow the requirements stated in section 5.
- The content of the experimental data and results should comply with the requirements stated in section 6.

Additional points will be added for a correct Python implementation of your assignment.

3 Deliverables

For this assignment you have to produce three types of deliverables:

i) Technical report

ii) Source code and source code documentation

iii) Experimental data and results

3.1 Technical report

The technical report should describe briefly, concisely and clearly your work and achievements for the lab homework. The description must contain your assignment tasks (problems), how did you develop the software and the outcome of your work. More details are given in section 4.

3.2 Source code

The source code should be created in a modular way, such that multiple files are developed and used together. The source code should be accompanied by command lines and instructions for building the executable.

3.3 Experimental data and results

You should check your software on non-trivial input data. You should provide at least 10 non-trivial input data sets, and the corresponding outputs produced by your software. The source code for generating non-trivial input data will be provided as it is described in subsection 3.2. More details are given in section 6.

3.4 Homework Delivery

To allow for an easier check of every delivered project/homework the following steps must be followed as well:

- The homework must be delivered in an archive of type and with extension .zip whose name will be as follows:
  - English section: AD-CEN1.1A-LastName-FirstName-Homework-Assignment.zip
  - Romanian section: PA-CR1.1A-Nume-Prenume-Homework-Assignment.zip
- The name of the above archive is made from the following elements:
  - Abreviation of the course name:
    - AD: Algorithm Design
    - PA: Proiectarea Algoritmilor
  - Section, year and group:
    - CEN: Computer English
    - CR: Calculatoare Romana
  - Last Name and First Name:
    - Your Last and First Name (And middle name as well where available) as it appears in your Faculty registration papers
  - The full and complete name of the homework must be used as assigned to you in Google Classroom).

  Archive examples:
  - AD-CEN1.1A-Popescu-Ion-Homework-Assignment.zip
  - PA-CR1.1A-Popescu-Ion-Homework-Assignment.zip
- The zip archive must be uploaded in Google Classroom (without any external links or other storage where the homework might be available). Every homework is stored in Google Drive (when using Google Classroom):
  - After the homework is uploaded it is necessary to press the "Submit" button (otherwise the homework doesn't appear as delivered)
  - Not delivering the homework until the deadline leads to an automatic grade of 0
  - You can upload the homework and change it as many times as needed until the deadline (but only for urgent changes)
- To implement your homework:
  - Read the requirements multiple times
  - Deliver all requirements (the technical report, all observations, etc.)
  - Don't postpone the delivery until the final moment

- Note:
  - Your archive names must be exact as mentioned above. Any other type/format of upload will not be taken into consideration
  - Only one archive must be uploaded in Google Classroom per assignment. Multiple files will not be taken into consideration
  - Not delivering the assignment will lead to an automatic generated grade of 0
  - The homework must be individually made. Do not copy from other colleagues, sources from the Internet, etc. All deliveries must be personally developed, implemented, created and tested.

4 Technical report

The technical report must be typeset using LaTeX or .doc/.docx and provided in electronic form (PDF and sources in case of LaTeX). For documentation about LaTeX you can consult reference [3].

The technical report must be divided into a number of sections, including:

i) Cover page

ii) Problem statement

iii) Algorithms

iv) Experimental data

v) Results & Conclusions

4.1 Cover page

This is a one-page section containing the title of the lab homework, the name of the student/s, the group, year and section where the student is enrolled.

4.2 Problems statement

This section is the introduction of your technical report. It should describe clearly the task of your lab homework.

4.3 Algorithms

This section should contain the pseudo-code description of the algorithms employed throughout your assignment. The pseudo-code should use the format introduced in [2]. An example is shown in figure 1.

Moreover, this section must discuss each algorithms' memory and computational complexity, in the best and worst case scenarios. Any written explanations, exemplifications or figures regarding the algorithms should appear here.

INSERTION-SORT(A)
1. **for** $j = 2$,$length[A]$ **do**
2. $key = A[j]$
3. $\triangleright$ Insert $A[j]$ into the sorted sequence $A[1 \dots j-1]$
4. $i = j - 1$
5. **while** $i > 0$ and $A[i] > key$ **do**
6. $A[i+1] = A[i]$
7. $i = i - 1$
8. $A[i+1] = key$

Figure 1: Example for typesetting an algorithm.

4.4 Experimental data

Your application must also address the method for automated generation of non-trivial input test data. Non-trivial data can be obtained by randomly generating large and very large data-sets, e.g., vectors of length $10^3 - 10^8$.

4.5 Results & Conclusions

Figures comparing the average times (cpu time) on multiple runs on each data-set should be present here. The figures can be created with Microsoft Excel, Matlab, R, Python, etc. The results should be commented and explained.

This section must contain your own conclusions after performing the lab homework. Some suggestions about what to include in the list of conclusions are: a summary of your achievements, which were the most challenging and interesting parts and why, future directions for extending the lab homework in short and long term, a.o.

5 Source code

The C source code must contain the complete source of your application, including *.c* files, header files (*.h*), and Code::Blocks project.

The Python source code must contain all the complete source of your application, *.py* files and any other libraries used in your lab homework.

The following requirements must be met by the C source code. They will directly influence the grading of your lab homework.

- The lab homework code must successfully compile and build; when using GCC the build process should use the *make* tool and a Makefile. It should also be easy to open/import and build/run the delivered Code::Blocks project.
- The C code must follow the C99 standard (e.g. no compiler-specific extensions).

- The code must be portable; it must compile under two different compilers (e.g. Visual C++ and GCC)
- The code must follow the C coding style, imposed for this course.
- The code should be commented (every function and important variables and code blocks inside functions).
- The code must be modular (minimum 2 *.c* files and minimum one *.h* file)
- You must use an automated method for generating non-trivial input test data

The following requirements must be met by the Python source code. They will directly influence the grading of your lab homework.

- The lab homework code must be successfully interpreted and executed.
- The code must use the PEP 0008 indent style and address at least the following issues:
  - the variable and functions names should be descriptive, but not verbose
  - the use of global variable should be kept to a minimum
- The code should be commented (every function and important variables and code blocks inside functions).
- The code must be modular (minimum 2 *.py* files)
- You must use an automated method for generating non-trivial input test data.

6 Experiments and results

In this section you must explain the method that you used for testing your application, as well as the experimental results.

The experimental results will contain:

- A description of the output data that you obtained by running your algorithm, as well as the method that you used to test that this output is correct according to the algorithm specification.
- Optionally, the execution time of your algorithm, for each input data set.

You must also present the results (and optionally the recorded execution time) of your experiments in a meaningful way for the reader. The presentation method is left for your choice.

References

[1] Leslie Lamport, *LaTeX: A Document Preparation System*. Addison Wesley, Massachusetts, 2nd Edition, 1994.

[2] Thomas H. Cormen and Charles E. Leiserson and Ronald L. Rivest and Clifford Stein, *Introduction to Algorithms*. MIT Press, 3rd Edition, 2009.

[3] LaTeXproject site, http://latex-project.org/, accessed in April 2013.