ENGINEERING PROGRAMMING

MS in Earth Science Engineering

Semester 1, 2018/19

COURSE COMMUNICATION FOLDER

University of Miskolc
Faculty of Earth Science and Engineering
Institute of Geophysics and Geoinformatics

Course datasheet

<table>
<tr><td>Course Title: Engineering programming
Instructors: Péter Tamás Vass Dr., associate professor</td><td>Code:
Responsible department/institute: Institute of Geophysics and Geoinformatics / Department of Geophysics
Type of course: Optional</td></tr>
<tr><td>Position in curriculum (which semester): 3</td><td>Pre-requisites (if any) -</td></tr>
<tr><td>No. of contact hours per week (lecture + seminar): 0+2</td><td>Type of Assessment (examination/ practical mark / other): practical mark</td></tr>
<tr><td>Credits: 2</td><td>Course: full time</td></tr>
<tr><td colspan="2">Course Description:
The main objective of the subject is to familiarize the students with the elements of programming necessary to solve engineering and scientific problems. The course tries to transmit the knowledge by means of which the algorithms of simpler problems can be constructed and the algorithms can be implemented in the form of computer programs.
The short curriculum of the subject:
Introduction. Some basic concepts and definitions. The elements, design and descriptions of algorithms. Main features of programming languages. Fundamental steps of program development. Making the source codes of simpler problems in C language. Compiling the source codes and executing as well as testing the programs. Short history and main features of C programming language. Lexical elements of C programming language. Structure of C programs. Definition of variables in C programs. Use of constants in C programs. Control structures. Pointers and their application. Dynamic memory allocation in C programs. Arrays and their application. Standard library functions. User-defined functions. Text file input and output. Introduction to the use of higher level programming languages (MATLAB and GNU Octave).
Competencies to evolve:
Knowledge: T4, T5, T6, T7, T10, T12
Ability: K1, K2, K3, K4, K5, K13
Attitude: A1
Autonomy and responsibility:</td></tr>
<tr><td colspan="2">Assessment and grading:
Conditions for obtaining the signature: the presence in at least 70 % of the lessons and passing two classroom tests.
The determination of the practical mark is based on the evaluation of problem solving in the lessons and the results of two classroom tests. The weights of the partial achievements: problem solving in the lessons = 20%, results of the two tests = 40% + 40%.
Grading scale (% value → grade): 0 – 49 % → 1 (fail), 50 – 64 % → 2 (pass),
65 – 79 % → 3 (satisfactory), 80 – 89 % → 4 (good), 90 – 100 % → 5 (excellent).</td></tr>
<tr><td colspan="2">Compulsory or recommended literature resources:
The presented slides converted in pdf format: http://geofizika.uni-miskolc.hu/segedlet.html
Brian W. Kernigham – Dennis M. Ritchie: C Programming Language, 2nd Edition, Prentice-Hall Inc., ISBN-10: 0131103628
Clovis L. Tondo, Scott E. Gimpel, 1989: The C Answer Book, Second Edition, Prentice-Hall International, Inc., ISBN: 7-302-02728-5
John W. Eaton, David Bateman, Søren Hauberg, Rik Wehbring: GNU Octave, A high-level interactive language for numerical computations, Edition 4 for Octave version 4.0.3
July 2016
MATLAB numerical computing, Tutorials Point (I) Pvt. Ltd., 2014: www.tutorialspoint.com</td></tr>
</table>

Syllabus of the semester

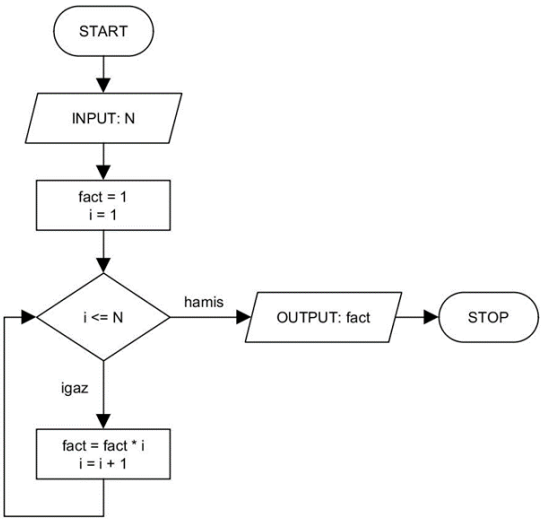
Week	Seminar
1	Introduction. Some basic concepts and definitions: computer program, software, algorithms, constants, variables, attributes of the variables. Elements of the algorithms: read, write, assignment, conditional branch, loop. The ways of expressing algorithms: natural languages, pseudocodes, flowchart.
2	Designing and expressing algorithms. Introduction to the use of a free flowchart editor. Doing exercises.
3	General characteristics of high-level programming languages. Semantics, syntax, compiler and interpreter. The fundamental steps of program development.
4	A short history of C programming language and its general characteristics. Introduction to the use of a free C and C++ program development environment. Writing the source codes of simpler algorithms in C language. Compiling and testing the implemented C programs.
5	Lexical elements of C programming language. identifiers, keywords, constants, comments, operators (precedence and associativity), punctuators. Structure of C programs. Data types in C programming language. Defining the variables in C programs. The use of constants.
6	Control structures: conditional branches (if, if – else, if – else if – else, switch –case), loops (the pre-test loop 'while', the post-test loop 'do –while', the for-loop. Special control statements: break, continue, return.
7	The first classroom test on the topics of algorithms and writing simpler C programs.
8	Pointers and their application. Dynamic memory allocation in C programs by means of a pointer.
9	Arrays and their application. Strings and character arrays. Multi-dimensional arrays.
10	Standard library functions. User-defined functions. Definition, declaration and function call.
11	Text file input and output in C language. Defining a file pointer. Opening a text file. Writing, reading and closing a text file..
12	Introduction to the application of higher level programming languages. The programming environments of Matlab and GNU Octave. Data types and formats. The execution of some important commands. Creating and running scripts in Matlab and GNU Octave.
13	The second classroom test on the topics of C programming.

14	Built-in functions in Matlab and GNU Octave. Operations with vectors and matrices. Assignment expressions. Control statements. Defining and calling user-defined functions. Plotting graphs..

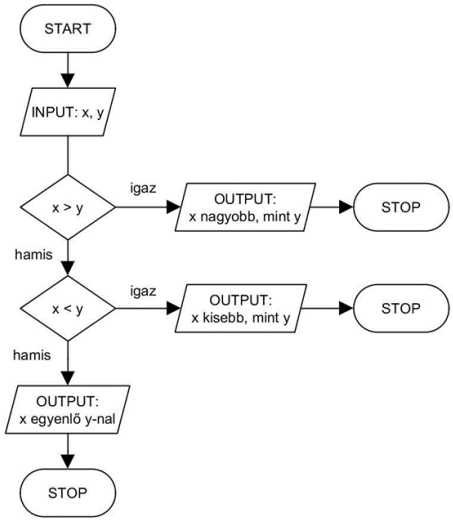
Sample for the classroom test with the answers

Engineering programming test
date

Task 1
Create a C program which asks the user to type a natural number, computes the factorial of the number, and displays the result on the monitor.
A computer can be used for implementing the program. Write down the source code of the program below the flowchart. (**max. points: 6**)
The flowchart of the algorithm which helps in solving the problem is presented here.



Task 2
Create a C program which asks the user to type two real numbers, compares them, and determines whether the first number is greater or less than the second one or the two numbers are equal. It also displays the result of the comparison on the monitor.
A computer can be used for implementing the program. Write down the source code of the program below the flowchart. (**max. points: 8**)
The flowchart of the algorithm which helps in solving the problem is presented here.



Sample answer for Task *1*

```
#include <stdio.h>

main(){

      int N, i, fact;

      printf("The program asks for a natural number, and computes its
factorial.\n");
      printf("Please type the number: ");
      scanf("%d",&N);
      fact=1;
      if (N==0)
            printf("The factorial of the number: %d", fact);
      else if (N<0)
            printf("The entered number is not a natural number.");
      else{
            for (i=1; i<=N; i++)
                  fact=fact*i;
            printf("The factorial of the number: %d", fact);
      }

}
```

Sample answer for Task 2

```
#include <stdio.h>

main(){

      float x, y;

      printf("The program asks for two real numbers, and compares
them.\n");
      printf("Please type the first number: x= ");
      scanf("%f",&x);
      printf("Please type the second number: y= ");
      scanf("%f",&y);
      if (x>y)
            printf("x is greater than y");
      else if (x<y)
            printf("x is less than y");
      else
            printf("x is equal to y");
}
```

Evaluation of the answer for Task 1

Including the necessary header files, the definition of main function and the necessary variables	2 points
Data input and variable incialization	2 points
Correct application of the control structures	2 points

Evaluation of the answer for Task 2

Including the necessary header files, the definition of main function and the necessary variables	2 points
Data input	2 points
Correct application of the control structures	3 points
Displaying the result	1 point

Grading scale

range	mark
< 6 ponts	fail (1)
≥ 6 ponts and < 9 ponts	pass (2)
≥ 9 ponts and < 11 ponts	satisfactory (3)
≥ 11 ponts and < 13 ponts	good (4)
≥ 13 ponts	excellent (5)