Programming languages 1

Practicing exercises

1. Write a C program, which reads in 2 numbers from keyboard (first is smaller than the second) and print out all the even numbers between them (excluding the given numbers).

   Example
   - *input: 3 10* → *output: 4, 6, 8,*
   - *input 4 17* → *output: 6, 8, 10, 12, 14, 16,*
   - *input 3 4* → *output: (nothing)*

2. Write a C program, which reads in 3 numbers from keyboard and determines which one is the greatest.

   Example
   - *input: 1, 4, 7* → *output: 7*
   - *input: -2, 100, 9* → *output:100*

3. Write a C program, which read in 9 integer number and store them in an array. Then it determines that the number of even numbers or the number of odd numbers is larger and prints out the result.

   Example
   - *input: 3, 5, 1, 4, 66, 45, 123, 5, 2* → *output: more odd*
   - *input: 324, 3, 5, 6, 88, 24, 6, 0, 7* → *output: more even*

4. Write a C program, which read in 10 integer number from keyboard and store them in an array. Then it prints out all the 10 numbers in increasing order.

   Example
   - *input: 41, 3, 9, 1, 3, 6, 45, 12, 5, 2* → *output: 1, 2, 3, 3, 5, 6, 9, 12, 41, 45*
   - *input: 1, 3, 5, 2, 4, 9, 8, 7, 6, 0* → *output: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9*

5. Write a C program, where you declare an integer array with 10 elements, and the program reads in an integer number from keyboard. The first element of the array will be this input number. Then the program fills the other element according to the following rule: the i^{th} element is greater by 3 than the $(i-1)^{th}$ number, so the second number is equal to the first plus 3, the third is equal to the second plus 3, and so on. After all element is determined the program prints out all the values to the screen.

   Example
   - *input: 4* → *output: 4, 7, 10, 13, 16, 19, 22, 25, 28, 31*
   - *input: 21* → *output: 21, 24, 27, 30, 33, 36, 39, 42, 45, 48*

6. Write a C program, which reads in a real number from keyboard and determines the sign of the number by calling a self-written implementation of the mathematical sign function. So you have to write an own function code, which behaves in the following way:
   - *sign(x)==1, if x is positive*
   - *sign(x)==0, is x is zero*
   - *sign(x)==-1, if x is negative*

   The main program unit reads a number, calls the sign function and prints out the value returned by the function.

   Example:
   - *input: 51.37* → *output: 1*
   - *input: 0.0* → *output: 0*
   - *input: -2* → *output: -1*

7. Write a procedure in which you define a procedure to print out a chessboard of 'x' and '+' characters in a square area with side length N, which is the parameter of the procedure. In the main program unit it read in an integer number (N) and calls the procedure with this value.

   Example:
   - *input: 3* → *output:*

     + x +
     x + x
     + x +

   - *input: 6* → *output:*

     + x + x + x
     x + x + x +
     + x + x + x
     x + x + x +
     + x + x + x

8. Write a C program, which reads in a word (string) from keyboard and print it back to the screen replacing all vowel to an underscore ('_') character.

   Example:
   - *input: "Applepie"* → *output: "_ppl_p__"*
   - *input: "transportation"* → *output: "tr_nsp_rt_t__n"*

9. Write a C program, where you declare an integer array with 40 elements, and the program fills it with the element of the following mathematical sequence. The first element of the array will be 0, the second element is 1 and then the i^{th} element of the array is equal to the sum of the $(i-1)^{th}$ and the $(i-2)^{th}$ elements. The values of the elements must calculated in a loop by the program (not manually entered constants). Finally the program prints out all the elements of the array.

   Example
   - *output: 0, 1, 1, 2, 3, 5, 8, 13, 21, …*

10. Write a C program, which opens the “text.txt” file and determines how many capital and how many non-capital letters of the English alphabet are in the file.

    Example
    - *file content: ThiS Is An ExaMpLe TeXt In ThE fiLe.*
    - *output: 13 capital and 15 non-capital*

11. Write a C program, in which points of a 2D plane are represented by a record data structure (struct) of two float numbers. Define a function, which can determine the distance of a point (given as a parameter) from the origin/center. Call this function in the main function by values given by the user.

    Example
    - *input: 5 ; 6*
    - *output: The distance of (5;6) point from (0;0) is 7.81*

12. Write a C subroutine which receives two random integer values from the caller using parameter passing by (pseudo)address. The first of the values must be in the [65 ; 91] closed interval, the other is either 0 or 1. This procedure has the following header: `void rndlttr(int *L, int *C)`. In the main program, call it using the x and y local variables and then print the value of x as a character. The procedure have to overwrite the value stored in x with the value of the `x+y*32` expression.

    Example
    - *x=65, y=1 → call procedure → x=97, y=1 → output: ‘a’*
    - *x=66, y=0 → call procedure → x=66, y=0 → output: ‘B’*

13. Write a C program in which you open a text file called temp.txt and read all the characters from the file to a char array without any information about the size of the file. So create an array for 10 characters by dynamic memory allocation, start to read the characters from the file and store into the array. If the number of read characters reach the size of the array reallocate the array with double size. Finally print out the whole arraycontent.

14. Write a C program in which you handle dates with records (struct) having 3 integer fields: year, month, day. Ask a positive integer from user (called N) and dynamically allocate an array for N date structure. Fill the array with N random rational dates from the 20th century and then save them into a file called “dates.csv” separated the fields with semi-colon (each date in a new line). The dates must be produced by a void function which has two parameters the starting address of the date array and the number of needed dates (N).

    A possible content of “dates.csv”:

    1979;4;18
    1996;2;29
    1903;3;1
    1945;4;30
    1932;12;9

15. Create a task for yourself or fined one on the Internet which uses the studied programming tools/techniques. Solve this problem with a self-written C program. Repeat this exercise 100 times with different tasks.

    Examples of C programming exercises and tutorials on the Internet

    - *https://www.programiz.com/c-programming/examples*
    - *https://www.w3resource.com/c-programming-exercises/*
    - *https://fresh2refresh.com/c-programming/c-programs/*
    - *https://www.tutorialspoint.com/learn_c_by_examples/simple_programs_in_c.htm*
    - *https://codeforwin.org/2015/05/basic-programming-practice-problems.html*
    - *http://www.learntosolveit.com/cprogramming/*
    - *http://www.worldbestlearningcenter.com/index_files/c_tutorial_lesson.htm*