EE6411 Lab Sheet #2
Developing Functions

Reiner Dojen[1]

[1]reiner.dojen@ul.ie

1 Objectives

- Declare, define and call functions.
- Pass information to and from functions.
- Develop programmer defined functions to implement reoccuring tasks.
- Use the library function `rand()` to generate pseudo-random numbers.
- Passing information from the operating system to the `main` function.

2 Preparation

You should be familiar on how to create/compile/execute applications in Visual Studio IDE. I also recommend to have worked through E-Activities 1-4 Automatic Code Format and 1-5 Managing multiple Projects in a Solution.

3 Exercises

Ex2.1 Fahrenheit-to-Celsius Function

Develop a program that converts fahrenheit to celsius and celsius to fahrenheit. Create the following functions (declare these at the beginning of your source file):

double fahrenheit2celsius(double fahrenheit) : Converts the passed fahrenheit value into the corresponding celsius value ($°C = \frac{5}{9} \cdot (°F - 32)$).

double celsius2fahrenheit(double celsius) : Converts the passed celsius value into the corresponding fahrenheit value ($°F = (\frac{9}{5} \cdot° C) - 32$)

double fahrenheit2celsiusRange(double fmin, double fmax, double step) : Converts the passed fahrenheit range (from min to max in intervals of step) into the corresponding celsius values. Uses function `fahrenheit2celsius(...)` for each individual conversion.

double celsius2fahrenheitRange(doublec min, double cmax, double step) : Converts the passed celsius range (from min to max in intervals of step) into the corresponding fahrenheit values. Uses function `celsius2fahrenheit(...)` for each individual conversion.

int main() Prompt the user to select from the following:

- **Single fahrenheit to celsius:** Reads a single value from user and converts it to celsius.
- **Range fahrenheit to celsius:** Reads min, max and step values from user and converts given range to celsius.
- **Single celsius fahrenheit:** Reads a single value from user and converts it to fahrenheit.
- **Range celsius to fahrenheit:** Reads min, max and step values from user and converts given range to fahrenheit.

Make sure that your code is properly formatted (K&R coding style) and sufficiently commented (top of file, beginning of each function, code within function). Compile and test your program.

Ex2.2 Random Numbers

Your task is to write a programm that prints 10 random numbers in the range 0 - 20 to the screen. Make sure to use proceeds as follows:

- Create a function `int getRandomNumber(int min, int max)` that returns a random number in range min to max (both limits inclusive). Use the library function `rand()` (see hint below) to generate the numbers.
- Use a function called `printRandomNumbers(...)` that calls `getRandomNumber(...)` to generate ten numbers and to print them to the screen.
- Call the method `printRandomNumbers(...)` from the `main()` method.
- Declare the functions `getRandomNumber(...)`, `printRandomNumbers(...)` at the beginning of your source file, then implement `main()` and finally define the functions.
- **Do not use global variables!!!**
- Make sure that your code is properly formatted (K&R coding style) and sufficiently commented (top of file, beginning of each function, code within function).
- Compile and test your program.

Hint: The function `int rand()`, declared in `<cstdlib>` returns a pseudo-random integer value between 0 and the constant RAND_MAX (this function is the C++ equivalent to the Java `Random` class). Hence the expression

```
rand()%Number
```

always yields an integer value between 0 and Number-1 (inclusive).

Please note, that `rand()` always produces the same sequence of numbers. To start (an apparently) different sequence of pseudo-random numbers, use the function `void srand(int seed)` (also declared in `<cstdlib>`) with a user chosen value (or something like the current time) for the seed.

Further information regarding random numbers in C++ can be found at
`http://www.cplusplus.com/reference/cstdlib/rand/`
`https://en.cppreference.com/w/cpp/numeric/random/rand`
`https://www.geeksforgeeks.org/rand-and-srand-in-ccpp/`

If you don't understand any issue involved in above expression ask either a fellow student, your teaching assistant or your lecturer.

Ex2.3 Passing Information to the `main()` Function

Write a program that accepts three numbers (use either integers or floating point numbers) from the command line and then prints them in ascending order.

First, develop pseudo-code to sort three numbers in ascending order (past this pseudo-code at the beginning of your file and also use it as comments in your program).

Then, use the discussed mechanism of passing parameters from the OS to the `main(int argc, const char *argv[])` function. Implement your pseudo-code solution to print the numbers in ascending order (don't forget to convert the passed parameters to `int` or `double`).

Make sure that the user is calling this program correctly (i.e. with the correct number of arguments). If not, print a message to the screen: "Incorrect number of arguments - please call with mainArg num1 num2 num3".

Ex2.4 Prime Numbers

You are requested to write a program that establishes whether a number (that the user will input via the keyboard) is a prime number or not (prime numbers are numbers that are only divisible by 1 and themself, e.g. the first twenty prime numbers are 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71 - Note: some regard the number 1 also as a prime).

Create functions for the following:

- Function to test whether a given number is prime or not (returns `true` if the number is prime and `false` otherwise).
- Function to find all prime numbers in a given range (uses function to test for prime for all numbers in given range).
- A `main()` function that presents suitable choices to run/test the application.

Make sure that your code is properly formatted (K&R coding style) and sufficiently commented (top of file, beginning of each function, code within function). Compile and test your program.

`Hint:` The simplest method to establishing the prime property of a number `n` is simply to see if any number between 2 and $\frac{\mathtt{n}}{2}$ divides `n` evenly (strictly speaking, all numbers between 2 and $\sqrt{n}$ are sufficient, but as C++ does not have a native square root function, $\frac{n}{2}$ will do - of course feel free to use square root function from library `<cmath>`). Thus, you have to test whether division of `n` by any number between 2 and $\frac{\mathtt{n}}{2}$ results in an integer value. If any division gives an integer `n` is not prime. If all divisions yield a non-integer number, `n` is prime. Be aware that this is a very inefficient algorithm - many better algorithms do exist, e.g. the "Sieve of Eratosthenes" or the "Sieve of Atkin". Note: a list of prime numbers can be found at `http://en.wikipedia.org/wiki/List_of_prime_numbers#The_first_500_prime_numbers`