Lab: Standard Libraries
CSC 161, “Imperative Problem Solving and Data Structures”
Department of Computer Science · Grinnell College
May 6 and 9, 2016

Pairs

• Anna Blinderman and Faizaan Ali
• Erhaan Ahmad and Tyler Williams
• Sanjay Sudhir and Yuyin Sun
• Jong Hoon Bae and Dennis Chan
• Eli Salm and Ella Nicolson
• Gemma Nash and Jae Hong Shin
• Zachary Susag and Saung Thuya
• Colin Greenman and Sophie Gaschott
• Lilly Webster and Tanner Tufto
• Josh Lavin and Lex Martin
• Ying Zhang and Cory McCartan

Mattori Birnbaum will take the place of anyone who is absent.

Exercises

Exercise 0000: Write and test a C function log2 that computes the base-two logarithm of any positive value of type double. (The base-two logarithm of a positive real number \( x \) is the exponent \( p \) such that \( 2^p = x \).)

Exercise 0001: The great-circle distance between two points on the surface of the earth is the length of the shortest arc from one to the other along the surface of the earth, which for our purposes we’ll take to be a perfect sphere. If we think of the longitude of a point as its angle east (if positive) or west (if negative) of the meridian that runs through the Greenwich Observatory in England, and its latitude as its angle north (if positive) or south (if negative) of the equator, then the great-circle distance between one point, at latitude \( \phi_0 \) and longitude \( \theta_0 \), and another point, at latitude \( \phi_1 \) and longitude \( \theta_1 \), can be computed as

\[
\text{r} \cdot \arccos(\sin \phi_0 \cdot \sin \phi_1 + \cos \phi_0 \cdot \cos \phi_1 \cdot \cos(\theta_0 - \theta_1))
\]

where \( r \) is the radius of the earth, 6377 kilometers, and all of the angles are expressed in radians. Write and test a C function that takes as arguments the latitudes and longitudes of two points \( (\phi_0, \theta_0, \phi_1, \theta_1) \) and computes the great-circle distance between them.

Exercise 0010: Write and test a C function that takes a string str as argument and returns a freshly allocated copy of the substring that lies strictly between the first and second colons in str. (For instance, given the string "abcd:efg:hi" it should return "efg".) Your function should return a null pointer if str does not contain at least two colons.

Exercise 0011: One of the locales available on MathLAN is en_US.utf8. Write and execute a C program that outputs the international currency symbol for this locale.

Exercise 0100: Use the getenv function to write a C function that determines whether the user’s current working directory is their home directory, returning true if it is and false if it is not. (Hint: you can get a list of all of the available environment variables and their values by running the command env in the shell. You’ll be able to spot the relevant variables in this list.)
Exercise 0101: Design, write, and test a program that determines how many days old the driver of your pair is on the day when the program is run, using the functions in the `time` library.

Exercise 0110: Write a C function that takes no arguments, independently generates two random integers in the range from 1 up to and including 6, and returns their sum (simulating the rolling of a pair of dice).

Exercise 0111: Write a C program that calls the function in the preceding exercise one million times and then display the sum of the values of returned, the amount of processor time consumed (in seconds) and the running time of the program (also in seconds).

Advanced Exercises

Exercise 1000: Given one argument, the `−` procedure in Scheme computes the negative (the additive inverse) of that value; given two or more arguments, `−` subtracts all of the other arguments from the first one. Using the variable-argument mechanism, define and test a C function `minus` that works the same way, except that it takes an extra argument, placed ahead of all the others, to indicates how many arguments will follow. (This is the convention that King follows in the `max_int` example on page 678 of our textbook.) So, for instance, `minus(1, 15)` should return $-15$, while `minus(5, 12, -3, 4, 5, -6, 7)` returns 5.

Exercise 1001: Starting from the `stars.c` program from the “Structures, Enumerations, and Unions” lab (which you can copy from the `/home/stone/c/code` directory), write a C program that uses `qsort` to arrange the thirty given stars in the array in alphabetical order by common name, prints out the array in some human-readable form, uses `qsort` to sort them again, this time by absolute magnitude (as computed in exercise 000 of the earlier lab), and prints out the array again.

Exercise 1010: As described on pages 650 and 651 of the textbook, the eight-bit UCS Transformation Format (UTF-8) specifies a mapping between code points (which are integers in the range from 0 up to and including 0xFFFF) and certain byte sequences. Write and test C functions that carry out this transformation in each direction. That is: One function should take a code point as argument and return the corresponding byte sequence; the other should take a byte sequence as argument and return the corresponding code point.