Задания к работе №2 по Математическому практикуму.

Все задания реализуются на языке программирования *C* (стандарт *C99* и выше).

Реализованные приложения не должны завершаться аварийно.

Во всех заданиях запрещено пользоваться функциями, позволяющими завершить выполнение приложения из произвольной точки выполнения, вне контекста исполнения функции *main*.

Во всех заданиях при реализации необходимо разделять контексты работы с данными (поиск, сортировка, добавление/удаление, модификация и т. п.) и отправка данных в поток вывода / выгрузка данных из потока ввода.

Во всех заданиях все вводимые (с консоли, файла, командной строки) пользователем данные должны подвергаться валидации в соответствии с типом валидируемых данных, если не сказано обратное.

Во всех заданиях необходимо контролировать ситуации с невозможностью [пере]выделения памяти; во всех заданиях необходимо корректно освобождать всю выделенную динамическую память.

Все ошибки, связанные с операциями открытия файла, должны быть обработаны; все открытые файлы должны быть закрыты.

Во всех заданиях запрещено использование глобальных переменных.

Во всех заданиях при реализации функций необходимо оставлять “пространство для манёвра” в целях обработки ошибок различных типов на уровне вызывающего кода при помощи возврата целевых результатов функции через параметры функции и возврата значения перечислимого типа (*enum*), репрезентирующего статус-код функции, в целях обработки последнего в вызывающем коде через оператор *switch/case* либо через управляющие конструкции языка *if / else if / else*. Возвращаемые статус-коды функций необходимо продумать самостоятельно так, чтобы были покрыты всевозможные ошибки времени выполнения функций.

1. Через аргументы командной строки программе подаются флаг (первым аргументом), строка (вторым аргументом); и (только для флага -c) целое число типа *unsigned int* и далее произвольное количество строк. Флаг определяет действие, которое необходимо выполнить над переданными строками:
   - -l подсчёт длины переданной строки, переданной вторым аргументом;
   - -r получить новую строку, являющуюся перевёрнутой (reversed) переданной вторым аргументом строкой;
   - -u получить новую строку, идентичную переданной вторым аргументом, при этом каждый символ, стоящий на нечётной позиции (первый символ строки находится на позиции 0), должен быть преобразован в верхний регистр;
   - -n получить из символов переданной вторым аргументом строки новую строку так, чтобы в начале строки находились символы цифр в исходном порядке, затем символы букв в исходном порядке, а в самом конце – все остальные символы, также в исходном порядке;
   - -c получить новую строку, являющуюся конкатенацией второй, четвёртой, пятой и т. д. переданных в командную строку строк; строки конкатенируются в псевдослучайном порядке; для засеивания генератора псевдослучайных чисел функцией *srand* используйте *seed* равный числу, переданному в командную строку третьим аргументом. ./a.out -c seed num st_1 … st_num

   Для каждого флага необходимо реализовать соответствующую ему собственную функцию, выполняющую действие. Созданные функциями строки должны располагаться в выделенной из динамической кучи памяти. При реализации функций запрещено использование функций из заголовочного файла *string.h*. Продемонстрируйте работу реализованных функций.

2. 1. Реализуйте функцию с переменным числом аргументов, вычисляющую среднее геометрическое переданных ей чисел вещественного типа. Количество (значение типа *int*) переданных вещественных чисел задаётся в качестве последнего обязательного параметра функции.

   2. Реализуйте рекурсивную функцию возведения вещественного числа в целую степень. При реализации используйте алгоритм быстрого возведения в степень.

   Продемонстрируйте работу реализованных функций.

3. Реализуйте функцию с переменным числом аргументов, принимающую в качестве входных параметров подстроку и пути к файлам. Необходимо чтобы для каждого файла функция производила поиск всех вхождений переданной подстроки в содержимом этого файла. При реализации запрещается использование функций strstr и strncmp из заголовочного файла *string.h*. Продемонстрируйте работу функции, также организуйте наглядный вывод результатов работы функции: для каждого файла, путь к которому передан как параметр Вашей функции, для каждого вхождения подстроки в содержимое файла необходимо вывести номер строки (индексируется с 1) и номер символа в строке файла, начиная с которого найдено вхождение подстроки. В подстроку могут входить любые символы (обратите внимание, что в подстроку также могут входить символы пробела, табуляций, переноса строки, в неограниченном количестве).

4. 1. Реализуйте функцию с переменным числом аргументов, принимающую координаты (вещественного типа, пространство двумерное) вершин многоугольника и определяющую, является ли этот многоугольник выпуклым.
2. Реализуйте функцию с переменным числом аргументов, находящую значение многочлена степени n в заданной точке. Входными параметрами являются точка (вещественного типа), в которой определяется значение многочлена, степень многочлена (целочисленного типа), и его коэффициенты (вещественного типа, от старшей степени до свободного коэффициента в порядке передачи параметров функции слева направо).
Продемонстрируйте работу реализованных функций.

5. Реализуйте функции overfprintf и oversprintf, поведение которых схоже с поведением стандартных функций fprintf и sprintf, то есть эти функции имеют одинаковый прототип и логику работы, но в ваших функциях помимо стандартных флагов определены следующим образом дополнительные флаги:
    - %Ro – печать в поток вывода целого числа типа *int*, записанного римскими цифрами;
    - %Zr – печать в поток вывода цекендорфова представления целого числа типа *unsigned int* (коэффициенты *0* и *1* при числах Фибоначчи должны быть записаны от младшего к старшему слева направо с дополнительной единицей в конце записи, репрезентирующей окончание записи);
    - %Cv печать целого числа типа *int* в системе счисления с заданным основанием (при обработке флага первым параметром функции, “снимаемым” со стека, является целое число типа *int*, вторым - основание целевой системы счисления в диапазоне *[2..36]* (при основании системы счисления, не входящем в диапазон, значение основания системы счисления устанавливается равным *10*)); символы букв в результирующем строковом представлении целого числа должны быть записаны в нижнем регистре;
    - %CV – аналогично флагу %Cv, при этом символы букв во входном строковом представлении целого числа должны быть записаны в верхнем регистре;
    - %to – печать в поток вывода результата перевода целого числа, записанного в строковом представлении в системе счисления с заданным основанием в систему счисления с основанием *10* (при обработке флага первым параметром функции, “снимаемым” со стека, является строка, описываемая значением типа *char* *, вторым - основание исходной системы счисления в диапазоне *[2..36]* (при основании системы счисления, не входящем в диапазон, значение основания системы счисления устанавливается равным *10*)); символы букв во входном строковом представлении целого числа должны быть записаны в нижнем регистре;
    - %TO - аналогично флагу %to, при этом символы букв во входном строковом представлении целого числа должны быть записаны в верхнем регистре;
    - %mi – печать дампа памяти (байты значения, записанные в системе счисления с основанием *2*, в порядке нахождения в памяти “слева направо”; строковые представления байтов должны сепарироваться одним символом пробела) значения знакового целого 4-байтного числа;
    - %mu – печать дампа памяти (байты значения, записанные в системе счисления с основанием *2*, в порядке нахождения в памяти “слева направо”; строковые представления байтов должны сепарироваться одним символом пробела) значения беззнакового целого 4-байтного числа;
    - %md – печать дампа памяти (байты значения, записанные в системе счисления с основанием *2*, в порядке нахождения в памяти “слева направо”; строковые представления байтов должны сепарироваться одним символом пробела), значения вещественной переменной типа *double*;
    - %mf – печать дампа памяти (байты значения, записанные в системе счисления с основанием *2*, в порядке нахождения в памяти “слева направо”; строковые

представления байтов должны сепарироваться одним символом пробела), значения вещественной переменной типа *float*.

Продемонстрируйте работу реализованных функций.

6. Реализуйте функции overfscanf и oversscanf, поведение которых схоже с поведением стандартных функций fscanf и sscanf соответственно, то есть эти функции имеют одинаковый прототип и логику работы, но в ваших функциях помимо стандартных флагов добавляются дополнительные флаги:
   - %Ro – считывание из потока ввода целого числа типа *int*, записанного римскими цифрами;
   - %Zr – считывание из потока ввода целого числа типа *unsigned int*, записанного в виде цекендорфова представления (коэффициенты *0* и *1* при числах Фибоначчи должны быть записаны от младшего к старшему слева направо с дополнительной единицей в конце записи, репрезентирующей окончание записи);
   - %Cv считывание из потока ввода целого числа типа *int*, записанного в системе счисления с заданным основанием (при обработке флага первым параметром функции, “снимаемым” со стека, является адрес места в памяти типа *int* *, куда необходимо записать считанное значение, вторым - основание системы счисления (в которой записано число, находящееся в потоке ввода) в диапазоне *[2..36]* (при основании системы счисления, не входящем в диапазон, значение основания системы счисления устанавливается равным *10*)); символы букв во входном строковом представлении целого числа должны быть записаны в нижнем регистре;
   - %CV – аналогично флагу %Cv, при этом символы букв во входном строковом представлении целого числа должны быть записаны в верхнем регистре;

   Валидация данных, находящихся во входном потоке, не требуется.

   Продемонстрируйте работу реализованных функций.

   valgrind --leak-check=full ./a.out

7. Реализуйте функцию, которая находит корень уравнения одной переменной методом дихотомии. Аргументами функции являются границы интервала, на котором находится корень; точность (эпсилон), с которой корень необходимо найти, а также указатель на функцию, связанной с уравнением от одной переменной. Продемонстрируйте работу функции на разных значениях интервалов и точности, для различных уравнений.

8. Реализуйте функцию с переменным числом аргументов, вычисляющую сумму переданных целых неотрицательных чисел в заданной системе счисления с основанием *[2..36]*. Параметрами функции являются основание системы счисления, в которой производится суммирование, количество переданных чисел, строковые представления чисел в заданной системе счисления. Десятичное представление переданных чисел может быть слишком велико и не поместиться во встроенные типы данных; для решения возникшей проблемы также реализуйте функцию «сложения в столбик» двух чисел в заданной системе счисления, для её использования при

реализации основной функции. Результирующее число не должно содержать ведущих нулей. Продемонстрируйте работу функции.

9. Реализуйте функцию с переменным числом аргументов, определяющую для каждой из переданных десятичных дробей (в виде значения вещественного типа в диапазоне *(0; 1))*, имеет ли она в системе счисления с переданным как параметр функции основанием конечное представление. Продемонстрируйте работу функции.

10. Реализуйте функцию с переменным числом аргументов, находящую переразложение многочлена со степенями значения *x* по степеням значения *(x - a)*. Входными аргументами этой функции являются точность ε, значение a (вещественного типа), указатель на место в памяти, по которому должен быть записан указатель на динамически выделенную в функции коллекцию коэффициентов результирующего многочлена, степень многочлена (целочисленного типа) и его коэффициенты (вещественного типа), передаваемые как список аргументов переменной длины. То есть, если задан многочлен $f(x)$, то необходимо найти коэффициенты g_0, g_1, ..., g_n такие что:

$$f(x) = f_0 + f_1 x + f_2 x^2 + \ldots + f_n x^n = g_0 + g_1(x - a) + g_2(x - a)^2 + \ldots + g_n(x - a)^n.$$

Продемонстрируйте работу функции.