

Задания к работе №1 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.

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.

1. Через аргументы командной строки программе подаются число и флаг, определяющий действие с этим числом. Флаг начинается с символа '-' или '/'. Программа распознает следующие флаги:

- **-h** вывести в консоль натуральные числа в пределах 100 включительно, кратные указанному. Если таковых нету – вывести соответствующее сообщение;
- **-p** определить является ли введенное число простым; является ли оно составным;
- **-s** разделить число на отдельные цифры и вывести отдельно каждую цифру числа, разделяя их пробелом, от старших разрядов к младшим, без ведущих нулей в строковом представлении;
- **-e** вывести таблицу степеней (для всех показателей в диапазоне от 1 до заданного числа) оснований от 1 до 10; для этого флага работает ограничение на вводимое число: оно должно быть не больше 10;
- **-a** вычислить сумму всех натуральных чисел от 1 до указанного числа включительно и вывести полученное значение в консоль;
- **-f** вычислить факториал указанного числа и вывести полученное значение в консоль.

вероятностный тест простоты, проверка корректности эпсилон

2. Реализуйте функции, вычисляющие значения чисел e , π , $\ln 2$, $\sqrt{2}$, γ с точностью l знаков после запятой. Для каждой константы реализуйте три способа вычисления: как сумму ряда, как решение специального уравнения, как значение предела.

0 - ok

1 - overflow

2 - invalid arguments

Замечание. Вы можете использовать следующие факты:

	Предел	Ряд/Произведение	Уравнение
e	$\checkmark e = \lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$	$\checkmark e = \sum_{n=0}^{\infty} \frac{1}{n!}$	$\checkmark \ln x = 1$
π	$\checkmark \pi = \lim_{n \rightarrow \infty} \frac{(2^n n!)^4}{n((2n)!)^2}$	$\checkmark \pi = 4 \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{2n-1}$	$\checkmark \cos x = -1$
$\ln 2$	$\checkmark \ln 2 = \lim_{n \rightarrow \infty} n \left(2^{\frac{1}{n}} - 1\right)$	$\checkmark \ln 2 = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n}$	$\checkmark e^x = 2$
$\sqrt{2}$	$\checkmark \sqrt{2} = \lim_{n \rightarrow \infty} x_n$, где $x_{n+1} = x_n - \frac{x_n^2}{2} + 1$, $x_0 = -0.5$	$\checkmark \sqrt{2} = \prod_{k=2}^{\infty} 2^{2^{-k}}$	$\checkmark x^2 = 2$
γ	$\checkmark \gamma = \lim_{m \rightarrow \infty} \left(\sum_{k=1}^m C_m^k \frac{(-1)^k}{k} \ln(k!) \right)$	$\checkmark \gamma = -\frac{\pi^2}{6} + \sum_{k=2}^{\infty} \left(\frac{1}{\lfloor \sqrt{k} \rfloor^2} - \frac{1}{k} \right)$	$\checkmark e^{-x} = \lim_{t \rightarrow \infty} \left(\ln t \prod_{p \leq t, p \in P} \frac{p-1}{p} \right)$

Точность вычислений (значение эpsilon) подаётся программе в качестве аргумента командной строки. Продемонстрируйте выполнение реализованных функций.

3. Через аргументы командной строки программе подается флаг, который определяет действие, и набор чисел. Флаг начинается с символа '-' или '/'. Необходимо проверять соответствие количества параметров введённому флагу. Программа распознает следующие флаги:

Осталось всякие проверки сделать на валидность

- -q первый параметр (вещественное число) задаёт точность сравнения вещественных чисел (эпсилон), оставшиеся три (вещественные числа) являются коэффициентами квадратного уравнения; необходимо вывести в консоль решения этого уравнения при всевозможных уникальных перестановках значений коэффициентов при степенях переменной;
- -m необходимо задать два ненулевых целых числа, после чего определить, кратно ли первое число второму;
- -t первый параметр (вещественное число) задаёт точность сравнения вещественных чисел (эпсилон); необходимо проверить, могут ли оставшиеся три (вещественные числа) параметра являться длинами сторон прямоугольного треугольника.

./a.out flag input output

4. На вход программе, через аргументы командной строки, подается флаг и путь к файлу. Флаг определяет действие с входным файлом. Флаг начинается с символа '-' или '/'. Если флаг содержит в качестве второго символа опциональный символ 'n' (то есть "-nd", "/nd", "-ni", "/ni", "-ns", "/ns", "-na", "/na"), то путь к выходному файлу является третьим аргументом командной строки; иначе имя выходного файла генерируется приписыванием к имени входного файла префикса "out_". Вывод программы должен транслироваться в выходной файл. Программа распознает следующие флаги:

- -d необходимо исключить символы арабских цифр из входного файла;

- -i для каждой строки входного файла в выходной файл необходимо записать сколько раз в этой строке встречаются символы букв латинского алфавита;
- -s для каждой строки входного файла в выходной файл необходимо записать сколько раз в этой строке встречаются символы, отличные от символов букв латинского алфавита, символов арабских цифр и символа пробела;
- -a необходимо заменить символы, отличные от символов цифр, ASCII кодом, записанным в системе счисления с основанием 16.

5. Вычислить значения сумм с точностью ϵ , где ϵ (вещественное число) подаётся программе в виде аргумента командной строки:

- a. $\sum_{n=0}^{\infty} \frac{x^n}{n!}; \quad \text{---} \quad \exp(x)$
- b. $\sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!}; \quad \text{---} \quad \cos(x) \quad \text{ОПТИМИЗАЦИЯ}$
- c. $\sum_{n=0}^{\infty} \frac{3^{3n} (n!)^3 x^{2n}}{(3n)!}; \quad \checkmark$
- d. $\sum_{n=1}^{\infty} \frac{(-1)^n (2n-1)!! x^{2n}}{(2n)!!}; \quad \checkmark$

6. Вычислить значения интегралов с точностью ϵ , где ϵ (вещественное число) подаётся программе в виде аргумента командной строки:

- a. $\int_0^1 \frac{\ln(1+x)}{x} dx \quad \text{---} \quad \text{can be represented as a power series}$
- b. $\int_0^1 e^{-\frac{x^2}{2}} dx \quad \begin{array}{l} \text{интегралы расходятся; как считается:} \\ \text{численно, метод прямоугольников,} \\ \text{трапеции 1e-4 точность} \end{array}$
- c. $\int_0^1 \ln \frac{1}{1-x} dx \quad \begin{array}{l} \text{разложить по компонентам в ряд тейлора} \\ \text{и проинтегрировать отдельно каждую} \\ \text{компоненту} \end{array}$
- d. $\int_0^1 x^x dx$

7. На вход программе подаётся флаг и пути к файлам. Флаг начинается с символа '-' или '/'. Необходимо проверять соответствие количества параметров введённому флагу. Программа распознает следующие флаги:

- -r записать в файл, **путь** к которому подаётся **последним** аргументом командной строки, **лексемы** из файлов *file1* и *file2*, пути к которым подаются как **третий** и **четвёртый** аргументы командной строки соответственно, где на нечётных позициях находятся лексемы из *file1*, а на чётных – из *file2*. Лексемы во входных файлах разделяются произвольным количеством символов пробела, табуляций и переносов строк. Если количество лексем в файлах различается, после достижения конца одного из входных файлов, необходимо переписать в выходной файл оставшиеся лексемы из другого из входных файлов. Лексемы в выходном файле должны быть разделены одним символом пробела.

`./a.out <flag> <file1_route> <file2_route> <route_out>`

./a.out <flag> <file_route> <route_out>

- -а записать в файл, путь к которому подаётся последним аргументом командной строки, файл, путь к которому подаётся третьим аргументом командной строки, таким образом, чтобы:
 - в каждой **десятой** лексеме сначала все **символы** букв латинского алфавита были преобразованы в эквивалентные символы **строчных** букв латинского алфавита, а затем все символы были преобразованы в эквивалентные им ASCII-коды, записанные в системе счисления с основанием 4;
 - в каждой **второй** (и одновременно **не десятой**) лексеме все **символы букв латинского алфавита** были преобразованы в эквивалентные символы **строчных** букв латинского алфавита;
 - в каждой **пятой** (и одновременно **не десятой**) лексеме все символы были преобразованы в эквивалентные им **ASCII-коды**, записанные в системе счисления с основанием 8.

Лексемы во входном файле разделяются произвольным количеством символов пробела, табуляций и переносов строк. Лексемы в выходном файле должны быть разделены одним символом пробела.

8. В текстовом файле, путь к которому подаётся как второй аргумент командной строки, находятся числа записанные в разных системах счисления, при этом информация о конкретной системе счисления для каждого числа утеряна. В файле числа разделены произвольным количеством разделителей (символов пробела, табуляций и переносов строки). Для каждого числа из входного файла программа должна определить минимальное основание системы счисления (в диапазоне [2..36]), в которой представление этого числа корректно, и в выходной файл, путь к которому подаётся третьим аргументом командной строки, построчно выводит: входное число без ведущих нулей, определенное для него минимальное основание системы счисления и представление этого числа в системе счисления с основанием 10. Прописные и строчные символы букв латинского алфавита отождествляются.

9. 1. Заполнить массив фиксированного размера псевдослучайными числами в диапазоне [a..b], где a, b задаются в качестве аргументов командной строки. Реализовать функцию, выполняющую поиск максимального и минимального значений элементов массива и меняющую местами максимальный и минимальный элементы в исходном массиве за один проход по нему.

2. Заполнить динамические массивы A и B псевдослучайного размера в диапазоне [10..10000] псевдослучайными числами в диапазоне [-1000..1000]. Сформировать из них динамический массив C, где i-й элемент массива C есть i-й элемент массива A, к которому добавлено значение ближайшего к A[i] по значению элемента из массива B.

10. Пользователь вводит в консоль основание системы счисления (в диапазоне [2..36]) и затем строковые представления целых чисел в системе счисления с введённым основанием (цифры со значением больше 9 должны вводиться как прописные буквы латинского алфавита). Окончанием ввода является ввод строки "Stop". Найти среди введённых чисел максимальное по модулю. Напечатать его без ведущих нулей, а также его строковые представления в системах счисления с основаниями 9, 18, 27 и 36.