

ЛАБОРАТОРНАЯ РАБОТА 3

КАКИЕ ТЕМЫ ЗАТРОНУТЫ

- функции; передача аргументов по значению, по ссылке;
- статические одномерные массивы;

ТРЕБОВАНИЯ К ПРОГРАММАМ

- все требования из лабораторных №1 и №2 (конечно, за исключение запрета на использование массивов);
- размер массива из задания определяется в виде константы, а не разбегается в виде числа по всей программе;
- если в программе говорится о данном (заданном) массиве, то значения элементов заполняются либо с клавиатуры, либо (псевдо)случайным образом по выбору пользователя;
- массивы выводить с использованием форматирования (с указанием ширины поля вывода, а для действительных чисел — еще и точности) после заполнения и изменения;
- повторяющиеся действия выносить в отдельные функции (например, вычисления над парами соседей, расчёт чисел Фибоначчи, проверка на простоту числа) — говоря формальным языком, осуществить *функциональную декомпозицию программы*;
- не использовать глобальные переменные в функциях. Если функция осуществляет операции над массивом или его частью — размер должен передаваться как параметр функции. Исключения:
 - использование в **main** переменных, объявленных с модификатором **const**;
 - по каждой глобальной переменной отдельно: аргументированная обоснованность ее использования в конкретной программе, а также продумывание структуры кода с целью уменьшения вероятности случайного изменения ее значения;

ВСПОМОГАТЕЛЬНЫЕ ФРАГМЕНТЫ

1. Использование генератора псевдослучайных чисел.

Генераторы псевдослучайных чисел (ГПСЧ) представляют собой математический алгоритм получения последовательности чисел: $x_1, x_2, x_3, \dots, x_N$, в которой расположены «как бы» случайные значения. Случайность задаётся алгоритмически. Генераторы полезны в случаях, когда надо создать набор тестовых данных, а вводить руками — не слишком хочется. Как правило, ГПСЧ входит в стандартную библиотеку каждого языка программирования.

Под указанным выше индексом N понимают период последовательности. После получения числа x_N , ГПСЧ начинает выдавать последовательность заново с первого элемента.

Для того, чтобы влиять на получаемую последовательность, обычно ГПСЧ позволяют установить одно или несколько значений, используемых в их алгоритмах для получения последовательности чисел. Такие значения называют «зёрнами генератора». При этом, если начать использовать генератор без указания какого-либо значения в качестве зерна, то будет поставлено заранее определённое значение по умолчанию. И в этом случае программа при каждом последовательном запуске будет использовать один и тот же набор (псевдо)случайных чисел.

Для работы с ГПСЧ будем использовать функции **rand**, **srand** и константу **RAND_MAX** из заголовочного файла **<cstdlib>**. Первая функция создаёт (псевдо)случайное число типа **int**, равномерно распределённое в интервале $[0; \text{RAND_MAX}]$; вторая — позволяет задать *зерно генератора*. Для получения разных зёрен ГПСЧ при последовательных запусках программы воспользуемся функцией **time** из файла **<ctime>**

```
1 // Базовый пример на использование ГПСЧ
2
3 #include <stdio>
4 #include <cstdlib>
5 #include <ctime>
6
7 int main()
8 {
9     /* Ниже используется функция: time(nullptr) из библиотеки ctime.
10      Она возвращает количество секунд, прошедших с 00:00 часов
11      1 января 1970 года. Обычно в таком виде используется
12      инициализация ГПСЧ для получения различных последовательностей
13      чисел при повторных запусках программы.
14
15      nullptr — специальное значение языка C++, связанное с указателями.
16      Подробно будет рассмотрено на пятой лекции.
17
18      */
19     // устанавливаем зерно генератора
20     // Для эксперимента, закомментируйте этот вызов, пересоберите программу
21     // и узнайте, что за значения будут появляться при каждом
22     // последовательном запуске
23     srand( time(nullptr) );
24
25     // получаем некоторое текущее число последовательности
26     int rand_value = rand();
27     printf("random value is %d\n", rand_value);
28
29     printf("next random number is %d\n", rand());
30
31     // получение случайного числа с плавающей точкой в диапазоне
32     // от нуля до единицы
33     double rnd_0_1 = double( rand() ) / RAND_MAX;
34     printf("random real number is %.5f\n", rnd_0_1);
```

34 }

Когда определён расчёт случайного действительного числа в интервале $\text{rnd_0_1} \in [0.0; 1.0]$, то случайное число из любого интервала вида $[a; b]$ может быть определено по формуле: $a + (b - a) \cdot \text{rnd_0_1}$. Оформим универсальную функцию для получения случайного числа на заданном интервале в виде функции:

```
1 #include <stdio>
2 #include <stdlib>
3 #include <ctime>
4
5 double rand_a_b(const double a, const double b);
6
7 int main()
8 {
9     // Не забываем про зерно генератора
10    srand( time(NULLptr) );
11
12    // Случайное действительное число в нужном диапазоне
13    double rnd_real = rand_a_b(0.5, 4.5);
14    printf("random real from range [0.5; 4.5] is %.4f\n", rnd_real);
15
16    // Случайное целое число в нужном диапазоне
17    int rnd_int = rand_a_b(-5; 5);
18    printf("random integer from range [-5; 5] is %d\n", rnd_int);
19 }
20
21 double rand_a_b(const double a, const double b)
22 {
23     double rnd_0_1 = double( rand() ) / RAND_MAX;
24     return a + (b - a) * rnd_0_1;
25 }
```

2. Алгоритм Евклида нахождения наибольшего общего делителя (НОД, он же *greatest common divisor* — gcd):

- (a) получаем остаток от деления большего из двух чисел на меньшее;
- (b) если остаток равен нулю, то второе число и есть НОД;
- (c) иначе — применяем пункт (a) ко второму числу и найденному остатку.

В виде псевдокода:

```
1 // Для чисел a, b: a > b
2
3 while (b != 0) {
4     tmp = b;
5     b = a % b;
6     a = tmp;
7 }
8
9 return a;
```

ЗАДАНИЯ

3.1. Задан массив натуральных чисел из 27 элементов. Найти пару соседних элементов, для которых *наибольший общий делитель* (НОД) минимален. Вывести на консоль индексы и значения найденных соседей.

3.2. Задан массив натуральных чисел из 15 элементов. Найти пары соседних элементов, для которых их сумма является чётным числом. Вывести на консоль индексы и значения найденных соседей.

3.3. Задан массив натуральных чисел из 17 элементов. Найти пары соседних элементов, для которых младший разряд их разности является нечётным числом. Вывести на консоль индексы и значения найденных соседей.

3.4. Задан массив целых чисел из 15 элементов, значения которых находятся в интервале $[-8; 8]$. Найти тройку последовательных элементов, для которых *факториал модуля их суммы* максимален по сравнению с остальными. Вывести на консоль индексы и значения найденных элементов.

3.5. Задан массив действительных чисел из 10 элементов и некоторое пороговое значение `bound_value`. Парой будем считать два элемента, которые равноудалены от начала и конца массива (первый элемент образует пару с последним, второй — с предпоследним и так далее). Найти все пары, среднее значение которых выше порогового значения. Вывести на консоль значения найденных элементов вместе с индексами.

3.6. Задан массив целых чисел из 25 элементов и некоторое целое число `target`. Найти все пары элементов массива, сумма которых равна числу `target`. Вывести на консоль индексы и значения найденных пар.

3.7. Задан массив целых чисел из 22 элементов и некоторое целое число `target`. Найти все пары элементов массива, произведение которых равно числу `target`. Вывести на консоль индексы и значения найденных пар.

3.8. Задан массив целых чисел из 28 элементов и некоторое целое число `target`. Найти все пары элементов массива, одно из чисел которых равно заданному `target`. Вывести на консоль индексы и значения найденных пар.

3.9. Задан массив целых чисел из 31 элемента. Найти все пары элементов массива, в которых оба числа являются чётными. Вывести на консоль индексы и значения найденных пар.

3.10. Задан массив целых чисел из 35 элементов. Найти все последовательные четвёрки элементов массива, в которых значения расположены в порядке убывания. Вывести на консоль исходный массив вместе с начальным и конечным индексами найденных четвёрок.

3.11. Задан массив целых чисел из 18 элементов. Найти все последовательные тройки элементов массива, в которых ровно два из трёх значений являются простыми числами. Вывести на консоль индексы и значения найденных троек.

3.12. Задан массив целых чисел из 25 элементов. Найти все последовательные тройки элементов массива, в которых сумма их значений является нечётным числом. Вывести на консоль индексы и значения найденных троек.

3.13. Задан массив целых чисел, состоящий из 9 элементов (все значения являются двузначными числами). Получить новый массив, состоящий из цифр элементов исходного массива, стоящих в старших разрядах.

3.14. Задан массив целых чисел из 19 элементов. Найти все последовательные четвёрки элементов массива, в которых значения расположены в порядке возрастания. Вывести на консоль исходный массив вместе с начальным и конечным индексами найденных четвёрок.

3.15. Задан массив целых чисел из 17 элементов, значения которых содержат только двухзначные положительные числа. Найти все последовательные пары элементов массива, в которых сумма цифр значений равна. Вывести на консоль индексы и значения найденных пар.

3.16. Задан массив натуральных чисел из 37 элементов. Найти тройку последовательных элементов, для которых *наибольший общий делитель* (НОД) минимален. Вывести на консоль индексы и значения найденных троек. *Примечание:* $\text{НОД}(n_1, n_2, n_3) = \text{НОД}(\text{НОД}(n_1, n_2), n_3)$.

3.17. Задан массив целых чисел из 18 элементов. Упорядочить его значения таким образом, чтобы все отрицательные числа были расположены вначале по возрастанию, а все положительные — в конце по убыванию.

3.18. Задан массив действительных чисел из 28 элементов и некоторое пороговое значение `bound_value`. Найти все последовательные тройки элементов, для которых *квадрат суммы элементов* больше порогового значения. Вывести на консоль индексы и значения найденных троек.

3.19. Задан массив действительных чисел из 21 элемента. Найти все последовательные пары элементов, для которых модуль частного их значений меньше числа π . Вывести на консоль индексы и значения найденных пар.

3.20. Задан массив натуральных чисел из 19 элементов, значения которых находятся в интервале $[1; 35]$. Найти пару соседних элементов, для которых модуль разности чисел Фибоначчи, определяемых их значениями, максимален. Вывести на консоль индексы и значения найденных соседей.

3.21. Задан массив целых чисел из 14 элементов, значения которых находятся в интервале $(9; 10000)$ (обратите внимание, диапазон исключающий границы. Другими словами, допускаются только положительные числа, состоящие из двух, трёх или четырёх разрядов). Заполнить другой массив аналогичной длины по правилу: в каждом значении исходного массива *нулевой и последний* разряды числа меняются местами. Для примера, $\{1278, 34, 296, \dots\} \rightarrow \{8271, 43, 692, \dots\}$

3.22. Задан массив целых чисел из 22 элементов, значения которых находятся в интервале $(9; 1000)$ (обратите внимание, диапазон исключающий границы. Другими словами, допускаются только положительные числа, состоящие из двух, трёх или четырёх разрядов). Разбив массив на 11 последовательных пар, в каждой из них преобразовать значения по правилу: первое число пары домножается на сумму цифр второго числа; из второго — вычитается произведение цифр первого числа. Для примера, $\{ \underbrace{393, 77}_{1\text{-ая пара}}, \underbrace{12, 175}_{2\text{-ая пара}}, \dots \} \rightarrow \{5502, -4, 156, 173, \dots\}$

3.23. Задан массив целых чисел из 11 элементов. Сформировать из него массив уникальных значений с сохранением порядка следования чисел (убрать из исходного массива все повторяющиеся значения). В сформированном массиве найти два элемента, которые при сортировке по возрастанию и по убыванию смещаются со своих позиций наиболее далеко.

3.24. С помощью двух массивов целых чисел из 8 элементов заданы два числа в восьмеричной системе исчисления (элементы массива могут принимать только значения в интервале $[0; 7]$). Под **младшими** разрядами будем понимать разряды с 0-го по 3-ий, под **старшими** — с 4-го по 7-й. Выяснить, совпадают ли старшие или младшие разряды во введенных массивах. Вычислить:

$$f(n_1, n_2) = \begin{cases} \log_{10} |n_1| - \log_{10} |n_2|, & \text{совпадают только старшие} \\ n_1 \cdot n_2, & \text{совпадают только младшие} \\ \sqrt{n_1}, & \text{совпадают обе группы} \end{cases}$$

, n_1 , n_2 - введенные числа в десятичной системе исчисления.

3.25. Задан массив целых чисел из 12 элементов, значения которых содержат только трёхзначные положительные числа. Сформировать новый массив аналогичного размера по правилу: в *нечётных* по порядку элементах поменять местами цифры 0-го и 1-го разрядов, в *чётных* — цифры 1-го и 2-го разрядов. Для примера, $\{825, 354, 187, \dots\} \rightarrow \{852, 534, 178, \dots\}$

3.26. Задан массив действительных чисел из 18 элементов. Найти все последовательные тройки элементов, для которых выполняется неравенство $\ln(\text{arr}[i-1]) < \ln(\text{arr}[i+1]) < \ln(\text{arr}[i])$. Если логарифм от числа не может быть вычислен — неравенство не выполняется. Вывести на консоль индексы найденных троек и их порядок следования в исходном массиве (как пример, «triple #1: 1,2,3», «triple #2: 3,4,5», ...)

3.27. Задан массив целых чисел из 33 элементов. Определить, образуют ли значения его элементов геометрическую прогрессию. В случае положительного ответа — вывести знаменатель прогрессии, иначе — соответствующее сообщение о том, что последовательность не является прогрессией.

3.28. Задан массив целых чисел из 27 элементов. Определить, образуют ли значения его элементов арифметическую прогрессию. В случае положительного ответа — вывести разность прогрессии, иначе — соответствующее сообщение о том, что последовательность не является прогрессией.

3.29. Задан массив целых чисел из 17 элементов и массив целых чисел из 5 элементов. Найти в первом массиве все последовательные пары элементов, сумма которых равна значению одного из элементов второго массива.

3.30. Задан массив целых чисел из 26 элементов. Найти наиболее длинные участки возрастания и убывания значений последовательных элементов. Под «участком» понимаем хотя бы два элемента, значения которых расположены в

соответствующем порядке (либо возрастают, либо убывают). Вывести на консоль индексы границ участков и их длины. Если не найдено ни одного участка (при переходе от элемента к следующему, предыдущий порядок следования меняется) — вывести сообщение.