Требования к программам

1. Программа должна получать все параметры в качестве аргументов командной строки.
2. Задачи оцениваются независимо в двух группах: задачи 1–4 и задачи 5–8.
3. Аргументы командной строки **для задач 1–3**:
   1) n – размерность массива,
   2) p – количество выводимых значений в массиве,
   3) s – задает номер формулы для инициализации массива, должен быть равен 0 при вводе массива из файла,
   4) `filename` – имя файла, откуда надо прочитать массив. Этот аргумент **отсутствует**, если $s! = 0$.

   Например, запуск

   ```
   ./a01.out 4 4 0 a.txt
   ```

   означает, что массив длины 4 надо прочитать из файла `a.txt`, и выводить не более 4-х элементов массива, а запуск

   ```
   ./a01.out 1000000 6 1
   ```

   означает, что массив длины 1000000 надо инициализировать по формуле номер 1, и выводить не более 6-ти элементов массива.
4. Аргументы командной строки **для задач 4, 5, 8**: добавляется **дополнительный первый аргумент**, остальные аргументы – как в задачах 1–4:
   1) X – дополнительный аргумент: $X = k$ в задачах 4, 5 (k – целое число), $X = b$ в задаче 8 (b – имя файла (текстовая строка)),
   2) n – размерность массива,
   3) p – количество выводимых значений в массиве,
   4) s – задает номер формулы для инициализации массива, должен быть равен 0 при вводе массива из файла,
   5) `filename` – имя файла, откуда надо прочитать массив. Этот аргумент **отсутствует**, если $s! = 0$.
5. Аргументы командной строки **для задач 6, 7**:
   1) n – размерность массива,
   2) p – количество выводимых значений в массиве,
   3) `filename` – имя файла b.
6. Ввод массива должен быть оформлен в виде подпрограммы.

7. Ввод массива из файла. В указанном файле находится массив в формате:

   число-1 число-2
   ...
   число-n

   где n - указанный размер массива. Числа разделяются либо пробелами, либо символом новой строки. Программа должна выводить сообщение об ошибке, если указанный файл не может быть прочитан, содержит меньшее количество данных или данные неверного формата.

8. Ввод массива по формуле. Элемент a_i массива A полагается равным

   $$a_i = f(s,n,i), \quad i = 1,\ldots,n,$$

   где $f(s,n,i)$ - функция, которая возвращает значение (i)-го элемента массива по формуле номер s (аргумент командной строки). Функция $f(s,n,i)$ должна быть оформлена в виде отдельной подпрограммы.

   $$f(s,n,i) = \begin{cases} i & \text{при} \quad s = 1 \\ n-i & \text{при} \quad s = 2 \\ i/2.0 & \text{при} \quad s = 3 \\ n-i/2.0 & \text{при} \quad s = 4 \\ 2.0*i & \text{при} \quad s = 5 \\ n-2.0*i & \text{при} \quad s = 6 \end{cases}$$

9. Решение задачи должно быть оформлено в виде подпрограммы, получающей в качестве аргументов массив и его длину (в ряде задач также дополнительные аргументы). Получать в этой подпрограмме дополнительную информацию извне, а также выводить что-либо на экран, запрещается.

10. Программа должна содержать подпрограмму `print_array` вывода на экран массива длины не более p. Эта подпрограмма используется **для вывода исходного массива** после его инициализации, а также для вывода на экран результирующего массива, если массив изменялся. Подпрограмма выводит на экран не более, чем p элементов массива, где p – параметр этой подпрограммы (аргумент командной строки). Каждый элемент массива должен **печататься на новой строке**.

11. Вывод результата работы в функции `main` **для задач 1–7** должен производиться по формату:

```
printf ("New array:\n");
print_array (a, res, p); /* вывод нового состояния массива a */
printf ("%s : Task = %d Result = %d Elapsed = %.2f\n",
        argv[0], task, res, t);
```

   где

   - `argv[0]` – первый аргумент командной строки (имя образа программы),
   - `task` – номер задачи,

- `res` – возвращаемое значение функции, реализующей решение этой задачи,
- `t` – время работы функции, реализующей решение этой задачи.

12. Вывод результата работы в функции `main` **для задачи 8** должен производиться по формату:

```
printf ("%s : Task = %d Result = %d Elapsed = %.2f\n",
        argv[0], task, res, t);
```

где

- `argv[0]` – первый аргумент командной строки (имя образа программы),
- `task` – номер задачи,
- `res` – возвращаемое значение функции, реализующей решение этой задачи,
- `t` – время работы функции, реализующей решение этой задачи.

Задачи

1. Написать функцию, получающую в качестве аргументов массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и выбрасывающую из массива все элементы, меньшие среднего арифметического тех элементов исходного массива, для которых определено понятие среднего геометрического соседних с ними элементов и которые меньше этого среднего геометрического соседних с ними элементов (при выбрасывании элемента $a[i]$ все элементы с номером, большим i, сдвигаются на одну позицию к началу массива и длина массива уменьшается на 1), произведя при этом не более $2n+O(1)$ пересылок элементов. Функция возвращает длину получившегося в результате массива. Основная программа должна заполнять данными массив, выводить его на экран, вызывать эту функцию и выводить на экран результат ее работы.

2. Написать функцию, получающую в качестве аргументов массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и выбрасывающую из массива все элементы, меньшие среднего арифметического элементов исходного массива, находящихся в участках постоянства исходного массива (при выбрасывании элемента $a[i]$ все элементы с номером, большим i, сдвигаются на одну позицию к началу массива и длина массива уменьшается на 1), произведя при этом не более $2n+O(1)$ пересылок элементов. Функция возвращает длину получившегося в результате массива. Основная программа должна заполнять данными массив, выводить его на экран, вызывать эту функцию и выводить на экран результат ее работы.

3. Написать функцию, получающую в качестве аргументов массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и выбрасывающую из массива все элементы, меньшие среднего арифметического элементов исходного массива, находящихся в первом самом длинном участке неубывания исходного массива (при выбрасывании элемента $a[i]$ все элементы с номером, большим i, сдвигаются на одну позицию к началу массива и длина массива уменьшается на 1), произведя при этом не более $2n+O(1)$ пересылок элементов. Функция возвращает длину получившегося в результате массива. Основная программа должна заполнять данными массив, выводить его на экран, вызывать эту функцию и выводить на экран результат ее работы.

4. Написать функцию, получающую в качестве аргументов целое число k, массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и выбрасывающую из массива все элементы, меньшие максимального элемента во всех участках постоянства исходного массива длиной не менее k (при выбрасывании элемента $a[i]$ все элементы с номером, большим i, сдвигаются на одну позицию к началу массива и длина массива уменьшается на 1), произведя при этом не более $2n+O(1)$ пересылок элементов. Функция возвращает длину получившегося в результате массива. Основная программа должна заполнять данными массив, выводить его на экран, вызывать эту функцию и выводить на экран результат ее работы.

5. Написать функцию, получающую в качестве аргументов целое число k, массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и выбрасывающую из массива все элементы, меньшие среднего арифметического элементов исходного массива, находящихся в участках невозрастания исходного массива длиной не менее k (при выбрасывании элемента $a[i]$ все элементы с номером, большим i, сдвигаются на одну позицию к началу массива и длина массива уменьшается на 1), произведя при этом не более $2n+O(1)$ пересылок элементов. Функция возвращает длину получившегося в результате массива. Основная программа должна заполнять данными массив, выводить его на экран, вызывать эту функцию и выводить на экран результат ее работы.

6. Написать функцию, получающую в качестве аргументов имя файла b, массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и присваивающую $a[i]$, $i=0,\ldots,n-1$ значение, равное i-му по минимальной величине элементу этой последовательности, находящемуся в участках постоянства (т.е. $a[0]$ должен стать равным минимальному элементу во всех участках постоянства, $a[1]$ – следующим за ним по величине и т.д.). Функция возвращает количество заполненных элементов в массиве a (например, на постоянной последовательности, это 1). Функция должна возвращать -1, -2 и т.д., если она не смогла открыть файл b, прочитать элемент и т.д.. Основная программа должна выделять память под массив, вызывать эту функцию и выводить на экран результат ее работы.

7. Написать функцию, получающую в качестве аргументов имя файла b, массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и присваивающую $a[i]$, $i=0,\ldots,n-1$ значение, равное i-му по максимальной величине строгому локальному максимуму этой последовательности (т.е. $a[0]$ должен стать равным максимальному элементу x_i такому, что $x_{i-1}<x_i>x_{i+1}$, $a[1]$ – следующим за ним по величине и т.д.). Функция возвращает количество заполненных элементов в массиве a. Функция должна возвращать -1, -2 и т.д., если она не смогла открыть файл b, прочитать элемент и т.д.. Основная программа должна выделять память под массив, вызывать эту функцию и выводить на экран результат ее работы.

8. Написать функцию, получающую в качестве аргументов имя файла b, массив $a[n]$ вещественных чисел, целое число n, являющееся длиной этого массива, и возвращающую целое число, равное количеству вхождений подпоследовательности $a[n]$ в последовательность, содержащуюся в заданном файле b. Под вхождением понимается совпадение n подряд идущих элементов последовательности в файле b с элементами массива $a[0],\ldots a[n-1]$. Например, для массива $a[2]=\{1,1\}$ и последовательности $b=\{1,1,1,1,1\}$ число вхождений равно 4. Функция должна возвращать -1, -2 и т.д., если она не смогла открыть файл b, прочитать элемент и т.д.. Основная программа должна заполнять данными массив, выводить его на экран, вызывать эту функцию и выводить на экран результат ее работы.