

## Требования к программам

1. Программа должна получать все параметры в качестве аргументов командной строки.
2. Задачи оцениваются независимо в двух группах: задачи 1–7 и задачи 8–10.
3. Аргументы командной строки для задач 1–10:
  - 1)  $p$  – максимальное количество выводимых значений списка,
  - 2) `filename` – имя файла, откуда надо прочитать список.

Например, запуск

```
./a.out 6 a.txt
```

означает, что список надо прочитать из файла `a.txt` и выводить не более 6 элементов списка.

4. В файле `filename` находится список строк, по одному элементу списка в строке. Строкой в файле называется последовательность символов до завершающего символа `'\n'`, который не включается в результирующую строку. Длина строк в файле не превосходит 1233 байт. Концом списка при чтении считается конец файла.
5. Узел однонаправленного списка строк задается структурой

```
struct _node;           /* предварительное объявление */
typedef struct _node    /* объявление шаблона структуры */
{
    char * string;       /* данные – строка */
    struct _node * next; /* использование объявления */
} node;                 /* присвоение синонима шаблону */
```

6. Функции в задачах 1–7 имеют прототип

```
int function (node * head);
```

в задачах 8–10 имеют прототип

```
node * function (node * head);
```

7. Программа должна содержать функцию чтения списка, которая должна возвращать  $-1$ ,  $-2$ ,  $-3$  и т.д., если она не смогла открыть файл, прочитать элемент, выделить память и т.д.. В узле списка память под строку выделяется по фактическому размеру строки (длина прочитанной строки без завершающего символа `'\n'` плюс 1).
8. Программа должна содержать подпрограмму вывода на экран не более указанного количества элементов списка. Эта программа должна вызываться после чтения списка, а в задачах 8–10 и после вызова функции, выполняющую указанную в условии задачу. Подпрограмма выводит на экран не более, чем  $p$  первых элементов списка, где  $p$  – параметр этой подпрограммы (аргумент командной строки). Каждый элемент списка должен печататься на новой строке.

9. Количество выделений памяти в программе должно быть равно  $2n$ , где  $n$  – число строк во входном файле.
10. Функция, реализующая задачу, **не должна выделять или использовать дополнительную память**.
11. Вывод результата работы функции в функции `main` должен производиться по формату:

- **Для задач 1–7**

```
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 (1–12),
- `res` – возвращаемое значение результата функции,
- `t` – время работы функции, реализующей решение этой задачи.

- **Для задач 8–10** выводится новое состояние списка, а также

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

где

- `argv[0]` – первый аргумент командной строки (имя образа программы),
- `task` – номер задачи (1–12),
- `len` – длина полученного в результате списка, вычисляемая отдельной подпрограммой, вызываемой из функции `main`,
- `t` – время работы функции, реализующей решение этой задачи.

**Вывод должен производиться в точности в таком формате**, чтобы можно было автоматизировать обработку запуска многих тестов.

## Задачи

1. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное количеству максимальных элементов этого списка.
2. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное количеству элементов этого списка, больших предыдущего элемента.
3. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное количеству нестрогих локальных максимумов этого списка.
4. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное максимальному количеству элементов между соседними нестрогими локальными максимумами этого списка.

5. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное количеству элементов в самом длинном участке строгого возрастания этого списка.
6. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное количеству участков постоянства этого списка.
7. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и возвращающую целое значение, равное максимальному количеству элементов между соседними участками постоянства этого списка.
8. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и выбрасывающую из него все элементы, не меньшие следующего элемента. Функция возвращает указатель на начало получившегося в результате списка.
9. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и выбрасывающую из него все элементы, не меньшие своих двух соседей. Функция возвращает указатель на начало получившегося в результате списка.
10. Написать функцию, получающую в качестве аргумента указатель на начало однонаправленного списка строк, и выбрасывающую из него все участки нестрогого убывания. Функция возвращает указатель на начало получившегося в результате списка.