МГТУ им. Н. Э. Баумана

ИУ 7 – 32

Отчет о лабораторной работе №3

# «Обработка текста. Массив символов, список символов»

Вариант №8

Исаев Д.С.

# Цель работы

Цель работы - обработать текст без использования стандартных библиотек для работы с данными символьного типа, провести. сравнение эффективности обработки текста с использованием символьных массивов и динамических списков, отработать навыки работы со списками.

# Условие задачи

Слова текста из малых латинских букв записаны не менее чем через один пробел; текст оканчивается точкой. БЕЗ ИСПОЛЬЗОВАНИЯ конструкции String и стандартных процедур работы со строками написать программу ввода такого текста с клавиатуры и его обработки, используя: а) массив символов, б) список символов.

Напечатать все слова, отличающиеся от последнего слова, и не содержащие повторяющихся букв.

# Входные данные

Слова текста из малых латинских букв записаны не менее чем через один пробел; текст оканчивается точкой.

**Âûõîäíûå äàííûå:**

Все слова, отличающиеся от последнего слова, и не содержащие повторяющихся букв.

**Ïðèìåð:**

**Âõîäíûå äàííûå**

Friends are forever it is true true.

**Âûõîäíûå äàííûå**

Friends are it is

**Реализация**

**Символьные данные**

Юникодовый символ занимает 2 байта, обычный – 1 байт.

Символы удобно хранить в массивах, образуя строки.

**Массив**

При использовании статического массива весь объем требуемой памяти выделяется на этапе компиляции. В процессе заполнения массива можно обеспечить запоминание количества элементов в нем, организуя тем самым работу только с заполненной областью памяти. Оставшаяся часть памяти в этом случае расходуется непроизводительно.

Для оптимизации расходования памяти можно использовать динамический массив. Его преимуществом является то, что память под динамический массив выделяется во время выполнения программы по соответствующему запросу. Однако следует учитывать, что при необходимости добавить в массив один элемент придется каждый раз запрашивать память для создания динамического массива новой длины, что потребует времени на поиск непрерывного участка памяти (концепция массива требует, чтобы соседние элементы располагались в последовательных байтах памяти), и также выделение этой памяти, переписи в нее данных из «старого» массива и сохранения новой длины. Таким образом, уйдя от непроизводительной потери памяти, мы придем к дополнительному расходу времени. Следовательно, изменение размера динамического массива на единицу – очень ресурсоемкое и нецелесообразное действие. Память под динамический массив необходимо запрашивать «разумными» частями, чтобы минимизировать возможные потери ресурсов. Обычно новый массив удваивает используемый объём памяти.

Физическая структура и статического и динамического массива дополняется дескриптором, который хранит имя массива, адрес первого элемента, описание элемента, его длину и размер массива.

**Список**

Список – это динамическая структура данных, каждый элемент которой, содержит данные и адрес следующего элемента (указатель на следующий элемент). Этот адрес у последнего элемента – нулевой. Память под каждый элемент списка выделяется динамически по мере необходимости. При этом в статической памяти хранится адрес первого элемента, или нулевой указатель, если список пуст.

Дескриптор списка содержит имя списка, адрес первого элемента, количество элементов, описание структуры элемента и его длину.

Для доступа к элементу списка необходим просмотр списка с «головы». Такой доступ называется последовательным. В элементе списка можно хранить адрес не только последующего, но и предыдущего элемента. В этом случае список называется двусвязным. Адрес предыдущего у первого элемента и последующего у последнего элемента – нулевой. Двусвязные списки можно просматривать в двух направлениях, т.е. доступ к элементам двусвязного списка возможен с любого конца. Если сделать список кольцевым (то есть в указатель последнего элемента записать адрес первого элемента), то начинать просмотр можно с любого элемента, и в дескрипторе можно хранить адрес любого элемента. Для односвязного, двусвязного, а также для кольцевого списка необходимо хранить в статической памяти хотя бы один указатель, содержащий адрес элемента списка.

Главными операциями над списками являются вставка элемента в список, исключение элемента из списка, поиск элемента в списке. При этом необходимо обеспечить вывод соответствующих сообщений при попытке удаления элемента из пустого списка и не забывать освобождать память при удалении элемента из списка.

При удалении элемента память освобождается, т.е. она может быть снова выделена для других переменных.

**Результаты**

В результате работы программы были получены следующие данные:

Размер текста: 500 букв

Количество повторов: 10000

Время обработки:

* Списком – 1.73 с
* Массивом – 1.4 с

Затраты памяти:

* Списком – 6 байт на элемент
* Массивом – 2 байта н элемент

Результаты показывают, что использование массива для поставленной задачи и эффективнее по быстродействию и экономнее по памяти.

Список выгодно использовать в случае частой модификации текста, вставки в произвольное место и удаления текста. Для более стабильных операций – вставки в конец и других выгоднее использовать массив. Причем динамический, автоматически изменяющий размер, так как это более экономно по памяти.

**Вывод:**

Из результатов работы программы видно, что в данной технической задаче выгоднее использовать массив, так как операции удаления текста и вставки применяются здесь довольно редко.