Практическое занятие № 6.
Работа с массивами данных. Массивы. Сортировка массивов. Применение функции ко всем элементам массива. Выделение подмассива.

Цель работы:

1. Изучение возможностей современных средств защиты документов, созданных в среде приложений OS Microsoft.
2. Закрепление теоретического материала.
3. Изучение способов и систем защиты файлов и файловых систем.

Учебно-наглядные пособия и ТСО: ПЭВМ, ОС Windows, пакет Microsoft Office, методическое пособие.

Приемы практической работы с массивами продемонстрируем на примерах выполнения конкретных операций: сортировка массива, поиск в массиве, транспонирование матрицы и др. На примере работы с массивами очень удобно обсуждать особенности реализации этих операций, так как в общем случае элементы массива могут быть не просто скалярами, но и более сложными по структуре данными. Поэтому дальнейшее наше изложение помимо демонстрации приемов работы с массивами в программах на ассемблере будет посвящено рассмотрению возможных подходов к реализации нескольких классов практически важных алгоритмов.

Под сортировкой понимается процесс перестановки элементов некоторого множества в порядке убывания или возрастания. Если элементы множества – скаляры, то сортировка ведется относительно их значений. Если элементы множества – структуры, то каждая структура должна иметь поле, относительно которого будет производиться упорядочивание местоположения элементов (то есть сортировка) относительно других элементов-структур множества.

*Различают два типа алгоритмов сортировки: **сортировку массивов** и **сортировку файлов**. Другое их название – алгоритмы внутренней и внешней сортировки. Разница заключается в местонахождении объектов сортировки: для внутренней сортировки – это оперативная память компьютера, для внешней – файл. В данном разделе будут рассмотрены алгоритмы внутренней сортировки массивов. Алгоритмы внешней сортировки будут рассмотрены в разделе, посвященном работе с файлами.*

Существует несколько алгоритмов сортировки массивов, которым следует уделить внимание в контексте проблемы изучения ассемблера. По критерию эффективности алгоритмы сортировки массивов делят на простые и улучшенные. Среди простых методов сортировки, которые также называют сортировками "на том же месте", мы рассмотрим следующие:

сортировка прямым включением; сортировка прямым выбором;

сортировка прямым обменом.

Улучшенные методы сортировки в нашем изложении будут представлены следующими алгоритмами:

сортировка Шелла; сортировка с помощью дерева; быстрая сортировка.

Сортировка прямым включением

Идея сортировки прямым включением (программа **prg4_96.asm**) заключается в том, что в сортируемой последовательности masi длиной n (i = 0..n – 1) последовательно выбираются элементы начиная со второго (i< 1) и ищутся их местоположения в уже отсортированной левой части последовательности. При этом поиск места включения очередного элемента x в левой части последовательности mas может закончиться двумя ситуациями:

- найден элемент последовательности mas, для которого masi<x;
- достигнут левый конец отсортированной слева последовательности.

Первая ситуация разрешается тем, что последовательность mas начиная с masi раздвигается в правую сторону и на место masi перемещается x. Во второй ситуации следует сместить всю последовательность вправо и на место mas0 переместить x.

В этом алгоритме для отслеживания условия окончания просмотра влево отсортированной последовательности используется прием "барьера". Суть его в том, что к исходной последовательности слева добавляется фиктивный элемент X. В начале каждого шага просмотра влево отсортированной части массива элемент X устанавливается в значение того элемента, для которого будет осуществляться поиск места вставки.

```
ПРОГРАММА prg4_96
//prg4_96 – программа на псевдоязыке сортировки прямым включением
//Вход: mas[n] – неупорядоченная последовательность байтовых двоичных значений.
//Выход: mas[n] – упорядоченная последовательность байтовых двоичных значений.
//.---------…----…--……
ПЕРЕМЕННЫЕ
INT_BYTE n=8: //количество элементов в сортируемом массиве
INT_BYTE mas[n]: //сортируемый массив размерностью п (О..п-l)
INT_BYTE X; //барьер
```

```
INT_BYTE i=0: j=0 //индексы
НАЧ_ПРОГ
ДЛЯ 1:-1 ДО п-1 /Л изменяется в диапазоне О..п-l
НАЧ_БЛ0К_1
//присвоение исходных значений для очередного шага
X: = mas[i]: mas[0]: = X; j: = i-l
ПОКА (X<mas[j]) ДЕЛАТЬ НАЧ_БЛОК_2
mas[j+l]: = mas[j]; j: = j-l КОН_БЛ0К_2
mas[j+l]: = X
КОН_БЛОК_1 КОН_ПРОГ
;prg4_96.asm – программа на ассемблере сортировки прямым включением.
.data
mas db 44.55.12.42.94.18.06.67:задаем массив
n=$-mas
X db 0:барьер
.code
mov ex.п-1:цикл по 1
movsi.l:i=2 cyc13: присвоение исходных значений для очередного шага
mov al,mas[si]
movx.al:X: = mas[i]: mas[0]:-X: j: = i-l
push si;временно сохраним i. теперь J-1:еще один цикл, который перебирает элементы
; до барьера x=mas[i] cyc12: mov al,mas[si-l]
emp x,al;сравнить x < mas[j-l]
ja ml:если x > mas[j-l]: если x < mas[j-l]. то
mov al,mas[si-l]
irovmas[si],al
dec si
jmpcycl2 ml: mov al,x
movmas[si].al
pop si
incsi
loop cyc!3
```

Применение функции ко всем элементам массива. Функция array_walk(массив, функция [, данные]) применяет созданную пользователем функцию ко всем элементам массива массив и возвращает true в случае успешного выполнения операции и false – в противном случае.

Пользовательская функция, как правило, имеет два аргумента, в которые поочередно передаются значение иключ каждого элемента массива. Но если при вызове функции array_walk() указан третий аргумент, то он будет рассмотрен как значение третьего аргумента пользовательской функции, смысл которого определяет сам пользователь. Если функция пользователя требует больше аргументов, чем в нее передано, то при каждом вызове array_walk() будет выдаваться предупреждение.

Если необходимо работать с реальными значениями массива, а не с их копиями, следует передавать аргумент в функцию по ссылке. Однако нужно иметь в виду, что нельзя добавлять или удалять элементы массива и производить действия, изменяющие сам массив, поскольку в этом случае результат работы array_walk()считается неопределенным.

Выделение подмассива Функция array_slice

Поскольку массив – это набор элементов, вполне вероятно, потребуется выделить из него какой-нибудь поднабор. В PHP для этих целей есть функция array_slice. Ее синтаксис таков:

array_slice (массив,
номер_элемента [, длина])

Эта функция выделяет подмассив длины длина в массиве массив, начиная с элемента, номер которого задан параметром номер_элемента. Положительный номер_элемента указывает на порядковый номер элемента относительно начала массива, отрицательный – на номер элемента с конца массива.

Ход выполнения лабораторной работы:

1. Изучить приведенный в методическом описании к практической работе материал.
2. Выполнить приведенный пример программы, оформить отчет в установленной форме.
3. Ответить на контрольные вопросы.
4. Представить программу и отчет по лабораторной работе преподавателю.

Контрольные вопросы:

1. Перечислите виды операций с массивами. Дайте понятие сортировки.
2. Перечислите типы алгоритмов сортировки.
3. Какие действия предполагает сортировка прямым включением?
4. Какие действия предполагает применение функции ко всем элементам массива?