Лабораторная работа № 7

КОМАНДЫ ПЕРЕДАЧИ УПРАВЛЕНИЯ. ЦИКЛЫ

Цель работы

Целью работы является приобретение навыков написания программ с циклами на языке Ассемблер.

Порядок выполнения работы

1. Создать рабочую папку для текстов программ на ассемблере и записать в нее файлы tasm.exe, tlink.exe, rtm.exe и td.exe из пакета tasm, а также файл с исходным текстом программы на ассемблере, который сохранить с именем prog7.asm.
2. Создать загрузочный модуль, протестировать работу программы.

Содержание отчета:

1. Цель работы.
2. Постановка задачи (Приложение 1).
3. Листинг программы.
4. Блок-схема для каждого пункта задания.
5. Результаты работы программы для каждого пункта задания.
6. Вывод.

Постановка задачи

Создать программу обработки числовых массивов используя циклические структуры и макросы для ввода и вывода десятичных чисел.

Теоретическая часть

Цикл – это просто-напросто блок кода, завершающийся условным переходом, благодаря чему данных блок может выполняться повторно до достижения условия завершения.
Возможно, вам уже знакомы такие конструкции циклов, как **for** и **while** в языке **Си**, **while** и **repeat** в **Паскале**.

Циклы служат для:

- работы с массивами,
- проверки состояния портов ввода-вывода до получения определенного состояния,
- очистки блоков памяти,
- чтения строк с клавиатуры,
- и вывода их на экран и т.д.

В ассемблере классический цикл можно реализовать следующим образом
Пусть некоторую группу команд (тело цикла) надо повторить N раз (N>0).

```
   MOV CX,N        ;CX — счетчик цикла
LOOP: …             ;тело цикла
   DEC CX          ;CX-1→CX
   CMP CX,0        ;CX=0?
   JNE LOOP        ;если CX не равен 0 идти к метке LOOP
```

Создатели языка ассемблера обратили внимание, что в конце таких циклов всегда применяется одна и та же тройка команд (DEC CX; CMP CX,0; JNE LOOP). Учитывая это, в систему команд была введена специальная команда LOOP («петля»), которая объединяет в себе действие всех этих команд.

Команды этого вида организуют циклические вычисления (LOOP- цикл), используя регистр счетчика CX по своему прямому назначению.

В регистр CX должно быть предварительно занесено количество повторений цикла.

Циклы - это основное средство, которое используется для выполнения повторяющихся действий.

Поэтому используются они довольно часто, настолько часто, что в наборе инструкций процессора 8086 предусмотрено фактически несколько инструкций циклов:

Мнемокоды/ синонимы	**Описание**
LOOP	Циклическое выполнение, пока содержимое CX не равно **нулю**
Особенностью последующих команд является то, что они анализируют не только содержимое регистра **CX**, но и состояние флага **ZF.**	
LOOPNE **LOOPNZ**	Циклическое выполнение, пока не равно/ циклическое выполнение, пока **не нуль.** Оба обозначения представляют собой синонимы и относятся к одной команде. **Если ECX/CX=0 то, передать управление следующей за** LOOPxx команде, **Если ECX/CX> 0 и ZF = 0, то выход из цикла** Команда выполняет декремент содержимого регистра CX, и если оно **не равно нулю, и флаг ZF сброшен, осуществляется переход на указанную метку**.
LOOPE **LOOPZ**	Циклическое выполнение, пока равно/ циклическое выполнение, пока **нуль** Оба обозначения представляют собой синонимы и относятся к одной команде. **Если ECX/CX=0 то, передать управление следующей за** LOOPxx команде, **Если ECX/CX> 0 и ZF = 1, то выход из цикла** Команда выполняет декремент содержимого регистра CX, и если оно **не равно нулю, и флаг ZF установлен, осуществляется переход на указанную метку**.
JCXZ	Циклическое выполнение, пока содержимое CX не равно **нулю**

Примечание. Нужно помнить о том, что флаг нуля устанавливается в значение 1, если результат последней арифметической операции был нулевым или два операнда в последней операции сравнения не совпадали).

Циклы со счетчиком с постусловием

Команда LOOP – переход по счетчику

В системе команд микропроцессора x86 циклы реализуются, главным образом, с помощью команды **LOOP (петля),** хотя имеются другие способы организации циклов.
Относительно инструкций циклов можно сделать **пару интересных замечаний**.
Во-первых. Во всех случаях число шагов в цикле определяется содержимым регистра CX/ECX, поэтому максимальное число шагов составляет 65536.
Во-вторых. Команда LOOP выполняет декремент содержимого регистра CX и если оно не равно 0 осуществляет переход на указанную метку вперед или назад в том же программном сегменте в диапазоне **от –128 до +127.**
! Циклы, превышающие 128 байт, требуют использования условных переходов с помощью безусловных переходов.
В-третьих. Обычно метка помещается перед первым предложением тела цикла, а команда LOOP является последней командой цикла.
В-четвертых. Команда не воздействует на флаги процессора.

Команда LOOP реализует **классический цикл со счетчиком с постусловием**.

Синтаксис команды: LOOP короткая_метка

Логика работы команды:

```
    <CX> = Counter
Short_label: выполнение тела цикла
    <CX>=<CX> - 1
```

If (<CX> <>0) goto short_label

Аналог реализации команды LOOP в Ассемблере:

```
        MOV CX, Counter
Short_label:
;       Выполнение тела цикла
;       Проверка условия ПРОДОЛЖЕНИЯ цикла
        DEC CX
        CMP CX, 0
        JNE short_label
```

DEC- Декремент, уменьшает содержимое регистра на 1, затем передает управление метке, если содержимое CX не равно 0.

Команда LOOP уменьшает содержимое регистра CX на 1, затем передает управление метке, если содержимое CX не равно 0.
Слайд

ПРИМЕЧАНИЕ.
! Если по какой-либо причине перед началом цикла регистр CX/ECX равен 0, то тело цикла все равно будет выполнено, из содержимого регистра CX/ECX будет отнята 1 и уже после этого регистр CX/ECX будет проверен на равенство 0.
Что, по вашему, окажется в регистре CX/ECX?
Правильно, содержимое регистра CX/ECX равно —1 (0FFFFh/0FFFFFFFFh) и цикл, вместо того чтобы остановиться, будет запущен еще 65535 раз, если мы используем регистр CX, или 4 миллиона раз (!), если использовали регистр ECX, — это один из источников ошибок.
Поэтому, если возможен вариант, что число повторений может быть и нулевым, то при CX /ECX =0 надо делать обход цикла:
обычно **до начала цикла** проверяют содержимое регистра CX на ноль.
Таким образом, стандартная последовательность команд для организации цикла со счетчиком имеет следующий вид:

```
        MOV CX, Counter
        JCXZ ExitCicle ; если CX=0, цикл обойти
Short_label:
        ; Выполнение тела цикла
        LOOP short_label
ExitCicle:
```

Пример 1. Пусть требуется сформировать ряд натуральных чисел от 0 до 4095.

```
; в полях данных:
array dw 4096 dup (?)           ; массив из 4096 слов
; в программном сегменте:
    LEA  BX, array              ; BX→array
    XOR  SI, SI                 ; SI=0
    MOV CX, 4096                ; счетчик повторений
    MOV  AX, 0                  ; начальное значение заполнителя
arr: MOV [BX] [SI], AX          ; очистка элемента массива
    INC      SI                     ; сдвиг к следующему
    INC      SI                     ; слову массива
```

```
INC     AX                  ; инкремент заполнителя
LOOP    arr                 ; повторить CX раз
```

Примечание: команда LEA загружает в регистр указанный в команде в качестве первого операнда относительный адрес второго операнда.
SI- регистр используется при исполнении команд перемещения. Он определяет смещение начала данных, которые должны быть перемещены.
Так как мы формируем ряд **двухбайтовых слов,** то операцию модификации индекса **INC SI** приходится повторять дважды, если нужно заполнить байтовый массив, то в двойном повторении нет смысла.
В данном примере следует поговорить об основных типах операндов

Команда LOOPE (LOOPZ) переход по счетчику и если равно

Данная команда имеет два равнозначных мнемонических имени.
LOOP**E** (if Equal – если равно – более человеческая команда).
LOOP**Z** (if Zero – если Ноль – более машинная команда, т.к предполагает знание того факта, что если два операнда равны, флаг нуля ZF=1 (установлен)).
Синтаксис команды:

```
LOOPE короткая_метка
LOOPZ короткая_метка
```

Логика работы команды:

```
    <CX> = Counter
Short_label: выполнение тела цикла
    <CX>=<CX> – 1
    If (<CX> <>0 and <ZF> =1) goto short_label
```

Команда LOOPE используется для организации цикла с известным числом повторений, из которого возможен досрочный выход.
До начала цикла в регистр CX /ECX записывается число повторений. Команда
LOOPE ставится в конец цикла, а перед ней помещается команда, меняющая флаг ZF — обычно это команда сравнения CMP.
Команда **LOOPE** заставляет цикл повторяться CX/ECX раз, но только если команда CMP фиксирует неравенство сравниваемых величин — происходит выход из цикла.
По какой именно причине произошел выход из цикла (по ZF=0 или CX/ECX=0), надо проверять после цикла.
Чаще всего команда LOOPE используется для поиска первого элемента некоторой последовательности, отличного от заданной величины.

Команда LOOPZ — синоним команды LOOPE.
Команда **LOOPNE** аналогична команде LOOPE, но выход из цикла осуществляется при ZF=1, если предыдущая команде LOOPNE команда CMP зафиксировала равенство или по CX/ECX=0.

Все то, что говорилось для команды LOOP, справедливо и для команды LOOPE(Z), добавляется еще проверка флага ZF.
Применяется данная команда в случае, если нужно досрочно выйти из цикла, как только находится первый элемент, отличный от заданной величины.

Пример 2. Сканирование массива, пока не встретится нулевое значение.
Команда LOOPZ позволяет легко проходить по массиву целых чисел, пока не встретится нулевое значение. Так как команда CMP сравнивает каждое значение с 0, то флаг нуля будет

установлен или сброшен. Если встречается ненулевое значение, то команда LOOPZ не делает переход на метку next.

```
. data
Intarray dw 0, 0, 0, 0, 1, 20, 35,-12, 66, 4, 0
Array_Size =($- Intarray)/2

.code
      MOV BX, offset Intarray     ; указать на массив
      SUB  BX, 2                  ; отступить на одну позицию
      MOV EX, Array_Size  ; повторить Array_Size раз
Next:                             ; начало цикла
      ADD       BX, 2                   ; перейти к следующему значению
      CMP [Bx], 0                 ; сравнить значение с 0
      LOOPZ     Next                    ; повторять, пока ZF=1, CX>0
```

Выражение **($-строка) –** дает команду ассемблеру рассчитать длину строки, вычитая начальное смещение строки из текущего значения счетчика команд.

Команда LOOPNE (LOOPNZ) переход по счетчику и если НЕ равно

Команда LOOPNE обычно используется для поиска в некоторой последовательности первого элемента, имеющего заданную величину.

Данная команда имеет два равнозначных мнемонических имени.

LOOP**NE** (if Not Equal – если не равно – более человеческая команда).

LOOP**NZ** (if Not Zero – если не Ноль – более машинная команда, т.к. предполагает знание того факта, что если два операнда равны, флаг нуля ZF=0 (сброшен)).

Синтаксис команды:

```
LOOPNE короткая_метка
LOOPNZ короткая_метка
```

Логика работы команды:

```
      <CX> = Counter
Short_label: выполнение тела цикла
      <CX>=<CX> – 1
      If (<CX> <>0 and <ZF> =0) goto short_label
```

Все то, что говорилось для команды LOOP, справедливо и для команды LOOPE(Z), добавляется еще проверка флага ZF.

Применяется данная команда в случае, если нужно досрочно выйти из цикла, как только находится первый элемент, равный заданной величине.

Пример 3. Вычислить значение суммы чисел натурального ряда: S=1+2+3+…+n.Вычисления закончить, как только значение суммы станет равным некоторому числу **k** или не будет перебраны все **n** чисел.

```
; в программном сегменте:
      MOV CX, n             ; количество повторений
      XOR  AX, AX           ; начальное значение заполнителя
      XOR  SI, SI
      JCXZ   Exit           ; if CX =0 then Exit
Begin:                            ; начало цикла
      INC       SI                  ; сдвиг к следующему
      ADD       AX, SI
      CMP AX, k             ; выход из цикла, если AX=k или CX=0
      LOOPNE   Begin
Exit:
      MOV s, AX
```

ПРИЛОЖЕНИЕ 1

Варианты

Вариант 1
1. Организовать ввод массива 2, -34, 56, 89, 65, -34, 0, -12, 14, 5 с клавиатуры.
2. В заданном числовом массиве определить сумму положительных элементов.
3. В заданном числовом массиве каждый элемент с четным индексом заменить на нуль.
4. .Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 2
1. Организовать ввод массива 8, 0, -45, 98, -15, 2, 19, -72, 35, -4 с клавиатуры.
2. .В заданном числовом массиве определить среднее арифметическое отрицательных элементов.
3. В заданном числовом массиве удвоить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 3
1. Организовать ввод массива 4, 0, 87, -34, -25, 13, 87, 0, 20, 0 с клавиатуры.
2. В заданном числовом массиве определить количество нулевых элементов.
3. В заданном числовом массиве определить индексы элементов, имеющих значение 87.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 4
1. Организовать ввод массива 15, -46, -15, 0, -85, 74, 15, 0, 17, 15 с клавиатуры.
2. В заданном числовом массиве определить количество элементов, имеющих значение 15.
3. В заданном числовом массиве определить среднее арифметическое элементов с четными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 5
1. Организовать ввод массива -67, 1, 78, -9, -2, -1, 34, 72, -13, 0 с клавиатуры.
2. В заданном числовом массиве определить сумму отрицательных элементов.
3. В заданном числовом массиве каждый элемент с четным индексом заменить на единицу.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 6
1. Организовать ввод массива 56, -18, 39, 12, -12, 0, 69, 38, -3, -93 с клавиатуры.
2. В заданном числовом массиве определить среднее арифметическое положительных элементов.
3. В заданном числовом массиве удвоить элементы с четными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 7
1. Организовать ввод массива -53, 0, 57, -35, 0, 53, 0, -14, 53, 62 с клавиатуры.
2. В заданном числовом массиве определить количество ненулевых элементов.
3. В заданном числовом массиве определить индексы элементов, имеющих значение 53.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 8

1. Организовать ввод массива -18, 0, 17, 94, -23, 17, 90, -17, 64, -3 с клавиатуры.
2. В заданном числовом массиве определить количество элементов, имеющих значение 17.
3. В заданном числовом массиве определить среднее арифметическое элементов с нечетными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 9

1. Организовать ввод массива 45, -19, 0, 3, 0, -56, 91, 38, 14, 29 с клавиатуры.
2. В заданном числовом массиве определить произведение положительных элементов.
3. В заданном числовом массиве каждый элемент с нечетным индексом заменить на нуль.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 10

1. Организовать ввод массива -15, -84, 76, 82, -4, 37, 29, 0, -12, 0 с клавиатуры.
2. В заданном числовом массиве определить среднее геометрическое отрицательных элементов.
3. В заданном числовом массиве утроить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 11

1. Организовать ввод массива 38, 0, 23, 91, 0, -13, 23, 10, 46, 23 с клавиатуры.
2. В заданном числовом массиве определить сумму ненулевых элементов.
3. В заданном числовом массиве определить количество элементов, имеющих значение 23.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 12

1. Организовать ввод массива 39, 70, 46, -17, -19, 46, 46, 0, -20, 46 с клавиатуры.
2. В заданном числовом массиве определить индексы элементов, имеющих значение 46.
3. В заданном числовом массиве определить произведение элементов с четными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 13

1. Организовать ввод массива 6, 39, 0, -56, 42, -36, -60, 99, -82, 5 с клавиатуры.
2. В заданном числовом массиве определить среднее арифметическое положительных элементов.
3. В заданном числовом массиве каждый четный элемент заменить на нуль.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 14

1. Организовать ввод массива 49, -28, 0, -17, -2, 30, 49, -67, 54, 10 с клавиатуры.
2. В заданном числовом массиве определить среднее геометрическое отрицательных элементов.
3. В заданном числовом массиве удвоить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 15

1. Организовать ввод массива -10, 0, 27, 0, -59, 18, 27, 0, -72, 0 с клавиатуры.
2. В заданном числовом массиве определить сумму ненулевых элементов.
3. В заданном числовом массиве определить сумму индексов элементов, имеющих значение 27.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 16

1. Организовать ввод массива 41, 31, -35, 0, 41, -92, 25, 41, 0, 49 с клавиатуры.
2. В заданном числовом массиве определить сумму индексов элементов, имеющих значение 41.
3. В заданном числовом массиве определить среднее арифметическое элементов с четными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 17

1. Организовать ввод массива 60, -24, 9, 2, -15, 52, 0, 35, -28, 14 с клавиатуры.
2. В заданном числовом массиве определить сумму положительных элементов.
3. В заданном числовом массиве каждый элемент с четным индексом заменить на нуль.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 18

1. Организовать ввод массива 0, -28, -16, 83, 62, -18, 42, 90, 0, -26 с клавиатуры.
2. В заданном числовом массиве определить среднее арифметическое отрицательных элементов.
3. В заданном числовом массиве удвоить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 19

1. Организовать ввод массива 56, 0, 87, -27, 0, 87, 0, -28, 87, 0 с клавиатуры.
2. В заданном числовом массиве определить количество нулевых элементов.
3. В заданном числовом массиве определить индексы элементов, имеющих значение 87.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 20

1. Организовать ввод массива -15, 0, 47, 15, 61, -94, 15, 18, 15, 82 с клавиатуры.
2. В заданном числовом массиве определить количество элементов, имеющих значение 15.
3. В заданном числовом массиве определить среднее арифметическое элементов с четными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 21.

1. Организовать ввод массива -20, 84, 0, 65, -28, 13, 54, -71, -40, 51 с клавиатуры.
2. В заданном числовом массиве определить сумму положительных элементов.
3. В заданном числовом массиве каждый элемент с четным индексом заменить на нуль.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 22.

1. Организовать ввод массива 2, -39, 41, 17, 39, -12, 30, -48, 0, -71 с клавиатуры.

2. В заданном числовом массиве определить среднее арифметическое отрицательных элементов.
3. В заданном числовом массиве удвоить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 23

1. Организовать ввод массива 0, -29, 87, 34, 0, 87, 87, 0, 0, 33 с клавиатуры.
2. В заданном числовом массиве определить количество нулевых элементов.
3. В заданном числовом массиве определить индексы элементов, имеющих значение 87.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 24

1. Организовать ввод массива -15, 15, 48, 77, 15, -37, 0, 15, -10 , 93 с клавиатуры.
2. В заданном числовом массиве определить количество элементов, имеющих значение 15.
3. В заданном числовом массиве определить среднее арифметическое элементов с четными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 25

1. Организовать ввод массива 33, -19, 0, 50, -26, 13, 61, -92, -16, 0 с клавиатуры.
2. В заданном числовом массиве определить произведение положительных элементов.
3. В заданном числовом массиве каждый элемент с нечетным индексом заменить на нуль.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 26

1. Организовать ввод массива 48, -48, 0, 77, 91, -38, 17, 1, -6, 53 с клавиатуры.
2. В заданном числовом массиве определить среднее геометрическое отрицательных элементов.
3. В заданном числовом массиве утроить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.

Вариант 27

1. Организовать ввод массива 3, 0, -45, 0, 23, -6, 23, 0, 67, 23 с клавиатуры.
2. В заданном числовом массиве определить сумму ненулевых элементов.
3. В заданном числовом массиве определить количество элементов, имеющих значение 23.
4. В заданном числовом массиве каждый элемент, начиная со второго, заменить на значение предыдущего элемента.

Вариант 28

1. Организовать ввод массива 12, 0, 46, -35, 18, 46, 15, 46, -17, 0 с клавиатуры.
2. В заданном числовом массиве определить индексы элементов, имеющих значение 46.
3. В заданном числовом массиве определить произведение элементов с четными индексами.
4. В заданном числовом массиве каждый элемент, начиная со второго до предпоследнего, заменить на сумму соседних с ним элементов.

Вариант 29

1. Организовать ввод массива 69, 0, -26, 40, 3, -17, -84, -9, 82, 5 с клавиатуры.
2. В заданном числовом массиве определить среднее арифметическое положительных элементов.

3. В заданном числовом массиве каждый четный элемент заменить на нуль.
4. Изменить заданный числовой массив так, чтобы элементы были расположены в нем в обратном порядке.

Вариант 30

1. Организовать ввод массива -27, 0, -71, 57, 90, -27, 64, 31, -54, 73 с клавиатуры.
2. В заданном числовом массиве определить среднее геометрическое отрицательных элементов.
3. В заданном числовом массиве удвоить элементы с нечетными индексами.
4. В заданном числовом массиве переставить местами соседние элементы с четными и нечетными индексами.