

Московский государственный университет

имени М. В. Ломоносова

Факультет вычислительной математики и кибернетики

Е. А. Бордаченкова

**Задачи и упражнения
по языку ассемблера MASM**

(Учебное пособие)

Москва, 2023

УДК 004.42(075.8)

ББК 32.973-018я73

Б82

*Печатается по решению Редакционно-издательского совета
факультета вычислительной математики и кибернетики
МГУ имени М.В.Ломоносова*

Рецензенты:

В.Г.Баула – доцент кафедры АСВК ВМК МГУ

Е.А.Кузьменкова – доцент кафедры СП ВМК МГУ

Бордаченкова, Елена Анатольевна

Б82 Задачи и упражнения по языку ассемблера MASM: Учебное пособие для студентов 1 курса/Е.А. Бордаченкова. – Москва: Издательский отдел факультета ВМиК МГУ имени М.В.Ломоносова (лицензия ИД №05899 от 24.09.2001 г.); МАКС Пресс, 2020. – 92 с.

ISBN 978-5-89407-617-1 (ВМК МГУ имени М.В. Ломоносова)

ISBN 978-5-317-06539-3 (МАКС Пресс)

Пособие содержит задачи и упражнения по языку ассемблера MASM 6.14, необходимый для решения теоретический материал и ответы к некоторым задачам. Может быть использовано на семинарских занятиях по курсу «Архитектура ЭВМ и язык ассемблера».

Ключевые слова: обучение программированию, ассемблер

УДК 004.42(075.8)

ББК 32.973-018я73

Учебное издание

БОРДАЧЕНКОВА Елена Анатольевна

ЗАДАЧИ И УПРАЖНЕНИЯ ПО ЯЗЫКУ АССЕМБЛЕРА MASM

Учебное пособие для студентов 1 курса

Издательский отдел факультета ВМК МГУ имени М.В.Ломоносова

Лицензия ИД №05899 от 24.09.2001 г.

119992, ГСП-2, Москва, Ленинские горы, МГУ имени М.В.Ломоносова,
2-й учебный корпус

Издательство МАКС Пресс

Главный редактор: Е.М. Бугачева

Напечатано с готового оригинал-макета Подписано в печать 25.12.2020

Формат 60x90 1/16. Усл.печ.л.5,75. Тираж 300 экз. Заказ 211

Издательство ООО “МАКС Пресс”. Лицензия ИД №00510 от 01.12.99 г.

119992, ГСП-2, Москва, Ленинские горы,

МГУ имени М.В.Ломоносова, 2-й учебный корпус, 527 к.

Тел. 8(495)939-38-90/91 Тел./Факс 939-38-91

© Издательский отдел факультета ВМК МГУ
имени М.В.Ломоносова, 2020

© Бордаченкова Е.А., 2020

©Оформление ООО “МАКС Пресс”, 2020

ISBN 978-5-89407-617-1

ISBN 978-5-317-06539-3

Содержание

Предисловие	4
Введение	5
1. Директивы определения данных	8
2. Команды пересылок. Оператор ptr	15
3. Арифметические команды	19
4. Команды ввода и вывода. Структура программы	26
5. Команды перехода	30
6. Массивы	38
7. Структуры	45
8. Команды работы с битами	48
9. Записи	54
10. Стек	57
11. Процедуры	60
12. Строковые команды	66
13. Макросредства	72
Ответы	82
Приложение А. Самостоятельные работы	98
Приложение Б. Список директив, операторов, мнемокодов . . .	101

Предисловие

Книга содержит задачи и упражнения по языку ассемблера MASM 6.14, сгруппированные по тематическим разделам. Порядок разделов соответствует порядку изучения материала в курсе «Архитектура ЭВМ и язык ассемблера» на втором и третьем потоках первого курса факультета ВМК МГУ. Задачник предназначен для использования на семинарских занятиях, которые проводятся в поддержку лекционного курса.

Из всех возможностей языка ассемблера MASM 6.14 было выбрано подмножество средств, наиболее существенных для курса «Архитектура ЭВМ и язык ассемблера». Знания этих средств достаточно для освоения курса.

Каждый раздел сопровождается кратким изложением теоретического материала, необходимого для решения задач этого раздела. Более полное описание можно найти в лекциях и в электронном пособии В. Г. Баулы «Введение в архитектуру ЭВМ и системы программирования» на сайте arch32.cs.msu.ru. В конце задачника даны ответы и решения некоторых задач. Номера задач, для которых есть ответы, заключены в рамку. В Приложении приведены три самостоятельные работы. *Если в условии задачи сказано **дано**, предполагается, что **данные вводятся с клавиатуры**.*

Большая часть задач навеяна пособием Владимира Николаевича Пильщикова «Упражнения по языку MASM». Несколько лет назад планировалось совместно написать задачник, дополнив книгу В. Н. Пильщикова задачами и ответами. Однако по разным причинам проект не состоялся. Автор, пользуясь возможностью, выражает искреннюю признательность Владимиру Николаевичу Пильщикову.

Автор выражает благодарность сотрудникам факультета ВМК МГУ В. Г. Бауле, И. В. Горячей, Т. Ю. Грациановой, В. Б. Захарову, А. П. Капитоновой, Т. К. Матвеевой, М. Д. Новикову за замечания и полезные советы. Особую благодарность автор выражает А. А. Панфёрову за внимательное чтение задачника, рекомендации по тексту и большую помощь при подготовке пособия к печати.

В электронной версии задачника были исправлены замеченные опечатки печатного издания.

Введение

Язык ассемблер – машиннозависимый язык, в котором используются символические обозначения (имена). Слово «машиннозависимый» значит, что язык ассемблер строится на основе машинного языка процессора. В ассемблере вместо числовых кодов машинных операций используются названия команд – мнемокоды; вместо адресов ячеек памяти (операндов команд) используются имена переменных и имена меток.

Ассемблер MASM 6.14 основан на машинном языке процессора Intel (архитектура IA-32). Архитектура IA-32 похожа на архитектуру учебных машин – переменный формат команд, большинство команд двухадресные, наличие регистров, использование модификации адресов. При изучении ассемблера MASM нам необходимо помнить следующие особенности системы команд процессора Intel

1. Два операнда из памяти не используются ни в одной машинной команде (за исключением строковых команд и команд работы со стеком).
2. Если в машинной команде два операнда, у них должен быть одинаковый размер (байты, слова или двойные слова).

Программа, написанная на языке ассемблера, в конечном итоге преобразуется в машинную программу, находящуюся в оперативной памяти компьютера. Машинную программу будет выполнять процессор примерно так, как описывалось при изучении учебных машин. В целом, процесс выглядит так. Сначала текст программы на ассемблере обрабатывает транслятор (который также называется ассемблером), в результате получается заготовка машинной программы – объектный код. На втором шаге работает компоновщик, который объединяет объектный код нашей программы с объектными кодами других программ, необходимых для работы исходной программы (например, в таких вспомогательных программах могут быть описаны процедуры ввода-вывода). В итоге получается исполняемая программа. Наконец, исполняемая программа загружается в оперативную память, происходит окончательная настройка машинной программы, и в процессор передаётся адрес

команды, с которой нужно начать выполнение программы. Подробно этот процесс будет разбираться в конце курса лекций, при обсуждении темы «Элементы систем программирования». Сейчас нам важно понимать, что ассемблерная программа является неким довольно близким образом машинной программы и, конечно, помнить, как устроены машинные программы (это материал темы «Учебные машины»).

При изучении языка MASM необходимо учитывать, что синтаксис конструкций языка MASM и их семантика основаны на принципах работы ассемблера (транслятора).

Программа на языке MASM состоит из предложений, каждое предложение записывается в отдельной строке. Существует три типа предложений: комментарии, директивы и команды. В зависимости от типа предложения ассемблер обрабатывает предложения по-разному. **Комментарии** ассемблер пропускает. **Директивы** ассемблер выполняет. Для этого внутри ассемблера имеются специальные процедуры. Прочитав директиву, ассемблер запускает процедуру, соответствующую директиве. Есть директивы, которые только сообщают ассемблеру некоторую информацию; есть директивы, приводящие к заполнению байтов объектного кода. Для запоминания информации в ассемблере есть внутренние таблицы и специальные переменные. Семантику директив (как ассемблер обрабатывает директиву) будем разбирать по мере изучения языка MASM. **Команды** ассемблер транслирует в машинные команды и помещает в объектный код. При трансляции команд ассемблер по мнемокоду и по указанным в команде операндам выбирает код машинной операции. Затем транслируются операнды: имена регистров заменяются их номерами, имена переменных и метки заменяются соответствующими адресами, константы переводятся в машинное представление и помещаются в машинную команду. Ассемблер читает текст исходной программы на MASM'е сверху вниз, предложение за предложением, и обрабатывает каждое предложение, в итоге генерируется объектный код.

Подобно тому, как программы для учебных машин состоят из блока, содержащего команды, и блока с переменными, программы на MASM'е состоят из сегмента (секции) кода и сегмента (секции) данных. Адреса переменных отсчитываются от начала сегмента данных, самая первая переменная имеет адрес 0. Аналогично, адреса команд отсчи-

тываются от начала сегмента кода, первая команда имеет адрес 0. Адреса, отсчитанные от начала сегмента будем называть *смещениями*. Для вычисления смещений ассемблер использует переменную счётчик размещения.

Среди таблиц транслятора-ассемблера нас будет интересовать *таблица имён* (ТИ). Обрабатывая текст программы, ассемблер заносит в ТИ информацию об именах, описанных в программе. В то же время, при трансляции команд и при выполнении директив ассемблер использует информацию, хранящуюся в ТИ. Занося имя в ТИ, ассемблер записывает тип имени (константа, переменная, метка). Для переменных указывается тип (байт, слово, двойное слово, четверное слово), значение имени (смещение) и если переменная – это массив, то указывается количество элементов в нём. Для имён констант указывается значение константы. Если имя является меткой, в ТИ записывается смещение метки. Подчеркнём, что значением имени переменной (и метки) является смещение, а не содержимое переменной, хранящееся в байтах памяти. Дело в том, что при трансляции команды программы на MASM'е в машинную команду ассемблер заменяет имя операнда на значение, указанное в ТИ, а переменная в машинных командах отображается в виде адреса соответствующей ячейки.

1. Директивы определения данных

При описании синтаксиса конструкций используются обозначения:
КВ – константное выражение, АВ – адресное выражение.

курсив – понятия языка (нетерминальные символы)

ТЕКСТ, текст – названия команд, директив, операторов и т. п. (терминальные символы)

{}, [], | – метасимволы РБНФ

Внимание: В именах, описанных программистом, большие и маленькие буквы различаются!

1. Описание констант

- 1) имя = константное выражение (без ссылок вперёд)
 - 2) имя EQU константное выражение
имя EQU <текст>

Семантика. При выполнении директив "равно" и EQU ассемблер запоминает (заносит в таблицу имён) информацию о том, что описываемое имя является именем константы и значение этой константы.

2. Определение данных

Размер операнда должен соответствовать виду директивы. Адресное выражение не может быть операндом у директив DB и DW.

Особый случай: Директива DB используется для размещения строк символов, поэтому допускает операнд-строку. Следующие директивы эквивалентны

Семантика. 1) Если указано имя переменной, ассемблер заносит в таблицу имён имя переменной, её тип (определяется *DEF*) и адрес. 2) В соответствии с операндами директивы ассемблер заполняет байты объектного кода. Если используется конструкция DUP, байты дублируются столько раз, сколько задаёт множитель перед DUP.

3. Константные и адресные выражения

Выражения вычисляются в 32 разрядах. Значением константного выражения является константа (число). Значение адресного выражения — адрес (смещение в сегменте данных).

Константные выражения строятся из чисел, констант, операторов и круглых скобок.

Операторы (по убыванию приоритета)

унарные: +, -

бинарные:

мультипликативные *, /, mod, shl, shr

аддитивные +, -

Исключение: выражение *переменная1 – переменная2* является константным.

Адресные выражения:

переменная; *переменная + KB;* *переменная – KB;* *KB + переменная.*

В задачах данного раздела слово «адрес» используется в смысле «смещение в сегменте данных».

1.1 Для каждой из указанных директив выписать эквивалентную ей директиву, указав начальное значение переменной в 16-ричном виде.

- | | | | |
|-------------|------------|--------------|-----------------|
| a) A DB 17 | ж) J DB 1 | н) N DB 255 | у) U DW 256 |
| б) B DB -17 | з) Z DW 1 | о) O DW 255 | ф) F DW -256 |
| в) V DW 17 | и) I DD 1 | п) P DB 128 | х) H DW 65535 |
| г) G DW -17 | к) K DB -1 | р) R DB -128 | ц) TS DW 65525 |
| д) D DD 17 | л) L DW -1 | с) S DW 128 | ч) CC DD -65536 |
| е) E DD -17 | м) M DD -1 | т) T DW -128 | ш) SH DQ -65536 |

1.2 Для каждой из указанных директив выписать две эквивалентные ей директивы, в первой из которых начальное значение переменной записано в виде десятичного числа со знаком, а во второй — без знака.

- | | | |
|----------------|---|--|
| а) A DB 0Ah | д) D DW 127056q | и) I DW 0FFEh |
| б) B DB 0A0h | е) E DB 307o | к) K DD 0FFFF FFFF0h
(только число со знаком) |
| в) V DW 0A0h | ж) J DB 0B0h | л) L DB 80h |
| г) G DD 01101b | з) Z DD 0FFFF FFFFh
(только число со знаком) | м) M DW 8000h |

1.3 а) Описать переменные minb, maxb, minw, maxw, mind, maxd — байты, слова и двойные слова соответственно, записав в них наименьшее и наибольшее число без знака, допустимые для переменной соответствующего размера.

б) Выполнить то же упражнение для чисел со знаком.

Начальные значения указывать в виде 16-ричных чисел.

1.4 Есть фрагмент листинга

<i>адрес</i>	<i>объектный код</i>	<i>текст программы</i>
00000000	80	A DB 128
00000001	80	DB -128
00000002	0080	B DW 128
00000004	FF80	DW -128
00000006	00000002	C DD B-A
0000000A	0000	DW 0
0000000C	00000000	DD 0
00000010	00000010 R	D DD D

- 1) Какие имена (переменные) описаны в программе?
 - 2) Указать адрес каждой переменной и её тип.
 - 3) Существует ли переменная, соответствующая двойному слову с адресом 0Ch? Выписать адресное выражение для доступа к этому двойному слову.

1.5 Построить листинг для фрагмента программы, если адрес А равен 0.

- A DB 10h
 - B DB 20h
 - C DD B, C-A

1.6 A DW 1020h

- B DD 10203040h, 0h

Указать в 16-ричном виде значения

1.7 A DB 25

- B DW 25
X DD 25
C DD B, C-A

Нарисовать байты, указав их адреса, и заполнить эти байты так, как будут заполнены байты по указанным директивам, если адрес А равен 0.

1.8 По данному фрагменту листинга восстановить пропущенный текст в программе (две строки с минусами)

```
00000000 0001 0002 X DW 1, 2
00000004 03 07 -----
00000006 00000008 -----
0000000A 04 Y DB Y-B
```

1.9 Выписать таблицу имён и объектный код, которые построит ассемблер при трансляции следующего фрагмента программы. Считать, что адрес переменной a = 0.

a DW 123h
x DB 0A1h, 3h, 4h
y DD x+1

1.10 Описать переменную-двойное слово X, которая содержит

- а) адрес самой переменной;
- б) адрес байта памяти, следующего за первым байтом X;
- в) адрес байта памяти перед первым байтом X;
- г) адрес байта, следующего за переменной X;
- д) адрес слова памяти перед первым байтом X;
- е) адрес старшего (второго) слова переменной X;
- ж) адрес двойного слова, следующего за X;
- з) адрес двойного слова, предшествующего X

1.11 Описать байтовый массив PRIM из 7 элементов, начальными значениями которых являются первые семь простых чисел (2, 3, 5 и т. д.).

1.12 Выписать все возможные варианты (кроме тех, где указываются коды букв) описания символьного массива S, начальными значениями элементов которого являются первые три большие буквы английского алфавита.

1.13 Пусть текущее значение счётчика размещения равно 4. Какое значение будет у счётчика размещения после выполнения следующих директив?

t1 DB 'A', 'B'
DB 'CD'
t2 DD 'E'

1.14 Изобразить, как будут заполнены байты памяти по директивам

V1 DB 'CDE'
V2 DW 'CD'
V3 DD 'CDE'

Код символа 'C' равен 43h.

1.15 Изобразить, как будут заполнены байты при трансляции следующего фрагмента (Код символа 'A' равен 41h)

```
S DB 'ABC'  
DB 'A', 'B'  
X DW 'AB'  
Y DW 'A', 'B'  
Z DD 'ABC'
```

1.16 Выписать таблицу имён и изобразить, как будут заполнены байты при трансляции следующего фрагмента. Считать, что адрес V равен 0.

```
N = 5 mod 3  
V DB N  
DB S2-X  
X DW 'AB'  
S1 DB 'ABC'  
S2 DB 0ABh  
DW S2-S1  
DD X+5
```

1.17 Объяснить ошибки в листинге

00000000 12	B DB 18
00000001 EE	C DB -18
00000002	D DB 350

1. error A2071: initializer magnitude too large for specified size

00000003 41 42 43 44	E DB 'ABCD'
00000007	F DW 'ABC'

2. error A2071: initializer magnitude too large for specified size

00000009 41424344	G DD 'ABCD'
0000000D	H DB B

3. error A2071: initializer magnitude too large for specified size

E DW 0

4. error A2005: symbol redefinition : E

Какое шестнадцатиричное число содержится в байте с адресом 9?

1.18 N EQU 10
X DB 'ABCDE'
Y DW 1234h
V DD 0ABCDEFh

Для каждого выражения определить, является ли оно константным, адресным или неправильным. Для константных выражений определить значение, для адресных указать тип и вычислить смещение (считать, что адрес X равен 0), для неправильных выражений объяснить ошибку.

- | | | | |
|--------|------------|--------------------|--------------|
| а) N | е) N-Y | л) (V-Y)/2 | р) V-X+V |
| б) V | ж) 2*N+1 | м) X-Y/2 | с) 'A'+N/3 |
| в) Y-1 | з) 2*X+1 | н) N shr 2 | т) 'Z'-'Y' |
| г) Y-X | и) V+Y-V | о) N shr 1 | у) '9'-'8' |
| д) Y+N | к) V+(Y-V) | п) 0FFFF FFFFh+2+Y | ф) X+'B'-'A' |

1.19 Выписать директивы, заполняющие разряды 8 байтов следующим образом

- | | | | |
|-------------|-------------|-------------|-------------|
| а) 10000000 | б) 00000001 | в) 10000001 | г) 00011000 |
| 11000000 | 00000011 | 11000011 | 00111100 |
| 11100000 | 00000111 | 11100111 | 01111110 |
| 11110000 | 00001111 | 11111111 | 11111111 |
| 11111000 | 00011111 | 11111111 | 11111111 |
| 11111100 | 00111111 | 11100111 | 01111110 |
| 11111110 | 01111111 | 11000011 | 00111100 |
| 11111111 | 11111111 | 10000001 | 00011000 |

Решить задачу, используя директивы 1) DB; 2) DW; 3) DD. Двоичную систему не использовать.

1.20 Определить с помощью синтаксической диаграммы понятие *директива определения данных*, считая, что описаны понятия *константное выражение* и *адресное выражение*. Не проверять соответствие размера значений выражений типу директивы.

1.21 Описать массив X из 85 элементов-двойных слов со следующими начальными значениями:

- все начальные значения равны 0;
- все начальные значения равны 2023;
- первые 40 элементов имеют значение 10, следующие 20 элементов — значение '*', остальные — без начального значения;
- средний элемент имеет значение 1, все остальные — значение 0.

1.22 M EQU 30
N EQU 50

Описать байтовую матрицу A размера M × N, в каждой строке которой первые 10 элементов имеют значение 0, а остальные — значение -1.

1.23 N EQU 30

Описать байтовую единичную матрицу E размера N × N.

1.24 Z1 DQ 1FFFF FFFFh

Z2 DQ 0FFFF FFFFh shl 1

Изобразить, как будут заполнены байты памяти по указанным директивам.

1.25 При трансляции директивы

X DW 0FFFFh + 1

ассемблер зафиксирует ошибку; при трансляции директивы

Y DW 0FFFF FFFFh + 1

ошибки не будет. Почему?

1.26 Какое значение будет записано в переменную в результате выполнения директивы

X DQ 0FFFF FFFFh + 1

1.27 N=...; N>8

Разместить в памяти N единиц подряд. Например, для N = 10 нужно получить два байта

1111 1111
1100 0000

Для N = 24 потребуется описать три байта

1111 1111
1111 1111
1111 1111

2.3 B DB 'a'

W DW 2

D DD 1

0 DO 3

Среди перечисленных команд указать неправильные; объяснить, в чём ошибка.

- | | | |
|------------------|-----------------|----------------------------|
| a) MOV B, AX | и) MOV B, W-B | c) MOV B, byte ptr ECX |
| б) MOV W, AX | к) XCHG D, ECX | т) MOVSX EAX, B |
| в) MOV AX, W | л) MOV D, ESI+1 | у) MOVZX EAX, AL |
| г) XCHG ESI, B-1 | м) XCHG B, 125 | ф) MOVSX W, CL |
| д) MOV EAX, BL | н) MOV Q, 0 | х) MOVSX ESI, D |
| е) MOV B, 80h | о) XCHG B, B+1 | ц) MOV word ptr Q, 12 |
| ж) XCHG AH, AL | п) MOV EAX, W-B | ч) MOV Q, word ptr 2 |
| з) MOV SI, W+1 | р) MOV W, W+2 | ш) MOV dword ptr B, 12345h |

Изобразить содержимое байтов памяти после выполнения команды пункта (4).

2.4 A DB 12h

B DD 345678h

C DB 0ABh

Указать (в 16-ричном виде), какие значения получат регистры в результате выполнения последовательности команд

MOV AL, byte ptr B+1

MOV EBX, dword ptr A

MOVZX ECX, C

MOVZX EDX, C

2.5 A DB ?

B DW ?

C DD ?

D DO ?

Трактуя содержимое переменных A, B, C и D как числа без знака, записать в эти переменные

- а) наименьшие б) наибольшие
возможные числа, соответствующие размерам переменных

2.6 A DB ?

B DW ?

C DD ?

D DQ ?

Трактуя содержимое переменных A, B, C и D как числа со знаком, запи-

сать в эти переменные

2.7 A DB 3 DUP (?) ; массив a₁,a₂,a₃

Рассматривая переменную A как массив A[1..3], реализовать следующее действие

- а) Присвоить $A[1]:=1$, $A[2]:=2$, $A[3]:=3$. Решить задачу двумя способами: за три команды; за две команды.

б) Циклически сдвинуть на 1 позицию влево элементы массива A.

в) Переставить в обратном порядке элементы массива A.

2.8 A1 DB 100

A2	DB	20
X1	DW	1000
X2	DW	7301
V1	DD	345
V2	DD	206

Реализовать следующие операторы

- а) A1:=A2;
 - б) X1:=X2;
 - в) V1:=V2;
 - г) поменять местами содержимое переменных A1 и A2 ($A1 \leftrightarrow A2$)
 - д) поменять местами содержимое переменных V1 и V2 ($V1 \leftrightarrow V2$)

2.9 X DQ 123456h

Y DQ 789ABCh

- a) Записать в переменную Y значение переменной X.
 - б) Поменять местами содержимое переменных X и Y ($X \leftrightarrow Y$)

2.10 Z DW ?

Поменять местами байты слова z.

2.11 V DD ?

Переставить в обратном порядке байты переменной v.

2.12 Q DQ ?

Переменной Q присвоить значение а) 125; б) -125. Вспомогательные переменные и регистры не использовать.

2.13 A DW -19

...

MOVZX EAX, A
MOVSX EBX, A

Указать в 16-ричном виде значения регистров EAX и EBX.

2.14 A DB ? ; число без знака

Y DW ?

Записать в переменную Y значение переменной A.

2.15 X DB ? ; число со знаком

Y DD ?

Записать в переменную Y значение переменной X.

2.16 A DB ? ; число без знака

Z DD ?

Выписать последовательность команд, которые записывают в переменную Z значение переменной A. Решить задачу

- a) с использованием команды MOVZX;
- b) не используя команду MOVZX.

2.17 A DW -73

B DW ?

Указать в 16-ричном виде значение переменной B после выполнения следующих команд:

MOV AX, A
MOV AH, 0
MOV B, AX

2.18 A DW -73

B DW ?

Указать в 16-ричном виде значение переменной B после выполнения следующих команд:

MOV AX,A
MOV byte ptr B, AH
MOV byte ptr B+1, AL

3. Арифметические команды

Команды сложения и вычитания

```
ADD op1,op2 ; op1 := op1+op2
ADC op1,op2 ; op1 := op1+op2+CF
SUB op1,op2 ; op1 := op1-op2
SBB op1,op2 ; op1 := op1-op2-CF
NEG op1 ; op1 := 0-op1
```

Устанавливаются арифметические флаги CF, OF, SF, ZF.

```
INC op ; op := op+1
DEC op ; op := op-1
```

Команды устанавливают флаги OF, SF, ZF.

Команды умножения и деления

```
MUL op ; S := S1*op, числа без знака
IMUL op ; S := S1*op, числа со знаком
DIV op ; S1 := S div op, S2 := S mod op, числа без знака
IDIV op ; S1 := S div op, S2 := S mod op, числа со знаком
```

В зависимости от размера операнда команды обозначения S, S1 и S2 соответствуют следующим регистрам

op	S	S1	S2
byte	AX	AL	AH
word	DX:AX	AX	DX
dword	EDX:EAX	EAX	EDX

Требования на operand: op не может быть непосредственным.

Команды знакового расширения

```
CBW ; AL → AX, число со знаком
CWD ; AX → DX:AX, число со знаком
CDQ ; EAX → EDX:EAX, число со знаком
```

3.1 Объяснить ошибки (какие правила записи команд нарушены) во фрагменте листинга

```
; переменные
00000000 12          B      DB 18
00000001 EE          C      DB -18
00000002 0000015E     D      DD 350
```

```
; команды
MOV EAX, A
```

ошибка 1: undefined symbol : A
ADD EAX, B
ошибка 2: invalid instruction operands

00000000 B1 00 MOV CL, 0
 ADD CL, 532

ошибка 3: *invalid instruction operands*
 MOVZX D, B

ошибка 4: *memory operand not allowed in context*
 SUB 0, B

ошибка 5: *immediate operand not allowed*
 XCHG B, C

ошибка 6: *invalid instruction operands*

3.2 Указать значения регистров AH и AL (в виде десятичных чисел без знака) и флагов CF и ZF после выполнения следующих команд:

- | | | | |
|-------------|-------------|---------------|--------------|
| a) MOV AH,0 | б) MOV AH,0 | в) MOV AH,255 | г) MOV AH,20 |
| MOV AL,160 | MOV AL,160 | MOV AL,255 | MOV AL,10 |
| ADD AL,60 | ADD AL,160 | ADD AL,1 | SUB AL,16 |
| ADC AH,3 | ADC AH,3 | ADC AH,0 | SBB AH,0 |

3.3 Указать значения регистра BH (в виде десятичного числа со знаком) и флагов OF и SF после выполнения следующей пары команд:

- | | | | |
|--------------|---------------|---------------|---------------|
| a) MOV BH,80 | в) MOV BH,-80 | д) MOV BH,-80 | ж) MOV BH,80 |
| ADD BH,40 | ADD BH,-40 | ADD BH,40 | SUB BH,100 |
| б) MOV BH,80 | г) MOV BH,-80 | е) MOV BH,80 | з) MOV BH,-80 |
| ADD BH,50 | ADD BH,-50 | ADD BH,-40 | SUB BH,50 |

3.4 Указать значения регистра CL (в виде как знакового, так и беззнакового десятичных чисел) и флагов CF, OF, SF и ZF после выполнения следующей пары команд:

- | | | | |
|---------------|---------------|---------------|--------------|
| a) MOV CL,128 | б) MOV CL,-10 | в) MOV CL,246 | г) MOV CL,40 |
| ADD CL,128 | ADD CL,-40 | ADD CL,216 | SUB CL,100 |

3.5 Указать исходное значение регистра AL (любое из возможных), при котором после выполнения команды ADD AL,2 флаги имели бы следующие значения:

- а) CF=1, OF=0, SF=0 б) CF=0, OF=1, SF=1

3.6 Указать все возможные исходные значение регистра CL, при которых после выполнения команды SUB CL,25 флаги имели бы следующие значения:

3.7 Доказать следующие утверждения:

- a) Если операнды команды ADD интерпретировать как знаковые числа, то флаг OF получает в этой команде значение 1 тогда и только тогда, когда эти числа имеют один и тот же знак, а у результата команды — иной знак.
 - б) Если операнды команды SUB x,y интерпретируются как знаковые числа и $x < y$, то после выполнения этой команды флаги OF и SF обязательно будут иметь разные значения ($OF \neq SF$).

3.8 X DB 7

DB 3

Требуется записать в регистр CL значение переменной X, увеличенное на 1. Определить, какой из следующих двух фрагментов правильно решает эту задачу. Указать значение CL после выполнения каждого из фрагментов:

- a) MOV CL, X
ADD CL, 1

3.9 A EOU 7

X DB ? ; $0 \leq X \leq 80$

Реализовать наименьшим числом команд следующее присваивание:

- a) X:=3*X

3.10 X DW ? ; $0 \leq X \leq 10000$

Не используя команды умножения и стараясь минимизировать количество обращений к ОП, реализовать присваивание:

- a) $X := 4 * X$ b) $X := 5 * X$

3.11 N DB ? ; $0 \leq N \leq 9$

D DB ? ; '0'≤D≤'9'

- а) Записать в D символ-цифру, соответствующую числу N;
б) Записать в N число, соответствующее символу-цифре D.

3.12 X DQ ?

Y DQ ?

Z DQ ?

Реализовать следующие операции над «длинными» числами:

a) Z:=X+Y

б) Z:=X-Y

в) Z:=-Y (реализовать как вычитание 0-Y)

3.13 N DB ?, ?, ?

M DB ?, ?, ?

Пусть N и M — трёхбайтовые числа, используется традиционное представление: младшие разряды расположены по младшим адресам. Реализовать следующие операции над этими числами:

a) N:=N+1

б) N:=N-1

в) M:=M+N

3.14 A DB ?

B DW ?

C DD ?

Для каждой из команд, перечисленных ниже

1) MUL A

3) MUL C

5) IMUL AX

7) IMUL ESI

2) MUL B

4) MUL CL

6) MUL DX

8) MUL AH

ответить на вопросы:

а) что является первым операндом умножения?

б) где находится произведение?

3.15 A DB ?

B DW ?

C DD ?

Для каждой из команд, перечисленных ниже

1) DIV A

3) IDIV C

5) IDIV AX

7) IDIV ESI

2) IDIV B

4) DIV CL

6) DIV DX

8) DIV EDX

ответить на вопросы:

а) что является делимым?

б) куда записывается частное и куда — остаток деления?

3.16 Найти и объяснить ошибки в следующих фрагментах программ

- а)** Требуется записать в CL и CH — младшие цифры чисел X и Y соответственно

X DW 245

Y DW 24521

```
MOV AX, X
MOV BL, 10
DIV BL
MOV CL, AH
MOV AX, Y
MOV BL, 10
DIV BL
MOV CH, AH
```

- б)** Требуется записать в CL — последнюю (младшую) и в CH — предпоследнюю цифру числа Z. (т. е. CL=1, CH=2)

Z DD 54321

```
MOV EBX, 10
MOV EAX, Z
MOV EDX, 0
DIV EBX
MOV CL, DL
DIV EBX
MOV CH, DL
```

3.17 Указать (в 16-ричном и в 10-чном виде) значение регистра AX после выполнения следующих команд:

- | | | | |
|----------------|----------------|----------------|----------------|
| a) MOV AL, 80h | b) MOV AL, 80h | c) MOV AL, 7Fh | d) MOV AL, 80h |
| MOV BL, 2 | MOV BL, 2 | CBW | CBW |
| MUL BL | IMUL BL | | |
| d) MOV AL, 82h | e) MOV AL, 80h | | |
| CBW | MOV AH, 0 | | |
| MOV BL, 2 | MOV BL, 2 | | |
| IDIV BL | IDIV BL | | |

3.18 Реализовать действие команды CBW другими командами ассемблера.**3.19** Расширить байт AL до 64-битного числа EDX:EAX

- а) как число без знака;
б) как число со знаком.

3.20 A DB ?

B DW ?

C DD ?

Считая, что переменные содержат: 1) числа без знака; 2) числа со знаком, выписать команды, реализующие присваивания:

- | | | | |
|-------------|---------------|-----------------|-----------------|
| a) AX := A | b) DX:AX := B | d) EAX := B | j) EDX:EAX := B |
| б) EAX := A | г) DX:AX := C | е) EDX:EAX := C | з) EDX:EAX := A |

3.21 A DD ? ; число со знаком

B DD ?

Вычислить: $B = (A \text{ div } 1000) * (A \text{ mod } 1000)$

3.22 N DB ? ; число без знака

K DW ?

Вычислить: $K = (N * (N + 1)) \text{ div } 2$

3.23 N DW ? ; $0 \leq N \leq 999$

Записать в байтовые переменные D1, D2, D3 десятичные цифры числа $N = (d_1 d_2 d_3)_{10}$.

3.24 Пусть A, B, C и X — байтовые переменные, числа без знака, а Y — переменная типа dword. Вычислить: $Y = A * X^2 + B * X + C$

3.25 Решить предыдущую задачу в предположении, что переменные — числа со знаком.

3.26 N DW ? ; $1 \leq N \leq 365$

WD DB ?

Записать в WD номер дня недели (1 — понедельник, ..., 7 — воскресенье), на который приходится N-й день года, считая, что 1 января этого года — понедельник.

3.27 Пусть H, M и S — байтовые переменные, а T — переменная типа dword. Считая, что от начала суток прошло H часов ($0 \leq H < 24$), M минут и S секунд ($0 \leq M, S < 60$), определить, сколько всего секунд прошло от начала суток к этому моменту времени. Ответ записать в T.

3.28 Пусть T — переменная размером в двойное слово, а H, M и S — байтовые переменные. Считая, что прошло T секунд ($0 \leq T < 86400$) от начала суток, определить, сколько полных часов (H), минут (M) и секунд (S) прошло к этому моменту времени.

3.29 Объяснить ошибки (какие правила записи команд нарушены) в листинге

```
= 0000000A           base = 10
; переменные
00000000 18          A      DB 24
00000001 87          B      DB 135
; команды
00000000 A0 00000000 R      MOV AL, A
                           MUL base
ошибка 1: immediate operand not allowed
                           MUL AL, 10
ошибка 2: syntax error : ,
                           INC B-A
ошибка 3: immediate operand not allowed
                           CBW A
ошибка 4: syntax error : A
00000005 B8 000004D2      MOV EAX, 1234
                           IDIV 100
ошибка 5: immediate operand not allowed
```

3.30 Пусть D1, D2, D3 — байтовые переменные, N — переменная типа слово. Считая, что значения D1, D2 и D3 — это символы-цифры (от '0' до '9'), записать в N число, десятичная запись которого составлена из этих цифр ($N=D1*100+D2*10+D3$).

3.31 A DB 12h

B DW 3456h

Среди перечисленных ниже команд указать те, что записаны неверно; ошибки объяснить.

- | | | | |
|----------------|-------------|----------------|---------------------|
| a) ADD BX, '*' | b) INC B-A | c) ADC BH,BL | d) SBB AX,BL |
| d) ADD BX,B | e) SUB CX,A | j) ADC B-1,'B' | z) SUB A,byte ptr B |
| и) CBW AL | k) MUL AL | л) DIV 100 | m) DEC A+1 |

Изобразить содержимое байтов памяти после выполнения команды пункта (m).

4. Команды ввода и вывода. Структура программы

Используется оператор

offset имя переменной

Это константное выражение; значение равно адресу переменной в сегменте (секции) данных.

Команды

```
INCHAR op ; ввод символа, op - r8 или m8
ININT op ; ввод числа, op - r32 или m32
OUTCHAR op ; печать символа, op - r8, m8 или i8
OUTI op ; печать числа со знаком, op - r32, m32 или i32
OUTU op ; печать числа без знака, op - r32, m32 или i32
OUTSTR op ; печать строки от байта с адресом op до байта = 0h
            op - r32 или offset имя строки
NEWLINE ; writeln
EXIT ; завершение программы
```

Названия команд можно писать либо большими, либо маленькими буквами.
В операндах нельзя использовать регистр ESP.

Структура программы

```
INCLUDE settings.inc
INCLUDE io2020.inc
.STACK 4096

; константы
.DATA
    ; описание переменных

.CODE
start:
    ; команды
    EXIT
END start
```

4.1 Пусть в регистре BH находится код какой-то большой латинской буквы и требуется записать в этот регистр код одноименной малой латинской буквы. Определить, какой из следующих фрагментов правильно решает эту задачу:

- a) SUB BH, 'A' 6) ADD BH, 'a'-'A' b) MOV BH, BH-'A'+'a'
 ADD BH, 'a'

4.2 Пусть в регистре BH находится код какой-то цифры от '0' до '9'. Записать в этот регистр соответствующую десятичную цифру.

4.3 Пусть в регистре BH находится какая-то десятичная цифра. Записать в этот регистр соответствующий символ от '0' до '9'.

4.4 Что будет напечатано в результате работы следующих фрагментов

a) MOV AL, 200
MOVSX EAX, AL
OUTI EAX

6) MOV AL, -10
MOVZX EAX, AL
OUTU EAX

4.5 S DB 'N', 'N'+1, 'N'+2, 0

Что будет напечатано в результате выполнения команды

OUTSTR offset S

Во всех задачах до конца параграфа требуется написать полную программу, если не указано обратное.

4.6 Описать переменную X — двойное слово с начальным значением 0FFFF FFFFh. Напечатать значение X как число со знаком и как число без знака, в отдельных строках.

4.7 Описать байтовую переменную-строку S со значением 'Distant education', 0. Напечатать в отдельной строке значение переменной S.

4.8 Операнд команд печати числа — двойное слово. Как напечатать числа меньшего размера? Пусть есть описания

A DB ? ; без знака
B DB ? ; со знаком
X DW ? ; без знака
Y DW ? ; со знаком

Написать последовательность команд для выполнения действий

- а) напечатать содержимое байтовых переменных A и B;
- б) напечатать содержимое переменных-слов X и Y.

4.9 Написать программы решения следующих задач. Обратить внимание на правильность печати чисел.

- а) Описать переменную A размером в слово с начальным значением 60000. Напечатать значение этой переменной.
- б) Описать переменную B размером в слово с начальным значением -5536. Напечатать значение этой переменной.

4.10 Написать программу. Ввести число без знака N ($N \geq 0$). Вычислить и напечатать (в отдельных строках) значение выражений $N \bmod 10$, $N \bmod 10$. Считывать число в EAX, не использовать переменные для хранения чисел.

4.11 Ввести символ. Напечатать

- а) соответствующий ему код ASCII;
- б) символ, следующий за введённым в кодировке ASCII (считать, что такой символ есть);
- в) символ, предшествующий введённому в кодировке ASCII (считать, что такой символ есть).

4.12 Дан символ — маленькая латинская буква. Напечатать её номер в алфавите (считать, что у буквы 'а' номер 1).

4.13 Дано число n ($1 \leq n \leq 26$) — номер буквы английского алфавита. Напечатать соответствующую ему большую букву английского алфавита. Считать, что у буквы 'А' номер 1.

4.14 Дан символ — большая английская буква. Напечатать соответствующую маленькую английскую букву.

4.15 Даны два числа со знаком a и b . Числа принадлежат диапазону $[-32768, 32767]$. Вычислить и напечатать значение выражения
$$(a * b) \bmod 17$$

4.16 Даны два числа со знаком a и b . Вычислить и напечатать значение выражения $(a + b) \bmod 17$. Программа должна верно вычислить результат для любых исходных чисел, даже если их сумма превосходит размер двойного слова.

4.17 Дано неотрицательное число. Используя OUTCHAR, напечатать последнюю цифру в записи этого числа в 7-ричной системе.

4.18 Написать программу решения следующей задачи

- а) Напечатать наибольшее число без знака, которое можно разделить на байт 10. Напечатать также частное и остаток от деления этого числа на 10.
- б) Напечатать наибольшее по модулю число со знаком, которое можно разделить на байт 10. Напечатать также частное и остаток от деления этого числа на 10.
- в) Напечатать наибольшее число без знака, которое можно разделить на слово 10. Напечатать также частное и остаток от деления этого числа на 10.
- г) Напечатать наибольшее по модулю число со знаком, которое можно разделить на слово 10. Напечатать также частное и остаток от деления этого числа на 10.

4.19 $m = \dots ; 1 \leq m \leq 26$

А DB 'abcdefghijklmnopqrstuvwxyz', 0

Напечатать из строки А

- а) m последних букв английского алфавита;
- б) m первых букв английского алфавита.

4.20 Ввести число n , $1 \leq n \leq 26$.

а) Напечатать английский алфавит, начиная с n -й буквы. Считать, что буква 'а' имеет номер 1.

б) Напечатать n последних букв английского алфавита. Считать, что буква 'а' имеет номер 1.

4.21 Ввести число k из диапазона 1..7 — номер дня недели. Напечатать название (двухбуквенное: пн, вт, ср, чт, пт, сб, вс) k -го дня недели. Использовать команду OUTSTR.

4.22 Ввести число t из диапазона 1..12 — номер месяца в году. Напечатать название (трёхбуквенное: янв, фев, мар, апр, май, июн, июл, авг, сен, окт, ноя, дек) t -го месяца года. Использовать команду OUTSTR.

5. Команды перехода

Команды переходов не меняют флаги.

Безусловные переходы

JMP метка ; прямой переход на метку
JMP op ; op - r32, m32, косвенный переход по адресу из op

Команда сравнения

CMP op1,op2 ; op1-op2 \Rightarrow флаги CF, OF, SF, ZF

Требования на операнды те же, что и у команды SUB.

Условные переходы

Синтаксис JMP XXXX метка ; XXX задаёт условие перехода

Список мнемокодов

JE ; переход по равно
JNE ; по не равно

Переходы по остальным условиям различаются для чисел без знака и чисел со знаком.

без знака условие со знаком

JB	<	JL
JBE	\leq	JLE
JA	>	JG
JAE	\geq	JGE

Управление циклами

Внимание: Следующие команды позволяют перейти на команду, отстоящую не более чем на 128 байтов от команды перехода.

JECXZ метка ; Если ECX=0, переход на метку
LOOP метка

Действие команды LOOP: 1) ECX := ECX - 1; 2) Если ECX \neq 0, происходит переход на метку.

5.1 Для данных ниже команд определить тип перехода (прямой или косвенный), для неверных команд объяснить ошибку. Считать, что выше команды перехода были описаны имена: A — байт; В — двойное слово; K — константа со значением 4000FFFFh; L, M — метки.

- | | | | |
|----------|-----------|------------|------------|
| a) JMP L | b) JMP BX | d) JMP ECX | ж) JMP EIP |
| б) JMP A | г) JMP B | е) JMP M | з) JMP K |

5.2 X DB 206 ; (-50)

Определить, будет ли сделан переход на метку MET при выполнении следующих команд:

- | | | | |
|------------------------|------------------------|------------------------|----------------------------------|
| a) CMP X,100
JA MET | b) CMP X,100
JG MET | c) CMP X,210
JA MET | d) CMP X,210
JE LAB
JB MET |
|------------------------|------------------------|------------------------|----------------------------------|
- LAB:
- | | | |
|------------------------|------------------------|------------------------|
| d) CMP X,-40
JG MET | e) CMP X,-40
JL MET | f) CMP X,216
JL MET |
|------------------------|------------------------|------------------------|

5.3 X DW ? ; число со знаком

Определить, какие из следующих фрагментов эквивалентны условно-му оператору

if X>80 then X:=X-1 else X:=X+1

- | | | |
|---|--|---|
| a) CMP X,80
JG M
INC X
JMP L
M: DEC X
L: | b) CMP X,80
JLE M
DEC X
M: INC X
e) CMP X,80
JLE M
DEC X
M: INC X | c) CMP X,80
DEC X
JMP L
M: INC X
L: |
| d) CMP X,80
JLE M
ADD X,2
M: DEC X
M: INC X
L: | e) CMP X,80
JBE M
DEC X
JMP L
M: INC X
L: | |

5.4 Упростить (записать меньшим числом команд) следующий фрагмент:

- | | | | |
|---|---|---|--|
| a) CMP AH,0
JE M
JG M
NEG AH
M: | b) CMP AH,0
JL M1
JMP M
M1: NEG AH
M: | c) L: ADD EAX,EAX
DEC EBX
CMP EBX,0
JEM
JMP L
M: | d) L: ADD EAX,EAX
DEC ECX
CMP ECX,0
JNE L |
|---|---|---|--|

5.5 Реализовать следующие последовательности операторов (числа со знаком)

- a) if ESI>20 then AL:= 1 else AL:= 0;
 б) AL:=0;
 if ESI>20 then AL:=1

Сравнить полученные фрагменты кода.

5.6 Реализовать команды

а) JECXZ M б) LOOP M

если расстояние от команды перехода до метки M больше 128 байтов.

5.7 С DB ?

К DB ?

Проверить, является ли символ С 16-ричной цифрой. Если является, записать в К значение 1, иначе записать 0.

5.8 Пусть X, Y и Z — знаковые байтовые переменные. Реализовать наименьшим числом команд следующее действие:

- а) if ($X < 2$) and ($Y < 4$) and ($Z < 10$) then EAX:=102 else EAX:=-3;
б) if ($X < 2$) or ($Y < 4$) or ($Z < 10$) then EAX:=102 else EAX:=-3

5.9 Пусть X, Y, Z и W — знаковые переменные-слова. Реализовать следующее присваивание:

- а) $Y := \text{abs}(X)$ в) $W := \min(X, Y, Z)$
б) $Z := \min(X, Y)$ г) $W := \max(X+10, 2^*Y, 8-Z)$

Считать, что все промежуточные результаты умещаются в слово.

5.10 Пусть X — четверное слово, S — байт. Записать в S знак числа X:

$$S := \begin{cases} 1, & \text{если } X > 0 \\ 0, & \text{если } X = 0 \\ -1, & \text{если } X < 0 \end{cases}$$

5.11 Переменные x, y — двойные слова, числа со знаком. Записать в байтовую переменную k номер четверти координатной плоскости, в которой лежит точка (x, y) .**5.12** Считая, что Y — двойное слово, число без знака, определить, является ли год Y високосным

- а) реализовать полный условный оператор

```
if (Y mod 400 = 0) or (Y mod 4 = 0) and (Y mod 100 <> 0)
  then CL:=1
else CL:=0;
```

- б) реализовать последовательность операторов

```
CL:=0;
if (Y mod 400 = 0) or (Y mod 4 = 0) and (Y mod 100 <> 0)
  then CL:=1
```

5.13 Пусть X — слово, число без знака. Выписать последовательность команд, решающих задачу

- Записать в CL число 1, если X делится на 2, или на 3, или на 5, в противном случае записать в CL число 0.
- Записать в CL число 1, если X делится на 2, ($X+7$) делится на 3 и ($X+4$) — на 5, в противном случае записать в CL число 0. Результат должен быть вычислен корректно, даже если значения ($X+7$), ($X+4$) не умещаются в слово.

5.14 Реализовать следующие операторы (числа без знака)

a) ECX:=0; while EAX>=EBX do begin EAX:=EAX-EBX; ECX:=ECX+1 end ; деление EAX на EBX ; ECX – частное ; EAX – остаток	б) ECX:=0; repeat ECX:=ECX+1; EAX:=EAX div EBX until EAX=0 ; ECX – количество цифр ; в записи числа EAX ; в EBX-ичной системе
---	--

Команду DIV не использовать.

5.15 В регистре BL находится число от 0 до 15. Записать в BL символ — соответствующую 16-ричную цифру. В качестве «буквенных» цифр использовать большие латинские буквы от 'A' до 'F'.

5.16 Пусть в регистре BH находится код символа из набора {'0', ..., '9', 'A', ..., 'F'}. Записать в этот регистр значение соответствующей цифры в 16-ричной системе счисления.

5.17 X DQ ?
Y DQ ?

Реализовать условный переход на метку LESS, если $X < Y$

- переменные содержат числа без знака;
- переменные содержат числа со знаком.

Указание: реализовать команду CMP длинных чисел (т. е. вычитание $X - Y$ с установкой флагов без сохранения результата)

5.18 Пусть H, M и S — байтовые переменные, причем $0 \leq H < 24$, $0 \leq M, S < 60$. Переменные H, M и S задают текущий момент времени суток — часы, минуты и секунды соответственно. Увеличить время на 1 секунду. (Учесть смену суток.)

5.19 N DD ? ; число без знака

Записать в регистр CL:

- а) количество значащих цифр в десятичной записи числа N;
- б) сумму цифр десятичной записи числа N;
- в) количество нулей в десятичной записи числа N;
- г) количество нулей и пятёрок в десятичной записи N;
- д) наименьшую цифру из десятичной записи числа N.

5.20 Написать фрагмент решения задачи. Данна последовательность из 40 чисел со знаком.

- а) Посчитать и напечатать количество чисел последовательности, кратных 5.
- б) Посчитать и напечатать, сколько чисел оканчиваются на 0.

5.21 Пусть N — переменная-двойное слово ($N \geq 1$), а K — переменная-байт. Если N является степенью числа 3 (1, 3, 9, 27, ...), записать в K соответствующий показатель степени ($N = 3^K$), в противном случае записать в K число -1.

5.22 N DB ? ; $N \geq 1$

F DW ?

Записать в F:

- а) N-е число Фибоначчи (F_N);
- б) первое из чисел Фибоначчи, превосходящее 10000.

(Определение чисел Фибоначчи: $F_1 = F_2 = 1$, $F_k = F_{k-1} + F_{k-2}$, $k \geq 3$.)

5.23 Пусть N и K — переменные-слова и $2 < K < N$. Записать в регистр BX наибольший из остатков от деления N на числа 2, 3, ..., K.

5.24 Написать программу решения задачи. Дано число n, $n > 1$. Вычислить количество делителей числа n, включая 1 и само число.

5.25 N DD ? ; $N > 1$

Написать программу решения задачи: Определить, является ли N простым числом. Ответ напечатать.

5.26 Z DB ?

Записать в Z максимальное значение выражения $(X^2 + Y^2) \bmod 80$ в целочисленных точках квадрата $0 \leq X \leq 99$, $0 \leq Y \leq 99$.

5.27 R DB ? ; $0 < R < 150$

Записать в регистр DX количество целочисленных точек на плоскости, попадающих в круг радиуса R с центром в начале координат.

В следующих задачах требуется написать фрагмент решения задачи — необходимые директивы описания переменных и команды.

5.28 Дан текст — непустая последовательность символов, за которой следует точка. Посчитать, сколько раз в тексте встречаются прописные английские буквы ('A', 'B', ..., 'Z'). Результат записать в EAX.

5.29 Дана непустая последовательность символов из набора {'0', '1', '2', '3', '4', '5'}, за которой следует пробел. Последовательность представляет собой запись числа в 6-ричной системе. Ввести число и записать его в EAX. Считать, что число записано правильно и умещается в двойное слово.

5.30 Вводится последовательность символов из набора {'0', ..., '9'}, оканчивающаяся пробелом — правильная запись целого числа в десятичной системе. Используя операцию INCHAR, ввести число и записать его в регистр EAX. Считать, что число умещается в двойное слово.

5.31 Вводится последовательность символов, оканчивающаяся пробелом — правильная запись целого числа (возможно, со знаком) в десятичной системе. Используя операцию INCHAR ввести число и записать его в регистр EAX. Считать, что число умещается в двойное слово.

5.32 Дано число n (двойное слово, $n \geq 0$). Посчитать количество цифр 0 и 2 в записи числа n в 5-ричной системе. Результат записать в CL.

В следующих задачах требуется написать полную программу.

5.33 Дан непустой текст, оканчивающийся символом 'h'. Проверить, является ли текст записью числа в 16-ричной системе (без знака). Ответ (является/не является) напечатать.

5.34 Данна последовательность из 130 чисел. Определить и напечатать наибольшее из них. Задачу решить в предположении, что вводятся числа: а) без знака; б) со знаком. Первое число последовательности нельзя читать вне цикла.

5.35 Данна последовательность чисел, за последовательностью следует 0. Определить и напечатать наибольшее число последовательности. Задачу решить в предположении, что вводятся числа: а) без знака; б) со знаком. Первое число последовательности нельзя читать вне цикла. Предварительно выписать алгоритм на Паскале.

5.36 Данна последовательность из 100 чисел со знаком. Вычислить наибольшее и наименьшее из них. Реализовать алгоритм

```
read(a); min:=a; max:=a;
for i:= 2 to 100 do
begin read(a);
  if a>max then max:=a
  else if a<min then min:=a
end;
```

5.37 Данна последовательность из 100 попарно различных чисел. Вычислить два наибольших числа последовательности. Задачу решить в предположении, что вводятся числа: а) без знака; б) со знаком.

5.38 Данна последовательность из 100 чисел. Посчитать, сколько раз встречается минимальный элемент. Задачу решить в предположении, что вводятся числа: а) без знака; б) со знаком.

5.39 Дано 50 чисел, среди которых есть по крайней мере одно отрицательное. Найти наибольшее среди отрицательных чисел. Подсказка: в качестве начального значения максимума взять наименьшее машинное число со знаком.

5.40 Дано 150 чисел, среди которых есть по крайней мере одно положительное. Найти наименьшее среди положительных чисел. Подсказка: в качестве начального значения минимума взять наибольшее машинное число со знаком.

5.41 Данна непустая последовательность чисел со знаком, оканчивающаяся нулём.

- а)** Записать в CL значение 1, если в последовательности есть элемент, превосходящий число 100; в противном случае записать в CL значение 0.
- б) Напечатать первый элемент, превосходящий число 100, если такой есть; в противном случае ничего не печатать. В цикле не должно быть оператора печати.

5.42 Данна последовательность из 123 чисел со знаком. Определить, является ли эта последовательность возрастающей. Напечатать ответ («Является» или «Нет»).

5.43 Данна последовательность из 200 чисел без знака. Определить, есть ли в последовательности хотя бы одно число, кратное числу 3. Если есть, напечатать первое из таких чисел, если нет — напечатать сообщение об этом.

5.44 Данна последовательность чисел со знаком, возможно, пустая; за последовательностью следует 0. Определить, является ли эта последовательность знакочередующейся. Напечатать ответ («Является» или «Нет»). Пустая последовательность считается знакочередующейся.

5.45 Данна последовательность символов (отличных от точки), за которой следует точка. Определить, сбалансирована ли эта последовательность по круглым скобкам. Ответ: ДА или НЕТ.

5.46 Данна последовательность из 40 чисел. Определить, у скольких чисел этой последовательности равные «соседи» (т. е. равны предыдущее и последующее числа).

5.47 Данна непустая последовательность символов (отличных от точки), за которой следует точка. Напечатать эту же последовательность:

- а) заменив все 'PH' на 'F';
- б) удалив все лишние пробелы (т. е. из нескольких подряд идущих пробелов оставить только один)

5.48 Данна непустая последовательность непустых слов из больших латинских букв; между соседними словами — запятая, за последним словом — точка. Подсчитать количество слов, которые:

- а) начинаются с буквы 'S';
- б) оканчиваются буквой 'Z';
- в) начинаются и оканчиваются одной и той же буквой.

5.49 $P \text{ DD } ? ; P \geq 0$
 $Q \text{ DD } ? ; Q > 0$

Напечатать дробь P/Q в виде вещественного числа с 5 цифрами в дробной части.

5.50 $X \text{ DQ } ?$

Напечатать значение X в виде беззнакового десятичного числа. (Обратить особое внимание на возможность переполнения в команде DIV.)

6. Массивы

В этом параграфе используются обозначения: [] — символы РБНФ, жирные [] — терминальные символы.

Типичное описание массива (см. §1)

имя тип КВ DUP (операнд)

КВ задаёт количество элементов массива; операнд — начальные значения элементов (все элементы имеют одно и то же начальное значение).

Для описания матриц в качестве операнда указывается DUP-конструкция.

Операторы

type имя массива ; размер одного элемента массива в байтах

length имя массива ; длина массива (множитель в конструкции DUP)

size имя массива ; размер массива (type * length)

offset имя массива ; адрес начала массива (смещение первого элемента)

Адресное выражение с модификаторами

имя массива [+][r32][+][[множитель *] инд.рег.] + | - КВ
(1) (2) (3) (4)

Каждая из частей (1), (2), (3), (4) может отсутствовать.

множитель ::= 1 | 2 | 4 | 8 | (КВ1)

КВ1 должно принимать значение из набора {1, 2, 4, 8}. Допускаются ссылки вперёд.

инд.рег. ::= r32, кроме ESP

Константное выражение вида переменная1 – переменная2 нужно брать в скобки, если используются модификаторы.

Замечание. КВ можно вносить в квадратные скобки, т. е. запись [EBX-type A] является корректным выражением.

Команда LEA (load effective address)

LEA r32, m

Действие команды: 1) вычисляется Аисп второго операнда; 2) r32 := Аисп.

6.1 Пусть offset X = 1000h и есть описания

X DB 1

Y DW 2

Z DD 3

Разделить следующие выражения на верные и ошибочные.

- 1) Для неверных выражений объяснить, в чём ошибка.
- 2) Для верных адресных выражений определить тип и смещение (содержимое поля адреса команды в объектном коде после трансляции).
- 3) Для верных константных выражений вычислить значение.

- | | | | |
|-------------|---------------|------------------|--------------------|
| a) X[ESI-1] | e) EAX+ESI | л) X+Y[ESI] | p) Y[ESI][2*ESI] |
| б) X[1-EBX] | ж) [EAX][EBX] | м) Y[ESP][EDI] | с) X[EBX]+[2*ESI] |
| в) Y[EDI]-2 | з) SI+2 | н) Y[ESP][ESP] | т) X[3*EDI] |
| г) 2-Z[EDI] | и) X[BH] | о) X[ESI][ESI] | у) Z[2*EBX][4*ESI] |
| д) EBX[EAX] | к) Z[EBX]4 | п) X[ESP][2*ESP] | ф) X[EBX]+[ESI]+2 |

6.2 X DW 500h, 600h, 700h

Пусть EBX = 8000 0000h и ESI = 0FFFF 9102h, offset X = 120FFh. Указать исполнительный адрес операнда следующей команды:

- | | | |
|-----------------|----------------------|-------------------------------|
| a) DEC X+2 | г) INC X[ESI]-2 | ж) NEG X[EBX+ESI]+0A000 0000h |
| б) INC X[ESI] | д) NEG X[EBX][ESI] | з) DEC X[4*ESI] |
| в) INC X[ESI+2] | е) NEG X[EBX][ESI]+2 | и) NEG X[EBX+2*ESI] |

6.3 X DD 0,1,2

В этой задаче будем считать правильным адресным выражением запись вида KB+[r32]. Пусть X обозначает ячейку памяти с адресом 100. Указать значения регистров EAX, EBX и ECX после выполнения следующих команд:

- | | |
|--|---|
| a) MOV EAX, X
LEA EBX, X
MOV ECX, offset X | в) MOV EAX, X+4
LEA EBX, X+4
MOV ECX, offset X+4 |
| б) MOV ESI, 4
MOV EAX, X+[ESI]
LEA EBX, X+[ESI]
MOV ECX, offset X+[ESI] | г) LEA ESI, X
MOV EAX, 4+[ESI]
LEA EBX, 4+[ESI]
MOV ECX, [EBX] |

6.4 С помощью одной команды LEA реализовать следующее действие:

- увеличить значение регистра EBX на 10;
- уменьшить значение регистра EBX на 10;
- записать в регистр EAX значение регистра EDI;
- записать в регистр ESI значение регистра EBX, увеличенное на 1;
- увеличить в три раза значение регистра EAX (записать в регистр EAX значение 3*EAX);
- записать в регистр EBX сумму значений регистров ECX и EDX.

6.5 X DB 75 DUP (?)

Y DW 25 DUP (?)

Z DD 50 DUP (?)

a) Считая, что offset X = 0, определить значения выражений

offset X type X length X size X

offset Y type Y length Y size Y

offset Z type Z length Z size Z

б) Выписать адресное выражение, соответствующее первому элементу каждого массива, и адресное выражение, соответствующее последнему элементу.

6.6 X DB 50 DUP (?) ; X[0..49]

Пусть в регистре EBX находится адрес некоторого элемента массива X (т. е. offset X+i, где i — число от 0 до 49). Требуется в этом же регистре получить индекс (i) этого элемента. Определить, какая из следующих групп команд правильно решает эту задачу:

a) SUB EBX, X

6) LEA EAX, X

b) SUB EBX, offset X

SUB EBX, EAX

6.7 N=20

K DD ? ; 11≤K≤19

X DW N DUP (?) ; X[11..30]

Z DD N DUP (?) ; Z[0..19]

Выписать последовательность команд, реализующих оператор

begin X[K]:= 0; Z[K]:= 0 **end**

6.8 Q DB 256 DUP (?) ; Q[0..255] of 0..255

Считая, что в регистре AL находится число i от 0 до 255, реализовать присваивание Q[Q[i]]:=i.

6.9 Написать полную программу для решения задачи: ввести с клавиатуры число n из диапазона 1..26 и напечатать первые n букв латинского алфавита. Циклы не использовать.

6.10 Написать полную программу для решения задачи: ввести последовательность из 15 чисел со знаком (байты), записав их в массив. Напечатать элементы массива, являющиеся нечётными числами, в порядке их вхождения в последовательность.

6.11 Описать массив двойных слов (числа без знака). Считая, что массив заполнен, напечатать последний элемент массива.

6.13 N=30

A DB N DUP (?) ; A[0..N-1], числа со знаком

X DD N DUP (?) ; X[0..N-1], числа со знаком

Напечатать элементы массива А с указанием их индексов; затем напечатать таким же образом элементы массива Х.

6.14 N=100

X DB N DUP (?) ; X[0..N-1]

Y DW N DUP (?) : Y[0..N-1]

Z DD N DUP (?) : Z[0..N-1]

Решить следующую задачу двумя способами: 1) не используя множитель при модификаторе; 2) используя множители при модификаторах.

- а) Выписать последовательность операторов, которая заполняет массивы X, Y и Z по правилу $X[i]:=i$, $Y[i]=i$, $Z[i]:=i$.

б) Считая, что массивы содержат числа без знака, в каждом массиве посчитать количество элементов, которые больше своего индекса ($X[i]>i$, $Y[i]>i$, $Z[i]>i$). Результаты записать в регистры AL, AH и DL. Использовать один цикл.

6.15 B DD 146 DUP (?)

Записать в элементы массива В их адреса: $B_i := \text{адрес } B_i$.

6.16 N=100

X DD N DUP (?) ; X[0..N-1] - числа со знаком

Решить следующую задачу:

- а) записать в регистр EAX число нулевых элементов массива X;
 - б) записать в регистр EAX наименьший элемент массива X;
 - в) записать в регистр EAX наименьший элемент массива X, в регистр BL — его индекс (от 0 до 99);
 - г) обнулить все отрицательные элементы массива X.

6.17 Написать программу решения задачи. Ввести последовательность из 120 чисел без знака. Напечатать последовательность в обратном порядке.

6.18 Написать программу решения задачи. Дан текст из маленьких латинских букв, за текстом — точка. В тексте не больше 70 символов. Напечатать текст в обратном порядке.

6.19 L=31

A DW L DUP (?) ; числа со знаком

Выписать фрагмент программы решения следующей задачи. Внутри цикла нельзя менять элементы массива.

- а) Заменить на 1 первый отрицательный элемент массива A.
- б) Изменить знак у первого элемента массива A, отличного от 1.
- в) Заменить на -1 последний положительный элемент массива A.
- г) Изменить знак у последнего элемента массива A, отличного от 0.

6.20 N=25

A DB N DUP (?) ; A[0..N-1]

B DB N DUP (?)

Z DB N DUP (?)

Пусть массивы A, B и Z представляют собой длинные числа: каждый элемент массива A хранит одну десятичную цифру числа, например, элемент A[0] — старшая цифра, A[N-1] — младшая цифра (единицы). Реализовать следующие операции с длинными числами

- а) Z:=A+B
- б) Z:=A-B
- в) Z:=0-B

6.21 N=30

Z DB N DUP (?) ; Z[0..N-1], числа со знаком

Решить следующую задачу:

- а) Вычислить максимальный элемент среди элементов с нечётными индексами. Результат записать в AL.
- б) поменять знаки у всех элементов массива Z с чётными индексами.

6.22 N=15

X DD N DUP (?); числа без знака

- а) Проверить, упорядочен ли массив по возрастанию; ответ напечатать в виде текста («возрастает» или «не возрастает»).
- б) Проверить, образуют ли элементы массива арифметическую прогрессию. Если образуют, напечатать текст «арифметическая прогрессия» и напечатать разность прогрессии, если не образуют напечатать текст «нет».
- в) Проверить, образуют ли элементы массива геометрическую прогрессию с целым знаменателем. Если образуют, напечатать текст «геометрическая прогрессия» и напечатать знаменатель прогрессии, если не образуют напечатать текст «нет».

6.23 N=100

X DD N DUP (?)

- а) Проверить, является ли массив симметричным; ответ напечатать в виде текста «симметричен» или «не симметричен».
- б) Переставить элементы массива X в обратном порядке.

6.24 N=100

X DD N DUP (?)

- а) Циклически сдвинуть влево на две позиции массив X.
- б) Циклически сдвинуть вправо на две позиции массив X.

6.25 N=100

X DW N DUP (?)

Решить следующую задачу:

- а) определить, у скольких элементов массива X равные соседи (предыдущий и последующий элементы), и записать ответ в регистр AL;
- б) если левая и правая половины массива X совпадают, то обнулить последний элемент этого массива.

6.26 N=100

X DW N DUP (?)

Определить, есть ли в массиве X совпадающие элементы. Ответ (есть/нет) напечатать.

6.27 N DD ? ; число без знака

S DB 10 DUP ('0') ; символьная строка из цифр

- а) Записать в S десятичные цифры N (например, для N=304 будет S='0000000304').

- б) Решить обратную задачу: по массиву цифр S построить десятичное число N.

Считать, что младшая цифра числа содержится в последнем байте S.

6.28 Написать программу решения следующей задачи. Дан текст — непустая последовательность малых латинских букв, за которой следует точка. Длина последовательности не больше 255. Используя подходящий вспомогательный массив, решить задачу (ответ напечатать):

- а) Определить, сколько различных букв входит в текст.
- б) В алфавитном порядке напечатать буквы, входящие в текст.
- в) Определить, какая из букв чаще всего встречается в тексте. Если таких букв несколько, выбрать любую.

6.29 Написать программу решения задачи. Дан текст — непустая последовательность символов, за которой следует точка. Напечатать в порядке возрастания все различные цифры, входящие в текст. Если в тексте нет цифр, ничего не печатать.

6.30 A DQ 40 DUP (?) ; числа без знака

Заменить первый элемент массива A на максимальный элемент.

6.31 m=15

n=20

Описать матрицу A[m×n]. Считая, что матрица заполнена

- посчитать и напечатать количество отрицательных элементов в A;
- найти и напечатать минимальный элемент матрицы A.

6.32 m=10

n=15

B DD m DUP(n DUP(?)) ; матрица B[1..m,1..n] чисел со знаком

a) Напечатать матрицу B (по строкам).

б) Напечатать первый столбец матрицы B.

в) Напечатать последний столбец матрицы B.

г) Напечатать матрицу B по столбцам — в первой строке экрана напечатать первый столбец матрицы, во второй — второй столбец и т. д.

6.33 N=99

A DD N DUP (N DUP (?)) ; матрица чисел со знаком

a) Поменять знаки на противоположные у элементов, стоящих на главной диагонали.

б) Обнулить элементы на побочной диагонали.

в) Транспонировать матрицу A.

г) Проверить, является ли матрица A симметрической. Ответ записать в регистр AL: 1 — симметрическая, 0 — нет.

6.34 Пусть в регистрах ESI и EDI находятся начальные адреса двух (непересекающихся) областей памяти из 20 двойных слов в каждой.

Решить следующую задачу:

а) обнулить все двойные слова первой из этих областей (ESI);

б) записать в каждое двойное слово первой области его адрес;

в) сравнить содержимое обеих областей и записать ответ 1 (равны) или 0 в регистр AL.

г) скопировать содержимое первой области (ESI) во вторую область.

7. Структуры

Ниже использовано обозначение {}, [] — метасимволы РБНФ.

Описание типа

Состоит из нескольких директив.

имя типа STRUC

 описание поля

 { описание поля }

имя типа ENDS

Синтаксис конструкции *описание поля* такой же, как у директив определения данных. Допускаются поля — структуры и записи (см. §9).

Семантика. Ассемблер запоминает информацию об имени типа, именах и расположении полей в структуре. Значение имени поля — смещение поля относительно начала структуры.

Описание переменной

имя переменной имя типа операнд {, операнд}

операнд ::= <>-иниц. | KB DUP (<>-иниц.)

<>-иниц. ::= < [значение поля {, значение поля}] >

значение поля ::= ? | KB | AB | иниц. массива | иниц.структуры | иниц.записи

иниц. массива ::= KB DUP (оп1)

иниц.структуры ::= <>-иниц.

иниц.записи см. §9

Допустимые значения для оп1 в инициализации поля-массива определяются типом элемента массива.

Семантика. Ассемблер заносит в ТИ информацию об имени переменной и заполняет байты объектного кода в соответствии с указанными операндами.

Операторы

Операторы type, length и size применимы к имени типа, имени переменной и к имени поля. Смысл операторов — см. §6.

size имя типа ; размер в байтах одной переменной указанного типа.

Применение к полю:

оператор имя типа.имя поля

оператор имя переменной.имя поля

Доступ к полю переменной

Для доступа к полю используется оператор точка, выражение выглядит так

AB.имя поля

Тип выражения = типу поля.

Смещение = смещение переменной + значение имени поля.

Если в AB есть модификация адресов, AB нужно брать в скобки:

(AB).имя поля

Если в AB нет имени переменной, необходимо указать тип структуры

(имя типа ptr AB).имя поля

```

7.1 DATE STRUC ; дата
    Y DW 1997 ; год
    M DB ? ; номер месяца
    D DB ? ; число
    WD DB 'воскресенье' ; день недели
DATE ENDS

```

- а) Описать переменные D1, D2, D3 и D4 типа DATE со следующими начальными значениями их полей (знак ? означает, что поле не должно иметь начального значения):

	Y	M	D	WD
D1:	1945	5	13	среда
D2:	1997	12	?	четверг
D3:	1980	?	?	воскресенье
D4:	1997	?	?	воскресенье

Директивы, описывающие эти переменные, должны быть максимально короткими.

б) Указать значение выражений

- 1) type D1; size D1; size DATE
- 2) type D1.Y; type D1.D; type D1.WD
- 3) length D1.Y; length D1.WD; size D1.Y; size D1.WD
- 4) offset D1.D, если offset D1=100h

7.2 TIME STRUC ; время какого-то момента суток

```

    H DB ? ; час (от 0 до 23)
    M DB ? ; минута (от 0 до 59)
    S DB ? ; секунда (от 0 до 59)
TIME ENDS
T1 TIME <>
T2 TIME <>

```

- а) Записать в переменную T1 значение 17 часов 25 минут 0 секунд.
 б) Реализовать присваивание T2 := T1.
 в) Увеличить время T2 на 1 секунду, учитывая смену суток.
 г) Реализовать присваивание CL:=T1<T2. Считать, что «истина» изображается числом 0FFh, «ложь» — числом 00h.

7.3 Описать массив T из 70 моментов времени типа TIME (см. № **7.2**). Считая массив заполненным, решить задачу: напечатать первый из моментов времени после полудня, если таких элементов нет, ничего не печатать.

- а) Полдень 12:00:00 печатать.
- б)** Полдень 12:00:00 не печатать.

7.4 Описать структуру DATE (дата) с полями день, месяц, год. Описать массив $dts[0..90]$ дат типа DATE.

- а) вычислить значения type, length, size для массива dts;
- б) ввести число k ($0 \leq k \leq 90$), записать в $dts[k]$ сегодняшнюю дату;
- в) записать в последний элемент массива dts дату 31.12.2023;
- г) посчитать и напечатать количество весенних дат в массиве.

7.5 Описать структуру DATE (дата) с полями день, месяц, год и структуру PERSON (человек) с полями: FAM (фамилия) — строка из 20 байтов, BDATE (дата рождения) — типа DATE. Описать переменные P1 и P2 типа PERSON. Считая, что этим переменным в программе уже присвоены какие-то значения, решить следующую задачу:

- а) В отдельных строках напечатать фамилии P1 и P2.
- б) Определить, моложе ли P1, чем P2? Ответ (1 — моложе, 0 — нет) записать в AL.

7.6 Описать структурный тип STUD (студент) со следующими полями: FAM (фамилия) — строка из 20 байтов, AGE (возраст) — 1 байт, MARKS (отметки) — массив из 4 байтов. Описать также (без начальных значений) переменную P типа STUD и массив GR (группа студентов) из 40 элементов того же типа.

Считая, что этим переменным в программе уже присвоены какие-то значения, решить следующую задачу, ответ записать в регистр AL.

- а) определить, является ли P человеком в возрасте 17 лет, вторая буква в фамилии которого — это «е» (ответ: 1 (является) или 0);
- б) определить, сколько людей из GR имеют тот же возраст, что и P;
- в) напечатать (в столбик) фамилии всех студентов группы GR;
- г) напечатать фамилию самого молодого человека из группы GR (любого, если таких несколько);
- д) определить, сколько людей из GR имеют хотя бы одну отметку «4»;
- е) определить, сколько отличников есть в группе GR;
- ж) определить, есть ли в GR хотя бы одна пара однофамильцев (ответ: 1 (есть) или 0).

8. Команды работы с битами

Команды трактуют содержимое операндов как последовательность битов, действие выполняется с каждым битом независимо, побитово.

Логические команды

```
AND op1,op2 ; op1 := op1 and op2  
OR op1,op2 ; op1 := op1 or op2  
XOR op1,op2 ; op1 := op1 ≠ op2  
NOT op1 ; op1 := not op1, не устанавливает флаги  
TEST op1,op2 ; op1 and op2 без сохранения результата, устанавливается ZF
```

Все команды, кроме NOT, устанавливают флаг ZF.

Удобно использовать команды переходов

```
JZ метка ; переход, если ZF=1 (сионим команды JE)  
JNZ метка ; переход, если ZF=0 (сионим команды JNE)
```

Сдвиги

Логические

```
SHL op1,op2 ; сдвиг op1 влево на op2 разрядов  
SHR op1,op2 ; сдвиг op1 вправо на op2 разрядов
```

Циклические

```
ROL op1,op2 ; поворот op1 влево на op2 разрядов  
RCL op1,op2 ; поворот op1 влево на op2 разрядов через CF  
ROR op1,op2 ; поворот op1 вправо на op2 разрядов  
RCR op1,op2 ; поворот op1 вправо на op2 разрядов через CF
```

Требования к operandам: op2 может быть либо регистром CL, либо константным выражением.

Все команды сдвигов устанавливают флаг CF. Удобно использовать команды переходов

```
JC метка ; переход, если CF=1  
JNC метка ; переход, если CF=0
```

8.1 Указать значения регистра AL и флага ZF после выполнения следующей пары команд:

- | | | |
|---------------------------------|--------------------------------|--------------------------------|
| a) MOV AL,1101b
AND AL,0111b | b) MOV AL,100b
AND AL,011b | c) MOV AL,100b
TEST AL,011b |
| g) MOV AL,0F0h
OR AL,0Fh | d) MOV AL,1010b
XOR AL,0FFh | e) MOV AL,101b
XOR AL,AL |

8.2 Пусть под логические переменные A и B выделено по байту:

A DB ?

B DB ?

и пусть выбрано следующее представление для логических значений: «ложь» — нулевой байт, «истина» — любой ненулевой байт. Реализовать для этого представления следующие операции:

a) A := not A

б) A := A and B

в) A := A or B

8.3 Предложить машинное представление (размером в байт) для логических значений, которое отлично от известного представления false = 00h, true = 0FFh и при котором команды NOT, AND и OR правильно реализуют логические операции отрицания, конъюнкции и дизъюнкции соответственно.

8.4 Написать фрагмент программы — секцию данных и последовательность команд для решения задачи: Ввести число. Если число нечётное, напечатать true, в противном случае напечатать false. Для проверки нечётности числа использовать команду TEST.

8.5 N=3000

Y DQ N DUP (?)

K DB ?

Записать в переменную K значение 1, если массив удовлетворяет указанному ниже условию, и значение 0, если массив не удовлетворяет условию. Тело цикла должно содержать не более двух команд, считая команду loop. Проверяемое условие:

- а) все элементы массива — нечётные числа;
- б) в массиве есть хотя бы одно нечётное число.

8.6 Указать значения регистра AL и флага CF после выполнения следующей группы команд:

- а) MOV AL,101b б) MOV AL,11001b в) MOV AL,110b г) MOV AL,110b
SHL AL,1 SHL AL,5 SHR AL,1 SHR AL,4
- д) MOV AL,0F0h е) MOV AL,101b ж) MOV AX,1234h з) MOV AX,1234h
ROL AL,1 ROR AL,2 SHR AH,1 SHR AH,1
 RCR AL,1 RCL AL,1
- и) MOV ECX,4 к) MOV ECX,4
MOV AX,1234h MOV AX,1234h
- L: SHR AH,1 L: SHR AH,1
RCR AL,1 RCL AL,1
LOOP L LOOP L

8.7 N=1000

X DQ N DUP (?)
K DB ?

Записать в переменную K значение 1, если массив удовлетворяет указанному ниже условию, и значение 0, если массив не удовлетворяет условию. Тело цикла должно содержать не более двух команд, считая команду loop. Проверяемое условие:

- а) все элементы массива — отрицательные числа;
- б) в массиве есть хотя бы одно отрицательное число.

8.8 N=200

X DD N DUP (?)

Для массива X посчитать количество а) нечётных; б) чётных; в) отрицательных; г) неотрицательных элементов. Результат записать в EAX. Элементы массива не портить. Использовать не более 4 команд в цикле, считая команду loop.

8.9 X DQ ? ; число со знаком

Не используя команды условного перехода, записать в регистр AL знаковый бит числа X, т. е. 1, если $X < 0$, и 0 иначе.

8.10 Вводится последовательность символов из набора {'0', '1', '2', '3'}, оканчивающаяся пробелом — правильная запись целого числа в 4-ричной системе. Ввести число и записать его в регистр EAX. Считать, что число умещается в двойное слово. Операцию умножения не использовать.

8.11 Используя команду OUTCHAR, напечатать содержимое регистра EAX в виде числа без знака в 4-ричной системе. Напечатать все цифры числа, включая незначащие нули. Операцию деления не использовать.

8.12 Вводится последовательность символов из набора {'0', ..., '9', 'A', 'B', 'C', 'D', 'E', 'F'}, оканчивающаяся пробелом, — правильная запись целого числа в 16-ричной системе. Ввести число и записать его в регистр EAX. Считать, что число умещается в двойное слово. Операцию умножения не использовать.

8.13 Напечатать содержимое регистра EAX в 16-ричной системе (восемь цифр). Команду деления не использовать.

8.14 Решить следующую задачу:

- а) Подсчитать число двоичных единиц в значении регистра EAX и записать это число в регистр DH. Тело цикла должно содержать не более трёх команд, считая команду LOOP.
- б) Не используя команды деления, с помощью команды OUTCHAR напечатать значение регистра EAX в виде беззнакового двоичного числа (32 двоичные цифры). В теле цикла не более 5 команд.
- в) Перевернуть содержимое регистра EAX.
- г) Не используя команды умножения, ввести беззнаковое двоичное число и записать его в регистр EAX. Считать, что число задано без ошибок, заканчивается пробелом и умещается в двойное слово.

8.15 Проверить, является ли содержимое регистра EAX как последовательность битов палиндромом (т. е. симметричным). Ответ (да/нет) напечатать.

8.16 С помощью команды OUTCHAR напечатать значение регистра EAX в виде беззнакового двоичного числа без незначащих нулей. Команду деления не использовать.

8.17 Проверить, является ли число без знака — содержимое регистра EAX — степенью двойки, т. е. существует k : $0 \leq k \leq 31$, $EAX = 2^k$. Команды умножения и деления не использовать. Указание: в двоичной системе степень двойки выглядит как 000...010...00, начальная или конечная последовательность нулей может отсутствовать. Ответ (да/нет) напечатать.

8.18 Проверить, является ли содержимое регистра EAX числом вида $2^k - 1$, т. е. последовательность разрядов выглядит как 000...011...1, начальная последовательность нулей может отсутствовать. Ответ (да/нет) напечатать.

8.19 Q DQ ?

K DB ? ; 0≤K≤63

Реализовать сдвиг SHL Q, K.

8.20 N=32

X DD N DUP (?) ; X[0..N-1]

Преобразовать массив X по правилу: циклически сдвинуть элемент $X[i]$ влево на i разрядов, для всех $i \in [0, N - 1]$.

8.21 Пусть X и Y — двойные слова, числа без знака. Не используя команды умножения и деления, реализовать следующее присваивание:

a) $Y := 4 * X - X \text{ div } 8 + X \text{ mod } 16$

б) $Y := 35 * X$

Считать, что результат вычислений и промежуточные результаты умещаются в двойное слово.

8.22 Пусть X — двойное слово, число без знака. Не используя команды умножения и деления, выписать последовательность команд, реализующих присваивание

a) $Y := (X + 52) \text{ div } 16 + X * 32$

б) $Y := X \text{ mod } 8 + X \text{ mod } 16 + X \text{ mod } 32$

Описать переменную Y подходящего размера. Фрагмент должен корректно работать для любых исходных данных, даже если промежуточные результаты не умещаются в двойное слово.

8.23 A DD ?

B DD ?

Z DQ ?

Пусть A, B — двойные слова, числа без знака, Z — четверное слово. Реализовать «быстрое» умножение $Z := A \cdot B$.

Алгоритм: Пусть $B = b_{31} \cdot 2^{31} + b_{30} \cdot 2^{30} + \dots + b_2 \cdot 2^2 + b_1 \cdot 2 + b_0$, $b_i \in \{0, 1\}$.

Тогда $A \cdot B = ((\dots (A \cdot b_{31} \cdot 2 + A \cdot b_{30}) \cdot 2 + \dots + A \cdot b_2) \cdot 2 + A \cdot b_1) \cdot 2 + A \cdot b_0$.

Умножение на 2 реализуется сдвигом влево на один разряд. Действие « $+ A \cdot b_i$ » означает, что нужно прибавить A , если $b_i = 1$.

Приходим к алгоритму:

```
begin Z:=0;
    for i:=31 downto 0 do
        begin Z<<1; {Z*2}
            B<<1; {CF=bi}
            if CF=1 then Z:=Z+A
        end
    end
```

Здесь знак $<<$ обозначает сдвиг содержимого первого операнда влево. Величины, используемые в цикле, хранить в регистрах.

8.24 Пусть переменные A, B — двойные слова, числа без знака, а C — переменная типа qword. Не используя команды умножения, реализовать операцию $C := 16 * A + B$.

8.25 A DW 0

 B DW 0

 C DW 0

Переменные А, В и С — представляют множества **set of '0'..'9'**, старший бит соответствует '0'. (Например, множество {'1'} изображается словом 4000h.) Реализовать следующие операции

- а) Записать в С множество {'2', '4', '5'}
- б) Записать в С объединение множеств С := А ∪ В
- в) Записать в С пересечение множеств С := А ∩ В
- г) Записать в С разность множеств С := А \ В
- д) Распечатать множество С: напечатать все элементы, входящие в множество С.
- е) Пусть CL содержит символ. Проверить, принадлежит ли множеству С символ CL. Результат (0FFh — true, 0 — false) записать в AL.

8.26 Написать программу решения задачи. Дан текст — последовательность строчных латинских букв, возможно пустая, за которой следует точка. В алфавитном порядке напечатать все различные буквы, встречающиеся в тексте. Хранить буквы текста в множестве **set of 'a'..'z'**; для представления множества использовать двойное слово.

8.27 Написать программу решения задачи. Дан текст — последовательность строчных латинских букв, возможно пустая, за которой следует точка. Посчитать, сколько различных букв входит в текст. Хранить буквы текста в множестве **set of 'a'..'z'**; для представления множества использовать двойное слово.

8.28 X DB 13 DUP (?) ; X: array [1..13] of 0..3
 PX DD ? ; PX: packed array [1..13] of 0..3

Реализовать следующую работу с упакованным массивом PX.

(Подсказка: для хранения одного элемента достаточно двух битов.)

- а) Упаковать массив X в двойное слово PX, т. е. записать элементы массива X в PX: X[1] — в два старшие бита PX, X[2] — в два следующих бита и т. д.
- б) Распаковать PX в массив X: заполнить массив X элементами, хранящимися в PX.
- в) Ввести число k ($1 \leq k \leq 13$) и записать ноль в PX[k].
- г) Ввести число k ($1 \leq k \leq 13$) и напечатать элемент PX[k].

9. Записи

Ниже использовано обозначение {}, [] — метасимволы РБНФ.

Описание типа

имя типа RECORD имя поля : KB1 = KB2 {, имя поля : KB1 = KB2 }
KB1, KB2 без ссылок вперёд; KB1 — ширина поля в битах, KB2 — начальное значение поля по умолчанию.

Имена полей должны быть уникальные. Суммарная ширина полей не больше 32, запись не может превосходить двойное слово.

Семантика. Ассемблер запоминает информацию об имени типа, именах полей, их расположении в записи, ширине и значениях полей.

Описание переменной

имя переменной имя типа операнд {, операнд}
операнд ::= <>-иниц. | KB DUP (<>-иниц.)
<>-иниц. ::= < [значение поля {, значение поля}] >
значение поля ::= | KB

Семантика. Ассемблер заносит в ТИ информацию об имени переменной и заполняет байты объектного кода в соответствии с указанными операндами.

Операторы и значение имени поля

имя поля — константа, значение = правый сдвиг (на сколько битов сдвинуть всю запись вправо для выравнивания поля по правой границе)
width имя поля (width имя типа) — ширина поля (записи) в битах
mask имя поля (mask имя типа) — битовая маска для выделения поля (записи)

9.1 Пусть есть описание типов

R1 RECORD A:1, B:2=3
R2 RECORD M:5=2, N:10, L:12=0

а) Указать значения выражений type R1, type R2.

б) Тип R1: вычислить значения выражений

width R1, width A, width B;
mask R1, mask A, mask B;
A, B.

в) Тип R2: вычислить значения выражений (значения масок указать в 16-ричной системе)

width R2, width M, width N, width L;
mask R2, mask M, mask N, mask L;
M, N, L.

9.2 Есть описание типа

R RECORD A:3=2, B:3, C:10=3FFh

Указать, как будут заполнены поля переменных, описанных следующими директивами

- а) V1 R <> б) V2 R <1,2,3> в) V3 R <, ,7> г) V4 R <7>

9.3 Описан тип time — момент времени (в 12-часовом формате) с полями час, минута, секунда

time RECORD h:4=0, m:6=0, s:6=0

и описана переменная T типа time

T time <>

- а) Записать в переменную T значение 2 час. 35 мин. 05 сек. Решение должно состоять из одной команды.
- б) Считая, что переменная T получила некоторое значение, напечатать значение T в виде час:минута:секунда, разделяя поля двоеточиями.
- в) Ввести три числа — час, минуту и секунду — и заполнить поля переменной T.

9.4 time RECORD h:4=0, m:6=0, s:6=0 ; час, минута, секунда

Описать массив T из 70 элементов типа time. Выписать последовательность команд для решения задачи

- а) Посчитать количество элементов массива, у которых поле секунда равно нулю.
- б) Посчитать в массиве T количество моментов времени до 7.00.
- в) Вычислить наибольший элемент в массиве T. Просмотр первого элемента массива из цикла не выносить. Значение наибольшего элемента напечатать.
- г) Вычислить наименьший элемент в массиве T. Просмотр первого элемента массива из цикла не выносить. Значение наименьшего элемента напечатать.
- д) Описать mins — переменную типа time. Записать в mins элемент массива T, у которого поле s имеет наименьшее значение. Если таких элементов несколько, записать любой из них.
- е) Описать maxm — переменную типа time. Записать в maxm элемент массива T, у которого поле m имеет наибольшее значение. Если таких элементов несколько, записать любой из них.

```
9.5 time RECORD h:4=0, m:6=0, s:6=0
      day STRUC
        WD DB 11 DUP (' '),0 ; день недели
        T time <>           ; время срабатывания будильника
      day ENDS
```

Описать массив D из 365 элементов типа day (D[1..365]).

- а) Записать в поле T второго элемента массива D значение 6 часов 45 минут.
- б) Напечатать индексы элементов массива D, у которых время срабатывания будильника не раньше 9.00. Считать, что нумерация элементов начинается с 1.
- в) Напечатать название дня недели, когда будильник срабатывает позже всего.

9.6 Описать запись DATE для хранения даты в 21 веке с полями год (от 00 до 99), месяц и день. Описать переменные D1 и D2 типа DATE.

- а) Записать в регистр AL число 0FFh, если D1 меньше D2, и 0 — в противном случае.
- б)** Напечатать дату D1 в формате день.месяц.год (например, 1.1.2020)
- в)** Напечатать дату D1 с указанием трёхбуквенного обозначения месяца (например, 1 янв 2020). Написать фрагмент программы: необходимые данные и команды.
- г) Описать массив X из 47 элементов типа DATE. Считая, что массив заполнен, вычислить и записать в байтовую переменную S количество летних дат в массиве X.

9.7 Описать структуру STUD (студент) с полями SNAME (фамилия), NAME (имя) и BDAY (день рождения). Поля SNAME, NAME — строки, поле BDAY — запись. Описать массив Gr из 25 структур типа STUD. Считая массив заполненным,

- а) посчитать количество студентов, родившихся в январе (ответ записать в EAX);
- б) напечатать фамилию и имя самого младшего студента.

10. Стек

Команды записи в стек

PUSH op ; op – r32, m32, i32

Действие команды: 1) ESP := ESP – 4; 2) [ESP] := op

PUSHFD ; PUSH EFlags

Команды взятия из стека

POP op ; op – r32, m32

Действие команды: 1) op := [ESP]; 2) ESP := ESP + 4

POPFD ; POP EFlags

10.1 Используя любые регистры как вспомогательные, описать через другие команды действие команды:

a) PUSH EAX

б) POP EAX

10.2 Указать значения байтов с адресами ESP, ESP+1, ESP+2, ESP+3 после выполнения команд

MOV EAX, 01020304h

PUSH EAX

10.3 Выписать фрагмент программы: Вводится последовательность символов, за которой следует точка. Напечатать последовательность в обратном порядке. Если последовательность пустая (введена только точка), ничего не печатать.

10.4 Выписать фрагмент программы, в котором вводится последовательность ненулевых чисел, заканчивающаяся нулем, и эти числа выводятся в обратном порядке, но только если среди них нет отрицательных чисел (в противном случае ничего не выводить). Исходное значение регистра ESP должно быть сохранено.

10.5 Написать полную программу решения задачи. Дано число n , $n > 0$. Напечатать квадрат размером $n \times n$ из символов '*'. Для реализации вложенных **for** использовать стек.

10.6 Фрагмент программы. Описать матрицу $A[m \times n]$, элементы — слова, числа со знаком. Считая, что матрица заполнена, вычислить и напечатать максимальные элементы строк матрицы A (каждое число — в отдельной строке). Для реализации вложенных **for** использовать стек.

10.7 Описать матрицу $A[m \times n]$, элементы — двойные слова, числа со знаком. Записать в EAX максимальный элемент матрицы. Для реализации вложенных **for** использовать стек. В качестве начального значения EAX использовать минимальное отрицательное число.

10.8 Описать матрицу $A[m \times n]$, элементы — двойные слова, числа со знаком. Описать переменные `min`, `im`, `jm` — двойные слова. Записать в `min` минимальный элемент матрицы, а в `im`, `jm` — его индексы. Если минимальных элементов несколько, сохранить индексы первого найденного. Для реализации вложенных **for** использовать стек.

10.9 Пусть в стеке записано 40 двойных слов. Реализовать указанную ниже операцию. Если в задаче не указано обратное, стек не портить.

- а) Поменять местами два «верхних» элемента (двойные слова) стека, сохранив при этом значения всех регистров и не используя переменные.
- б) Не вынимая элементы из стека, посчитать количество элементов, равных нулю. Ответ записать в регистр AL.
- в) Не вынимая элементы из стека, заменить все нечётные числа в стеке на число 1.
- г) Не вынимая элементы из стека определить, есть ли в стеке хотя бы два одинаковых двойных слова. Ответ записать в регистр AL: 1 — есть, или 0 — нет.
- д) Не вынимая элементы из стека, рассматривая элементы стека как адреса (смещения) некоторых байтов из сегмента данных, обнулить все эти байты.
- е) Не вынимая элементы из стека, рассматривая элементы стека как адреса (смещения) некоторых байтов из сегмента данных, подсчитать количество чётных чисел среди этих байтов, записать результат в AL.

10.10 Пусть в стеке записано 200 двойных слов и пусть в сегменте данных описан массив X из 200 двойных слов. Не используя команду POP и не изменяя значение регистра ESP, скопировать содержимое стека в массив X («верхнее» двойное слово стека должно быть записано в начальный элемент массива).

10.11 Используя только операцию OUTCHAR, вывести содержимое регистра EAX в виде

- а) беззнакового десятичного числа;
- б) знакового десятичного числа.

10.12 Написать программу. Дан непустой текст состоящий из круглых и квадратных скобок, за текстом следует точка. Проверить текст на согласованность скобок. Напечатать ответ Yes или No.

10.13 Написать программу решения задачи: Дан непустой текст, состоящий из цифр, знаков бинарных операций +, -, * и /. За текстом следует знак равенства (=). Текст представляет собой корректную запись формулы в постфиксном виде. Ввести текст, вычислить значение формулы и напечатать полученное значение. Считать, что промежуточные результаты не превосходят размера двойного слова и что операция / изображает `div`.

10.14 Написать программу для решения следующей задачи.

а) Для ввода задана последовательность символов, представляющая собой (без ошибок) формулу следующего вида:

⟨формула⟩ ::= ⟨цифра⟩ | (⟨формула⟩⟨знак⟩⟨формула⟩)

⟨знак⟩ ::= + | -

⟨цифра⟩ ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Вычислить значение этой формулы. (Пример: $((5-2)+7) \rightarrow 10$)

б) Для ввода задана последовательность символов, представляющая собой (без ошибок) формулу следующего вида:

⟨формула⟩ ::= ⟨цифра⟩ | M(⟨формула⟩,⟨формула⟩)

| m(⟨формула⟩,⟨формула⟩)

⟨цифра⟩ ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

где M трактуется как max (максимум), а m — как min (минимум).

Вычислить значение этой формулы. (Пример: M(2,m(5,7)) → 5)

11. Процедуры

Описание процедуры

имя процедуры PROC

; команды

имя процедуры ENDP

Семантика. По директиве PROC ассемблер заносит в ТИ информацию об имени процедуры (имя, смещение в сегменте кода). Директиву ENDP ассемблер пропускает.

Передача управления

CALL имя процедуры ; вызов процедуры

Действие команды: 1) адрес возврата → стек; 2) JMP имя процедуры

RET [KB] ; возврат из процедуры

Действие команды: 1) из стека вынимается адрес возврата;

2) если есть KB, то ESP := ESP + KB; 3) переход по адресу возврата.

Стандартные пролог и эпилог

Пролог (входные действия)

; настройка указателя кадра стека (при передаче параметров в стеке)

PUSH EBP

MOV EBP, ESP

; создание локальных переменных

SUB ESP, KB ; KB = общий размер лок. переменных

; сохранение используемых регистров

PUSH ...

Эпилог (выходные действия)

; восстановление используемых регистров

POP ...

; уничтожение локальных переменных

MOV ESP, EBP

; восстановление EBP

POP EBP

; возврат с удалением параметров из стека

RET [KB]

Паскалевская функция реализуется на ассемблере процедурой, результат возвращается в аккумуляторе (AL, AX, EAX в зависимости от типа результата). При передаче параметров в стеке ведущая часть кладёт параметры в стек в порядке, обратном тому, как они перечислены в заголовке процедуры. (Первый параметр процедуры лежит в стеке выше всех.)

Соответствие типов:

byte — char, boolean

dword — longint (знаковое число), longword (целое без знака), ссылка.

11.1 Используя регистр EAX как вспомогательный, описать через другие команды действие команды:

6) RET n

11.2 Описать процедуру PrLD(*n, p*: dword), которая печатает младшую цифру числа *n* (без знака) в системе счисления по основанию *p*. Передавать параметры в регистрах.

11.3 Для заголовка процедуры

```
procedure P(a: dword; var b: dword)
```

и секции данных

.DATA

x DD 1

y DD 7

выписать последовательность команд, реализующую вызов

- 1) $P(5, y)$; 2) $P(x, y)$; 3) $P(x+2, y)$; 4) $P(x, x)$.

а) Параметры передаются в регистрах: EAX — первый параметр, EBX — второй параметр.

б) Параметры передаются в стеке.

11.4 Заголовок процедуры имеет вид

procedure P (x: ptype);

Параметр передаётся в стеке. Пусть EBP установлен на базу стека стандартным образом. Выписать фрагмент процедуры P, который записывает в аккумулятор значение параметра x, если

- a) `ptype = char`
 - б) `ptype = integer` (соответствует word)
 - в) `ptype = longint` (соответствует dword)

11.5 Заголовок процедуры имеет вид

procedure S (**var** x: ptype);

Параметр передаётся в стеке. Пусть ЕВР установлен на базу стека стандартным образом. Выписать фрагмент процедуры S, который записывает в аккумулятор значение параметра x, если

- a) `ptype = char`
 - б) `ptype = integer` (соответствует word)
 - в) `ptype = longint` (соответствует dword)

11.6 Написать программу, которая вводит 4 положительных числа и определяет их наибольший общий делитель. В программе описать и использовать функцию нахождения наибольшего общего делителя

двух чисел с передачей параметров по значению в регистрах EAX и EBX; результат возвращать в EAX.

Следующие задачи решить в двух вариантах: а) с передачей параметров в регистрах; б) с передачей параметров в стеке. Требуется указывать интерфейс процедуры (т. е. заголовок процедуры на Паскале).

11.7 Реализовать на ассемблере процедуру

```
procedure A2(var x: int; y: int);
begin x:= x+y end;
```

Затем написать на ассемблере фрагмент программы, реализующий следующую ведущую часть

```
{a, b: int}
begin read(a, b); A2(a, b+1); write(a) end.
```

Считать, что типу int соответствует dword, число со знаком.

11.8 X DQ ?

Описать процедуру OUTB, которая печатает в виде двузначного (беззнакового) 16-ричного числа значение заданного байта. Используя эту процедуру, выписать фрагмент основной программы, печатающий значение переменной X в 16-ричном виде.

11.9 X DB 100 DUP (?) ; X[0..99]

Описать процедуру SETABS, которой передается адрес (смещение) некоторого знакового байта памяти и которая заменяет его значение на абсолютную величину. Предварительно выписать заголовок процедуры на Паскале. Используя эту процедуру, выписать фрагмент основной программы для решения следующей задачи: при $X[0] > X[27]$ заменить на абсолютную величину значение элемента $X[27]$, иначе — элемента $X[0]$.

11.10 A DD ? ; числа со знаком

B DD ?

C DD ?

Выписать фрагмент основной программы, в котором значения переменных A, B и C перераспределяются так, чтобы оказалось $A \geq B \geq C$. Для решения этой задачи описать процедуру MAXMIN(X,Y), которая перераспределяет значения X и Y так, чтобы большее из них оказалось в X, а меньшее — в Y. Предварительно написать заголовок процедуры и ведущую часть программы на Паскале.

11.11 Описать процедуру $\text{NGT}(x, y)$, которая реализует присваивание $x := -y$ для переменных типа word.

11.12 Описать процедуру NextDig , которая вычисляет цифру, следующую за цифрой d , считая что за '9' следует '0'. Написать полную программу, которая вводит по INCHAR цифру и печатает две следующие за ней цифры. Интерфейс процедуры:

a) **procedure** $\text{NextDig}(\text{var } d:\text{char})$;

процедура реализует оператор $d := \text{след. цифра}(d)$

b) **function** $\text{NextDig}(d:\text{char}): \text{char}$;

11.13 А DB 60 DUP (?) ; числа со знаком

В DB 101 DUP (?)

Описать процедуру OUTARR8, которой передается начальный адрес знакового байтового массива и число элементов в нём и которая печатает этот массив. Выписать заголовок процедуры на Паскале. Используя эту процедуру, выписать фрагмент основной программы для решения следующей задачи: если последний элемент массива А равен среднему элементу массива В, тогда напечатать массив А, иначе — массив В.

11.14 А DB 500 DUP (?) ; числа со знаком

В DD ?

Описать процедуру $\text{SUM}(X, N, S)$, которая присваивает параметру-двойному слову S сумму элементов массива X из N знаковых байтов. Выписать заголовок процедуры на Паскале. Выписать команды, соответствующие следующему обращению к процедуре: $\text{SUM}(A, 500, B)$.

11.15 Описать процедуру $F(X, N, P)$, определяющую, сколько элементов массива X из N байтов равно байту P , и возвращающую результат через регистр AL. Выписать заголовок процедуры на Паскале (это будет функция). Использовать эту процедуру для вычисления значения

$K := F(A, 70, F(B, 30, K))$

где А — массив из 70 байтов, В — массив из 30 байтов, а К — байтоваая переменная.

11.16 Описать логическую функцию $\text{SYM}(X, N)$, определяющую симметричность массива X из N элементов — двойных слов. Считать, что true кодируется числом 0FFh, false — числом 0.

11.17 Описать процедуру SHIFT23, которой передается начальный адрес некоторого массива из 100 байтов и которая за один просмотр этого массива циклически сдвигает его элементы на 23 позиции вперед (влево). В своей работе процедура должна использовать вспомогательный массив, отведя ему место в стеке.

В следующих задачах способ передачи параметров (в регистрах или в стеке) указан в условии.

11.18 В задаче нужно написать полную программу. Описать процедуру MD(n, d), реализующую присваивание $d :=$ наименьшая цифра (n). Используя процедуру, решить задачу: Ввести три числа. Напечатать эти числа в порядке возрастания их наименьших цифр. (Например, для чисел 325, 170, 3 нужно напечатать 170, 325, 3). Передача параметров в стеке.

11.19 Описать процедуру MAXLET для определения, какая из больших латинских букв чаще всего встречается в символьном массиве, начальный адрес и количество элементов которого передаются в стеке. Считать, что такая буква есть и она единственная. Ответ процедура должна вернуть через регистр AL. В своей работе процедура должна использовать вспомогательный массив, отведя ему место в стеке.

11.20 Описать рекурсивную процедуру OUTD, печатающую значение регистра EAX как беззнаковое число в системе счисления, основание которой (от 2 до 10) передается в регистре BL. (Обратить внимание на возможность переполнения в команде DIV.)

11.21 Описать процедуру, которая используя только операцию OUTCHAR, печатает содержимое регистра EAX, в виде знакового десятичного числа. Описать вспомогательную рекурсивную процедуру, печатающую положительное число.

11.22 Описать рекурсивную процедуру для решения следующей задачи. Пусть для обозначения «буквенных» 16-ричных цифр применяются прописные буквы A,...,F. Не используя команды умножения и деления

- a) вывести значение регистра EAX в виде 8-значного шестнадцатеричного числа;
- b) ввести шестнадцатеричное число и записать его в регистр EAX (считать, что число записано без ошибок, содержит от 1 до 8 цифр, за числом следует пробел).

11.23 Описать процедуру SUM, которой через регистр EBX передается начальный адрес, а через регистр ECX — число элементов некоторого массива, элементы которого (размером в двойное слово) являются адресами каких-то знаковых байтов в сегменте данных. Процедура должна найти сумму значений всех этих байтов и вернуть ответ через регистр EAX.

11.24 Описать процедуру ZERO от 20 параметров, которые передаются через стек и каждый из которых (размером в двойное слово) представляет собой адрес некоторой знаковой байтовой переменной из сегмента данных. Процедура должна обнулить те из этих переменных, значения которых положительны.

11.25 Написать программу решения задачи: Даная непустая последовательность символов, за которой следует точка. Напечатать эту последовательность в обратном порядке. Описать вспомогательную рекурсивную процедуру REVERSE, которая вводит последовательность и печатает её. Предварительно написать процедуру на паскале.

11.26 Написать программу решения задачи: Даная непустая последовательность чисел (со знаком), за последовательностью следует ноль. Напечатать эту последовательность в обратном порядке. Описать вспомогательную рекурсивную процедуру REVSEQ, которая вводит последовательность и печатает её. Предварительно написать процедуру на паскале.

11.27 Даная последовательность (возможно, пустая) ненулевых чисел, за последовательностью следует 0. Описать рекурсивную процедуру PR без параметров, которая вводит эти числа и выводит сначала все отрицательные числа, а затем — все положительные (в любом порядке).

11.28 Описать рекурсивную процедуру BITS1, которая подсчитывает количество двоичных единиц в регистре EBX и возвращает ответ через регистр AL.

12. Строковые команды

Обозначения операндов: источник (source) *src* — [ESI], получатель (destination) *dst* — [EDI]

Направление просмотра строки задаётся флагом направления DF: DF = 0, просмотр от начала к концу; DF = 1, просмотр от конца строки к началу.

Установка DF

```
CLD      ; DF:=0  
STD      ; DF:=1
```

Строковые примитивы

Мнемокод образуется из названия примитива и типа данных *(КОП)\(тип)*, например, LODSD

(тип) ::= B | W | D

Примитивы работают с аккумулятором (обозн. acc) в зависимости от типа: AL, AX, EAX.

Примитив выполняет 1) заданную операцию и 2) сдвигает указатель операнда на очередной элемент строки (при этом флаги не затрагиваются)

Примитивы

Пересылки

LODS	; acc := src
STOS	; acc → dst
MOVS	; dst := src

Сравнения

SCAS	; acc-dst, установка флагов
CMP	; src-dst, установка флагов

Префиксы повторения

Пересылки

REP примитив

Семантика:

- 1) ECX=0? Да — завершение работы команды
- 2) выполнение примитива
- 3) ECX := ECX-1

Сравнения

REPE примитив ; повторять, пока равно
REPNE примитив ; повторять, пока не равно

Семантика:

- 1) ECX=0? Да — завершение работы команды
- 2) выполнение примитива
- 3) ECX := ECX-1
- 4) условие повторения истинно? Да — переход к п.1, нет — завершение работы команды

12.1 Используя, если надо, регистр EBX как вспомогательный и считая, что флаг направления DF равен 0, описать через другие (не строковые) команды действие команды:

- а) MOVSD б) CMPSB в) SCASB г) LODSD д) STOSB е) REP MOVSD

12.2 S DB 'abcde',0

T DB 'abxyz'

Определить значения регистров ECX и BL и флага ZF после выполнения следующей группы команд:

- | | | | |
|--------------|--------------|--------------|--------------|
| а) CLD | б) STD | в) CLD | г) CLD |
| LEA ESI,S | LEA ESI,S+4 | LEA EDI,S | LEA EDI,S |
| LEA EDI,T | LEA EDI,T+4 | MOV ECX,5 | MOV ECX,5 |
| MOV ECX,5 | MOV ECX,5 | MOV AL,'e' | MOV AL,'f' |
| REPNE CMPSB | REPNE CMPSB | REPNE SCASB | REPNE SCASB |
| MOV BL,[ESI] | MOV BL,[ESI] | MOV BL,[EDI] | MOV BL,[EDI] |

12.3 X DD 1, 2, 3, 4, 5, 6, 7, 8, 9

Y DD 9 DUP (?)

Определить значения указанных ячеек памяти, регистров и флагов, которые получатся после выполнения следующей группы команд:

- | | |
|---------------------|---------------------|
| а) CLD | б) STD |
| MOV ECX,5 | MOV ECX,7 |
| MOV ESI, offset X | MOV EDI, offset X |
| MOV EDI, offset Y | MOV EAX, 3 |
| REP MOVSD | REPNE SCASD |
| ECX = ? | ECX = ? |
| dword ptr [ESI] = ? | dword ptr [EDI] = ? |
| ZF = ? | ZF = ? |
| в) CLD | г) CLD |
| MOV ECX,7 | MOV ECX,7 |
| MOV EDI, offset X | MOV EDI, offset X |
| MOV EAX, 8 | MOV EAX, 1 |
| REPNE SCASD | REPE SCASD |
| ECX = ? | ECX = ? |
| dword ptr [EDI] = ? | dword ptr [EDI] = ? |
| ZF = ? | ZF = ? |

В следующих задачах использовать строковые команды с префиксами повторения.

12.4 N = 100

S DB N DUP (?) ; S[0..N-1] of char
T DB N DUP (?) ; T[0..N-1] of char

- а) Заполнить строку S символами 'd'.
- б) Считая, что строка S заполнена, реализовать присваивание T:=S
- в) Заменить последние 10 символов строки S на первые 10 символов строки T.

12.5 N = 100

X DD N DUP (?)
Y DD N DUP (?)

- а) Выписать последовательность команд, инициализирующую все элементы массива X числом (-10).
- б) Скопировать массив X в массив Y.

12.6 N = 150

S DB N DUP (?)

- а) Сдвинуть элементы строки S на три позиции влево, заполнив последние элементы пробелами.
- б) Сдвинуть элементы строки S на три позиции вправо, заполнив первые элементы пробелами.
- в) Циклически сдвинуть элементы строки S на две позиции влево.
- г) Циклически сдвинуть элементы строки S на две позиции вправо.
- д) Циклически сдвинуть элементы строки S на 4 позиции влево.
- е) Циклически сдвинуть элементы строки S на 4 позиции вправо.
- ж) Сдвинуть элементы строки S на 15 позиций влево, заполнив последние элементы пробелами. Описать и использовать вспомогательный массив.
- з) Сдвинуть элементы строки S на 15 позиций вправо, заполнив первые 15 элементов пробелами. Описать и использовать вспомогательный массив.

12.7 N = 70

X DD N DUP(?) ; X[1..N]

- а) Проверить, входит ли в массив X число 5. Результат записать в BL (1 – входит, 0 – не входит).
- б) Если в массиве X есть число 10, напечатать индекс его первого вхождения, в противном случае напечатать -1.

12.8 N = 100

S DB N DUP(?) ; строка символов

- а) Определить, со скольких пробелов начинается строка S, записать ответ в регистр CL.
- б) Определить, сколькими пробелами заканчивается строка S, записать ответ в регистр CL.

12.9 N = 100

T DB N DUP(?) ; T[0..N-1] of char

- а) Заменить в строке T первое вхождение символа '.' на символ '!'.
- б) Заменить в строке T последнее вхождение буквы 'n' на букву 'N'.
- в) Найти индекс (от 0 до N-1) первого вхождения буквы 'Q' в строку T и записать ответ в регистр EDI; если эта буква не входит в T, то в EDI записать -1.

12.10 N = 100

S DB N DUP(?) ; S[0..N-1] of char

Определить, равны ли левая и правая половины строки S, и записать ответ 1 (равны) или 0 в регистр AL. Если строка нечётной длины, средний символ не принадлежит ни левой, ни правой половине.

12.11 N = 100

S DB N DUP(?)

- а) Используя строковые команды с префиксами повторения, заменить в S все символы '*' на символы '?'.
- б) Используя строковые команды с префиксами повторения, записать в регистр BH число вхождений символа '*' в строку S.

12.12 N = 200

X DD N DUP(?)

- а) Заменить в массиве X все элементы, равные 50, на число 100. Использовать строковые команды с префиксами повторения.
- б) Записать в регистр BH количество элементов, равных 50. Использовать строковые команды с префиксами повторения.

12.13 Пусть в стеке записано 200 двойных слов и пусть в сегменте данных описан массив X из 200 двойных слов. Не используя команду POP и не изменяя значение регистра ESP, скопировать содержимое стека в массив X («верхнее» двойное слово стека должно быть записано в начальный элемент массива). Использовать строковую команду с префиксом повторения.

12.14 N = 100

A DD N DUP (?)
B DD N DUP (?)

Описать процедуру COPY(X,Y,N), копирующую массив X из N двойных слов в массив Y. Параметры X и Y передаются по ссылке, N — по значению; параметры передать в стеке. В процедуре использовать строковую команду с префиксом повторения. Выписать фрагмент программы для вызова процедуры COPY(A,B,20).

12.15 N = 100

A DD N DUP (?)

Описать процедуру ZERO(X,N), проверяющую, входит ли в массив X из N двойных слов число 0 и возвращает результат —1, если входит, и 0 в противном случае. Параметр X передаётся по ссылке, N — по значению; параметры передать в стеке, результат вернуть в AL. В процедуре использовать строковую команду с префиксом повторения. Выписать фрагмент программы для вызова ZERO(A,20).

12.16 N = 20

S DB N DUP(?) ; S[0..N-1] of char
T DB N DUP(?) ; T[0..N-1] of char

Описать логическую функцию LESS(A,B,N) лексикографического сравнения строк. Параметры: A, B — строки (массивы) длины N, состоящие из строчных латинских букв и пробелов. Если A<B, функция возвращает true (0FFh), в противном случае — 0 в регистре AL. Параметры передать в стеке, A, B — по ссылке, N — по значению. В процедуре использовать строковую команду с префиксом повторения. Выписать фрагмент программы для вызова LESS(S,T,N).

12.17 S DB 256 DUP (?) ; длина(S)≤255

T DB 101 DUP (?) ; длина(T)≤100

Рассматривая S и T как символьные строки переменной длины с текущей длиной в начальном байте и используя строковые команды с префиксом повторения, выписать фрагмент программы для решения следующей задачи.

- а) В строке S оставить только первые 10 символов. Если длина строки меньше 10, строку не менять.
- б) Сделать значением S строку из 50 пробелов.
- в) Удалить все пробелы в конце строки S.
- г) Удалить все пробелы в начале строки S.

- д) Если в строке S от 10 до 40 символов, то продублировать 10-й символ.
- е) Удалить из строки S все пробелы.
- ж) Сравнить строки S и T и записать ответ 1 ($S > T$), 0 ($S = T$) или $-1 (S < T)$ в регистр AL.
- з) Определить, является ли строка T подстрокой строки S, и записать ответ 1 (является) или 0 в регистр AL.

12.18 N = 100000

S DB N DUP (?)

T DB N DUP (?)

LenS DD ?

Рассматривая S и T как символьные строки переменной длины с признаком конца (строка заканчивается байтом 0) и используя строковые команды с префиксом повторения, выписать фрагмент программы для решения следующей задачи.

- а) Записать в переменную LenS длину строки S.
- б) В строке S оставить только первые 10 символов. Если длина строки меньше 10, строку не менять.
- в) Сделать значением S строку из 50 пробелов.
- г) Удалить все пробелы в конце строки S.
- д) Удалить все пробелы в начале строки S.
- е) Если в строке S от 10 до 40 символов, то продублировать 10-й символ.
- ж) Удалить из строки S все пробелы.
- з) Сравнить строки S и T и записать ответ 1 ($S > T$), 0 ($S = T$) или $-1 (S < T)$ в регистр AL.

13. Макросредства

Обозначения:

символы <> являются частью конструкции языка (не метасимволы),
 KB — константное выражение.

В предложениях, относящихся к макросредствам, в выражениях ссылки вперед не допускаются.

Условная макрогенерация

Синтаксис

```
IFxxx операнды
    true-часть
ELSE | ELSEIFxxx операнды
    false-часть
ENDIF
```

Здесь *true-часть* и *false-часть* — последовательность любых предложений ассемблера. Количество и вид operandов зависят от вида директивы.

Семантика

Макрогенератор (МГ) вычисляет условие; если условие истинно, в текст программы вставляется *true-часть*, в противном случае в текст включается *false-часть*.

Каждой директиве IFxxx соответствует директива ELSEIFxxx.

Виды директив IF (ELSEIF)

Директива

IF KB	<i>KB = истина</i>
IFE KB	<i>KB = 0</i>
IFB <строка>	<i>строка пустая</i>
IFNB <строка>	<i>строка не пустая</i>
IFIDN <строка1>,<строка2>	<i>строка1 = строка2</i>
IFIDNI <строка1>,<строка2>	<i>строка1 = строка2 без учёта регистра</i>
IFDIF <строка1>,<строка2>	<i>строка1 ≠ строка2</i>
IFDIFI <строка1>,<строка2>	<i>строка1 ≠ строка2 без учёта регистра</i>

Проверяемое условие

Блоки повторения

Синтаксис

REPEAT KB	FOR форм.пар., <список>	FORC форм.пар., <строка>
тело	тело	тело
ENDM	ENDM	ENDM

Семантика

МГ вставляет тело блока в текст программы необходимое количество раз, при этом формальные параметры в теле блока заменяются фактическими параметрами.

Макросы

Обозначения: [], {} — метасимволы

Описание макрона

```
имя_макрона MACRO [форм.пар. {, форм.пар.}]
    тело
ENDM
```

Макровызов

имя_макрона список

Настройка тела макрона и блока повторения

Распознавание формальных параметров

- 1) Вне строк в " ". Если параметр сливаются с лексемой, то параметр нужно отделить & (например, *Lex¶m*).
- 2) Внутри " ". До и после параметра указывают & ("...¶m&...").

Формальный параметр вместе с && заменяется на фактический параметр.

Вхождение формальных параметров не распознаётся в комментариях (т.е. после символа ;).

Фактические параметры

список ::= [*оп* {, *оп*}]

Вид *оп*:

1. строка без , и ;
2. <строка>
3. % выражение
4. пусто

МГ вместо *оп* подставляет

- | |
|--------|
| строка |
| строка |
| число |
| ничего |

Если в *оп* нужно указать специальный символ, перед символом ставят !.

Если нужно указать ! в *оп*, его пишут два раза. Знак ! действует и в блоке повторения FORC *p*, <строка>.

Локальные имена

LOCAL имя {, имя}

Директива LOCAL должна быть первым предложением тела макрона или блока повторения.

В макрорасширении МГ заменяет локальное имя уникальным сгенерированным именем ??nnnn, где nnnn — значение счётчика локальных имён.

Комментарии

;; комментарий
предложение ;; комментарий

Удаляется из макрорасширения.

13.1 Заменить последовательность строк программы (A, B, C, D, E — переменные типа dword)

```
MOV A, 1  
MOV B, 1  
MOV C, 1  
MOV D, 1  
MOV E, 1
```

на фрагмент с использованием блока повторения

- a) FOR b) FORC

13.2 Реализовать в виде блока повторения

- a) FOR b) FORC

следующий фрагмент

```
ADD EAX, 1  
ADD EAX, 2  
ADD EAX, 3  
ADD EAX, 4  
ADD EAX, 5  
ADD EAX, 6
```

13.3 Описать с помощью подходящего блока повторения решение следующей задачи:

- а) записать в регистр AH сумму чисел из регистров AL, BL, CL и DH;
- б) обнулить переменные A, B и C типа qword;
- в) используя из операций вывода только операцию OUTCHAR, напечатать (включая кавычки) текст "A+B=B+A";
- г) зарезервировать (с помощью директивы DB) место в памяти для 40 байтов, присвоив им в качестве начальных значений первые 40 нечётных чисел (1, 3, 5, ..., 79).

13.4 Пусть A, B, C, D, E, F — переменные (байты или двойные слова). Написать блок повторения, который увеличивает на 1 значения всех байтовых переменных и на 2 — значения остальных переменных.

13.5 Пусть A, B, C, D, E, F — переменные (байты или двойные слова). Написать блок повторения, который записывает число -1 в байтовые переменные. Другие переменные не менять.

13.6 Выписать текст окончательной программы, который построит макрогенератор по следующему фрагменту исходной программы:

a) FOR p,<INC A,JE L>
p
ENDM

6) FOR W,<SW,< A,2>>
MOV&W
ENDM

b) N EQU 5
FOR OP,<N,N+1,%N+1>
ADD AX,OP
ENDM

г) FOR p,<K,LL,M>
p EQU <B p&p>
DB 'p'
DB 'B &p&p'
ENDM

д) FOR T,<AB,C>
FORC U,<T>
DW U,T&U,T&U
ENDM
DW U,T&U
ENDM

е) FORC sym,<",%2+1>
ADD AL,'&sym&'; код sym
ENDM

13.7 Описать в виде макросов операции над четырёхными словами.

Параметры X, Y и Z — переменные типа qword, L — метка.

- а) QMOV X,Y ; X:=Y
- б) QADD X,Y,Z ; X:=Y+Z
- в) QSUB X,Y,Z ; X:=Y-Z
- г) QJNE X,Y,L ; при X ≠ Y перейти на L

13.8 Описать в виде макроса DEF X,T,N,V определение массива X из N величин V, тип которых задается параметром T: значение параметра T=B соответствует типу byte, значения W, D, Q — типам word, dword, qword. Выписать текст, который построит макрогенератор по следующему фрагменту исходной программы:

```
K EQU 4
DEF C,B,10,'*'
DEF W,W,K+1,<TYPE C>
DEF ,D,%K+1,%(TYPE W)
DEF A,B,1,<1,2,3>
```

13.9 Описать в виде макроса указанное действие над знаковыми числами размером в байт. Регистры, кроме указанных в условии, не портить.

- а) ABS R,X ; R:=abs(X), где R — регистр, X — переменная
- б) SUM X ; AX:=сумма элементов массива X
- в) MAX X ; AL:=максимум элементов массива X

13.10 Описать в виде макроса указанную команду, предполагая, что её нет в MASM (в качестве вспомогательных можно использовать регистры EAX и EBX):

- а) PUSH X (X — переменная размером в двойное слово);
- б) POP X (X — переменная размером в двойное слово);
- в) CALL P (P — имя процедуры);
- г) RET N (N — константа);
- д) LOOP L (L — метка).

13.11 Описать в виде макроса MAX2 M,X,Y вычисление $M=\max(X,Y)$ и на его основе описать макрос MAX3 M,X,Y,Z для вычисления $M=\max(X,Y,Z)$, где M, X, Y и Z — знаковые байтовые переменные.

Выписать макрорасширение для макрокоманды

MAX3 A,<byte ptr B>,C+1,D

13.12 Пусть RUN — константа, которая своим значением 1 или 0 указывает режим текущего прогона программы: счёт или отладка. Используя средства условного ассемблирования, выписать фрагмент программы, в котором находится наибольший общий делитель двух положительных чисел с записью его в регистр AX, при условии, что в режиме счёта эти два числа должны вводиться, а в режиме отладки эти числа равны 45 и 30.

13.13 Пусть K — числовая константа с положительным значением, а α , β и γ — некоторые группы предложений. Используя средства условного ассемблирования, выписать фрагмент исходной программы, в котором α , β и γ указаны лишь по разу и по которому в зависимости от значения K формируются следующие варианты окончательной программы:

$$\begin{array}{lll} \text{при } K = 1: & \alpha & \text{при } K = 2: & \alpha \\ & \beta & & \gamma \\ & & & \gamma \\ & & & \vdots \\ & & & \gamma \end{array} \quad \left. \begin{array}{c} \gamma \\ \vdots \\ \gamma \end{array} \right\} K \text{ раз}$$

13.14 Имеются следующие описания макроса M X, который должен уменьшить значение переменной X (байт, слово или двойное слово) на 5, если X — это байт, или на 12, если X — слово или двойное слово:

a) M MACRO X LOCAL L2,L CMP type X,byte JNE L2 SUB X,5 JMP L L2: SUB X,12 L: ENDM	б) M MACRO X LOCAL L2,L MOV AL,type X CMP AL,byte JNE L2 SUB X,5 JMP L L2: SUB X,12 L: ENDM	в) M MACRO X IF type X EQ byte SUB X,5 ELSE SUB X,12 ENDIF ENDM
---	--	---

Какое из этих описаний неправильно (и почему) и какое из двух правильных описаний лучше другого (и почему)?

13.15 Описать макрос SHIFT X,K , где X — переменная, двойное слово, K — явно заданное число. Макрос сдвигает значение X на $|K|$ разрядов вправо при $K > 0$, влево при $K < 0$ и не меняет значение X при $K = 0$.

Выписать макрорасширения для макрокоманд

a) SHIFT B,5; б) SHIFT A,-1 и в) SHIFT C,2-2.

13.16 Описать в виде макроса IF0 X,L (X — переменная размером в байт, слово, двойное слово, или учетверённое слово, L — метка) переход на метку L в том случае, когда значение переменной X равно 0. Выписать макрорасширения для макрокоманд IF0 B,L1 и IF0 Q,L2 при условии, что B — переменная типа byte, а Q — типа qword.

13.17 Описать в виде макроса SIGN X (X — знаковая переменная размером в байт, слово, двойное слово или учетверённое слово) операцию засылки в регистр AL числа 1 при $X > 0$, числа 0 при $X = 0$ и числа -1 при $X < 0$.

Выписать макрорасширения для макрокоманд SIGN W и SIGN Q при условии, что W — переменная размером в слово, а D — размером в учетверённое слово.

13.18 Описать в виде макроса ZERO X,T (X — имя переменной-массива, T — это FIRST или LAST) обнуление либо начального (при T=FIRST), либо последнего (при T=LAST) элемента массива X.

Выписать макрорасширение для макрокоманды ZERO A,LAST.

13.19 Описать в виде макроса `NULL RS` обнуление регистров общего назначения. Здесь `RS` — это строка вида `<R1, R2, ..., Rk>`, R_i — регистр общего назначения, $k \geq 0$. Выписать макрорасширения для макрокоманд `NULL <AL,BX,ESI>` и `NULL <>`.

13.20 Описать в виде рекурсивного макроса `NULL r1,r2,r3,r4,r5,r6,r7,r8` (где r_i — имена регистров общего назначения) обнуление регистров-параметров. Выписать макрорасширения для макрокоманд `NULL ESI` и `NULL EAX,EBX,ECX,EDX`.

13.21 Описать макрос `PRINT1 LV`. Макрос должен напечатать в виде чисел без знака значение переменных — двойных слов из списка `LV`. Здесь `LV` — строка вида `<v1, v2, v3, ..., vk>`, v_i — имя переменной, $k \geq 0$. Выписать макрорасширение для макрокоманды `PRINT1 <A, X, B, Y, Z>` (где `A, B` — байты, `X, Y, Z` — двойные слова) и для макрокоманды `PRINT1 <>`.

13.22 Описать рекурсивный макрос `PRINT2 v1,v2,v3,v4,v5,v6`. Здесь v_i — имя переменной. Макрос должен напечатать в виде чисел без знака значение переменных — двойных слов из списка параметров. Выписать макрорасширение для макрокоманды `PRINT2 A, X, B` (где `A, B` — байты, `X` — двойное слово) и для макрокоманды `PRINT2`.

13.23 Описать макрос `LAST p1,p2,p3,p4,p5,p6,p7`, который печатает в виде числа без знака последний переданный фактический параметр. Фактические параметры — константы.

Выписать макрорасширение для макровызовов `LAST 1,2,3,4,5,6,7` и `LAST 5,4,3`.

13.24 Описать макрос `COUNT p1,p2,p3,p4,p5,p6,p7`, который печатает количество переданных фактических параметров.

Выписать макрорасширение для макровызовов `COUNT a,b,3,4,d,6,7` и `COUNT 5`.

13.25 Описать макрос `VOLUME p1,p2,p3,p4,p5,p6,p7`, который подсчитывает и печатает общее количество байтов, занимаемое переданными параметрами в сегменте данных. Параметры — имена переменных или константы. Учесть, что константы не хранятся в ОП.

Выписать макрорасширение для макровызова `VOLUME a,30,N,x,d,6` (здесь `a` — переменная типа `byte`, `N` — константа со значением 100, `x` и `d` — переменные типа `word` и `dword` соответственно).

13.26 Описать макрос PRDWRD $p_1, p_2, p_3, p_4, p_5, p_6, p_7$ (p_i — имена переменных), который печатает (как числа со знаком) значения тех параметров, которые являются двойными словами. Выписать макрорасширение для макровызовов PRDWRD B, W, D и PRDWRD D, D, D, D, D (B, W и D — имена переменных типа byte, word и dword соответственно).

13.27 Описать макрос PARMs p_1, p_2, p_3, p_4, p_5 . Макрос будет использоваться в теле процедуры, p_i — имена параметров процедуры. Считая, что EBP стандартно настроен на кадр стека, макрос должен назначать именам параметров их адреса в стеке. Выписать макрорасширение для макрокоманд PARMs X, Y, Z и PARMs X.

- а) Считать, что p_1 — верхний параметр в стеке.
- б) Считать, что p_1 — нижний параметр в стеке.

13.28 Описать макрос SUM R, где R — имя одного из 32-разрядных регистров общего назначения, для записи в R суммы значений всех остальных таких регистров.

13.29 Описать в виде макроса OUTF без параметров вывод текущих значений флагов CF, OF, SF и ZF в виде четверки из 0 и 1, причём значения всех флагов и используемых макросом регистров должны быть сохранены. Воспользоваться командами PUSHFD и POPFD. Учитывать, что в регистре флагов EFlags указанным флагам соответствуют биты со следующими номерами (нумерация битов справа налево от 0): CF — 0, OF — 11, SF — 7, ZF — 6.

13.30 Описать в виде макроса SL RS,OP (RS — это $\langle R_1, R_2, \dots, R_k \rangle$, где все R_i — 32-разрядные регистры общего назначения, $k > 0$; OP — это SAVE или LOAD) запись в стек (при OP=SAVE) или восстановление из стека (при OP=LOAD) регистров R_i . Выписать макрорасширение для макрокоманды SL $\langle EAX, ECX, EDI \rangle$, LOAD.

13.31 Описать в виде макроса SUM X (X — это $\langle X_1, X_2, \dots, X_k \rangle$, где X_i — знаковые байтовые переменные, $k > 0$) вычисление суммы значений X_i и записи её в регистр BX. Выписать макрорасширение для макрокоманды SUM $\langle A, B, C \rangle$.

13.32 Описать в виде макроса MAX X (X — это $\langle X_1, X_2, \dots, X_k \rangle$, где X_i — знаковые байтовые переменные, $k > 0$) вычисление максимума X_i и записи его в регистр AL. (Обратить особое внимание на метки, которые

будут появляться в макрорасширениях.) Выписать макрорасширение для макрокоманды MAX <A,B,A>.

13.33 Написать полную программу, которая вводит три числа H, M и S и проверяет, удовлетворяют ли они следующим условиям: $0 \leq H \leq 23$, $0 \leq M, S \leq 59$. Если нет, программа должна выдать сообщение об ошибке, а иначе, трактуя эти числа как час (H), минута (M) и секунда (S) некоторого момента суток, должна напечатать время суток, на 1 секунду большее (с учетом смены суток).

Определить и использовать в этой программе два макроса, один из которых проверяет условие $a \leq X \leq b$, а другой — увеличивает X на 1 и, если $X > b$, обнуляет X.

13.34 X DB 100 DUP(?) ; X[0..99]

Описать процедуру SetAbs(var z:byte), которая заменяет значение параметра на его абсолютную величину; параметр передаётся в стеке по ссылке. Описать макрос для вызова этой процедуры. Используя макрос, написать фрагмент ведущей части: при $X[0] > X[27]$ заменить на абсолютную величину значение элемента $X[27]$, иначе заменить на абсолютную величину значение элемента $X[0]$.

13.35 A DD ? ; числа со знаком

B DD ?

C DD ?

Описать процедуру MaxMin(var x,y: dword), которая перераспределяет значения x и y так, чтобы большее из них оказалось в x, а меньшее — в y; параметры передаются в стеке по ссылке. Описать макрос для вызова этой процедуры. Используя макрос, написать фрагмент ведущей части: перераспределить значения переменных A, B и C так, чтобы оказалось $A \geq B \geq C$.

13.36 Предположим, что имеется процедура P от двух параметров, которые передаются по значению через регистры AX и BX. Описать в виде макроса CALL_P X,Y команды обращения к этой процедуре, которые должны сохранять регистры AX и BX и которые должны корректно работать, когда в качестве X и Y указаны AX и/или BX.

Выписать макрорасширения для макрокоманд:

- а) CALL_P Q,2 б) CALL_P AX,BX в) CALL_P BX,5 г) CALL_P BX,AX

13.37 Дано макроопределение. Выписать макрорасширение для указанного макровызова:

а) Макроопределение	б) Макроопределение	в) Макроопределение
MA MACRO K	MB MACRO K	MV MACRO K
JMP M1	MOV AH,K	A EQU K
MOV AX,K	IFIDN <AH>,<2>	IFDIF <A>,<K>
M&K:ADD AX,K	ADD AH,K	ADD DX,A&K
ENDM	ENDIF	ENDIF
...	ENDM	ENDM
Макровызов
MA 1	Макровызов	Макровызов
	MB 2	MV 3

г) Макроопределение	д) Макроопределение	е) Макроопределение
MG MACRO K	MD MACRO K	ME MACRO K
IFDIF <K+1>,<5>	IF K+1 EQ 6	MOV AH,0
MOV AX,K	X DB '&K&+1'	FORC C,<K-1>
ENDIF	ENDIF	ADD AH,'&C&'
MOV BX,K&K	Y&K DW K+1	ENDM
ENDM	ENDM	ENDM
...
Макровызов	Макровызов	Макровызов
MG 4	MD 5	ME 6

13.38 Дано макроопределение. Выписать макрорасширение для указанного макровызова:

а) Макроопределение	б) Макроопределение	в) Макроопределение
m1 MACRO p,c	m2 MACRO ns	m3 MACRO k, L
k EQU 2	FOR n, <ns>	IF k+1 NE L
IF 2*p GE 3	IFDIF <n-1>,<2>	a&L DB "&L&L"
a2 DW c&k	ADD AX, n	ENDIF
ELSE	ENDIF	IFDIF <k-2>,<0>
c&c DB 'c&c&'	ENDM	b&k DW k DUP (k&k)
ENDIF	ENDM	ENDIF
ENDM	...	ENDM
...	Макровызов	...
Макровызов	m2 <1,2,3>	Макровызов
m1 k-1,A		m3 2,5

Ответы

1. Директивы определения данных

1.9 Таблица имён

имя	значение	тип
a	0	word
x	2	byte
y	5	dword

Объектный код. Содержимое ячеек выписано в 16-ричной системе.

смещение содержание

0	23
	01
2	A1
	03
	04
5	03
	00
	00
	00

1.17 Ошибка 1: число 350 не умещается в байт

Ошибка 2: значение выражения 'A'*2¹⁶+ 'B'*2⁸+ 'C' не умещается в слово

Ошибка 3: адрес (32 разряда) не умещается в байт

Ошибка 4: повторное описание имени E.

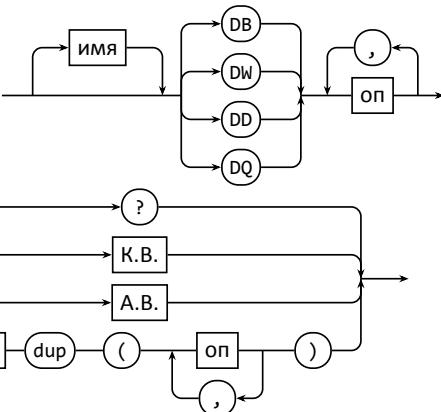
В байте с адресом 9 содержится число 44h.

1.18

Выражение	конст/адр	значение	тип	смещение
а) N	конст	10		
б) V	адр		dword	7
в) Y-1	адр		word	4
г) Y-X	конст	5		
д) Y+N	адр		word	15
е) N-Y	ошибка			
ж) 2*N+1	конст	21		
з) 2*X+1	ошибка			
и) V+Y-V	ошибка			
к) V+(Y-V)	адр		dword	5
л) (V-Y)/2	конст	1		
м) X-Y/2	ошибка			
н) N shl 2	конст	40		
о) N shr 1	конст	5		
п) 0FFFF FFFFh+2+Y	адр		word	6
р) V-X+V	адр		dword	14
с) 'A'+N/3	конст	код 'D'		
т) 'Z' - 'Y'	конст	1		
у) '9' - '8'	конст	1		
ф) X+'B' - 'A'	адр		byte	1

1.20

директива
определения
данных



1.22 A DB M DUP (10 DUP (0), N-10 DUP (-1))

1.25 При вычислении выражения $0FFFFh+1$ получится $10000h$, что больше, чем слово (тип X). При вычислении выражения $0FFFF FFFFh+1$ результат будет 0, т. к. выражения вычисляются по модулю 2^{32} ; число 0 уместится в слово, ошибки нет.

1.27 DB (N-1)/8 DUP (0FFh), (0FFh shl (8-N mod 8)) and 0FFh

2. Команды пересылок. Оператор ptr

2.2 Ошибка 1: разный размер операндов

Ошибка 2: два операнда из памяти

Ошибка 3: неверный мнемокод

Ошибка 4: первый операнд должен быть регистром, а не переменной; второй операнд должен быть адресным выражением, а не регистром

Ошибка 5: размер второго операнда равен размеру первого (должен быть меньше)

Ошибка 6: второй операнд не может быть константным выражением

2.3 Неправильные: а, г, д, л, м, н, о, р, с, ф, х, ч.

2.4 AL=56h; EBX=34567812h; ECX=0FFFFFFFABh; EDX=0ABh

2.6 a) MOV A, 80h ; 1000 0000b
 MOV B, 8000h
 MOV C, 8000 0000h
 MOV dword ptr D, 0
 MOV dword ptr D+4, 8000 0000h

6) MOV A, 7Fh ; 0111 1111b
 MOV B, 7FFFh
 MOV C, 7FFF FFFFh
 MOV dword ptr D, 0FFFF FFFFh
 MOV dword ptr D+4, 7FFF FFFFh

2.7. 6) MOV AX,word ptr A+1
 XCHG AX,word ptr A
 MOV A+2, AL

2.12 a) MOV dword ptr Q, 125
 MOV dword ptr Q+4, 06) MOV dword ptr Q, -125
 MOV dword ptr Q+4, 0FFFF FFFFh

2.14 1 способ

MOV AL, A
 MOV AH, 0
 MOV Y, AX

2 способ

MOVZX AX, A
 MOV Y, AX

3. Арифметические команды

3.6. Вспомним материал по УМ-2: при выполнении команд условного перехода анализируются значения флагов; условие $OF \neq SF$ соответствует переходу по меньше для чисел со знаком. Следовательно, $OF \neq SF$ получится для всех $CL < 25$. Условие $OF = SF$ есть отрицание условия $OF \neq SF$. **ОТВЕТ:** а) $CL \in [-128, 24]$; б) $CL \in [25, 127]$

3.9. **5** MOV AL,X
ADD AL,2*A+6
ADD X,AL

3.12. В Вычислим результат в паре EDX:EAX, затем запишем его в Z. Реализуем вычитание 0-Y в столбик, сначала вычитаем младшие разряды, потом — старшие, учитывая заём. Обращаем внимание на перевёрнутое представление чисел.

```
MOV EAX,0
MOV EDX,0
SUB EAX, dword ptr Y
SBB EDX, dword ptr Y+4
MOV dword ptr Y, EAX
MOV dword ptr Y+4, EDX
```

3.16. а частное от деления Y на 10 не умещается в байт

б неверное делимое у второй команды деления: $EDX:EAX = 1*2^{32} + 5432$, а должно быть $EDX:EAX = 5432$

3.24. Организуем вычисления по схеме Горнера: $(A*X+B)*X+C$

```
MOV BL,X
MOV BH,0      ; BX=X
MOV AL,A
MUL BL       ; результат - слово AX
ADD AL,B
ADC AH,0     ; AX+B, прибавили к слову байт без знака
MUL BX       ; результат DX:AX
ADD AL,C
ADC AH,0     ; AX+B, прибавили к слову байт без знака
ADC DX,0     ; DX:AX+C
MOV word ptr Y, AX,
MOV word ptr Y+2, DX
```

3.31. Неверные команды: б, г, е, з, и, л.

4. Команды ввода и вывода. Структура программы

4.16. При сложении двойных слов $a + b$ результат может превзойти двойное слово, поэтому будем вычислять сумму в четверном слове EDX:EAX.

```
INCLUDE settings.inc
INCLUDE io2020.inc
.STACK 4096
.DATA
    a DD ?
    b DD ?
.CODE
start:  ININT a
        ININT b
        MOV EAX, a
        CDQ
        MOV EBX, EAX
        MOV ECX, EDX    ; ECX:EBX = a
        MOV EAX, b
        CDQ              ; EDX:EAX = b
```

```

ADD EAX, EBX
ADC EDX, ECX    ; EDX:EAX = b+a
MOV EBX, 17
IDIV EBX
OUTI EDX
EXIT
END start

```

- 4.18. а) При делении на байт под частное и под остаток отводятся байты. Следовательно, наибольшее возможное частное $0FFh$. Значит, максимально возможное делимое – это $0FFh * 10 + 9$. Таким образом, приходим к решению:

```

OUTU 0FFh*10+9
OUTU 0FFh ; частное
OUTU 9      ; остаток

```

- б) При делении на байт под частное и под остаток отводятся байты. Наибольшее по модулю возможное частное со знаком $80h$. Максимально возможное делимое со знаком – это $80h * 10 - 9$. Ассемблер вычисляет выражения в 32 разрядах, значит, необходимо расширить число $80h$ (как число со знаком) – получим $0FFF FF80h$. Заметим, что в процессоре intel при делении отрицательного числа на положительное получается отрицательный остаток (не так, как в Паскале).

```

OUTI 0FFFF FF80h*10-9
OUTI 0FFFF FF80h ; частное
OUTI -9          ; остаток

```

- 4.19. а) Адрес начала подстроки, которую нужно напечатать, равен $offset A + 26 - m$.

```

INCLUDE settings.inc
INCLUDE io2020.inc
.STACK 4096

m = ... ; 1 ≤ m ≤ 26

.DATA
A DB 'abcdefghijklmnopqrstuvwxyz', 0

.CODE
start: OUTSTR offset A+26-m
        EXIT
END start

```

- б) Нужно напечатать буквы от 'а' до m -й буквы алфавита; для своевременного окончания печати запишем в позицию следующей буквы число 0. Фрагмент решения:

```

.CODE
start: MOV A+m-1, 0
        OUTSTR offset A
        EXIT
END start

```

- 4.20. б) Адрес начала подстроки, которую нужно напечатать, равен $offset A + 26 - n$. Сначала введём число n в регистр, затем вычислим нужное значение.

```

INCLUDE settings.inc
INCLUDE io2020.inc
.STACK 4096

.DATA
A DB 'abcdefghijklmnopqrstuvwxyz', 0

```

```
.CODE
start: OUTCHAR '>'      ;приглашение для ввода
        ININT EBX
        NEG EBX
        ADD EBX, offset A+26
        OUTSTR EBX
        NEWLINE
        EXIT
END start
```

4.21

```
INCLUDE settings.inc
INCLUDE io2020.inc
.STACK 4096

.DATA
k DD ?
WDay DB 'ПН',0,'ВТ',0,'СР',0,'ЧТ',0,'ПТ',0,'СБ',0,'ВС',0

.CODE
start:
        ININT k
        DEC k
        MOV EAX, k
        ADD EAX, EAX
        ADD EAX, k      ; 3(k-1) - адрес начала дня недели в WDay
        ADD EAX, offset WDay
        OUTSTR EAX
        EXIT
END start
```

5. Команды перехода

5.8

```
X DB ?
Y DB ?
Z DB ?
```

a)

```
MOV EAX, -3
CMP X, 2
JGE NO
CMP Y, 4
JGE NO
CMP Z, 10
JGE NO
MOV EAX, 102
```

NO:

6)

```
MOV EAX, 102
CMP X, 2
JL YES
CMP Y, 4
JL YES
CMP X, 2
JL YES
MOV EAX, -3
```

YES:

5.10

X DQ ?

S DB ?

Знак числа содержится в его старшем двойном слове. Если старшее двойное слово не равно нулю, оно однозначно определяет знак всего четверного слова. Если старшее двойное слово равно нулю, возможны две ситуации: всё четверное слово равно нулю, либо четверное слово – это небольшое положительное число, целиком уместившееся в младшем двойном слове. Реализуем алгоритм:

```
S:=1;
if dword(X+4) < 0 then S:=-1
else if dword(X+4) = 0 then
    if dword(X) = 0 then S:=0
```

Решение:

```

MOV S, 1
CMP dword ptr X+4, 0
JGE GR_EQ
MOV S,-1
JMP FIN
GR_EQ: JNE FIN ; dword ptr (X+4)>0, т.е. X>0
CMP dword ptr X, 0
JNE FIN
MOV S, 0
FIN:

```

5.17 ; EDX:EAX:= X
 MOV EAX, dword ptr X
 MOV EDX, dword ptr X+4
 ; реализация CMP X, Y
 SUB EAX, dword ptr Y
 SBB EDX, dword ptr Y+4 ; JL для чисел со знаком

5.19. а) Алгоритм

```

k:= 0;
repeat
    {обработка младшей цифры}
    k:= k+1;
    {удаление младшей цифры из n}
    n:= n div 10;
until n=0
Реализация
N      DD ?
    . . .
    MOV CL, 0      ; k
    MOV EBX, 10
    MOV EAX, N
CDig: INC CL
    MOV EDX,0
    DIV EBX
    CMP EAX, 0
    JNE CDig

```

5.21 Представим N в виде произведения простых множителей $N = 2^k \cdot 3^l \cdot 5^m \cdot \dots$. Вычеркнем из N все множители-тройки, подсчитывая их. Если в итоге получим частное 1, то N является степенью числа 3.

Алгоритм

```

k:= 0;
while n mod 3 = 0 do
    begin n:= n div 3; k:= k+1 end;
if n ≠ 1 then k:= -1

```

Реализация

При вычислении предусловия ($n \bmod 3$) значение n теряется, но оно требуется для проверки после цикла. Поэтому перед делением его надо сохранить.

```

N      DD ?
K      DB ?

    .      ININT N
          MOV CL, 0           ; k
          MOV EBX, 3
          MOV EAX, N
; предусловие
Cnt3: MOV ESI, EAX
      MOV EDX, 0
      DIV EBX
      CMP EDX, 0
      JNE Check
; тело цикла
      INC CL
      JMP Cnt3

Check: CMP ESI, 1
      JE FIN
      MOV CL, -1
FIN:  MOV K, CL

```

- 5.29.** Читаем входную последовательность символов до пробела, формируя число как многочлен по схеме Горнера.

Алгоритм

```

n:= 0; read (c);
repeat n := n*5 + ord(c)-ord('0');
       read (c)
until c =

```

Реализация

```

MOV EBX, 5
MOV EAX, 0      ; n
MOV ECX, 0
INCHAR CL      ; c
Inp: MUL EBX
ADD EAX, ECX
SUB EAX, '0'
INCHAR CL
CMP CL,
JNE Inp

```

- 5.34.** Алгоритм

```

{const n=130;}
max:= {минимальное машинное число};
for i:= 1 to n do
begin read(a);
      if a > max then max:= a
end

```

Реализация

Управление **for** проще всего реализуется с помощью команды LOOP. Ниже дан фрагмент программы.

```

N = 130
MOV EBX, 0      ; max. Для (б) число = 8000 0000h
MOV ECX, N
CMax: ININT EAX      ; a
      CMP EAX, EBX
      JBE next      ; JLLE для (б)
      MOV EBX, EAX
next: LOOP CMax
      OUTU EBX      ; OUTI для (б)

```

5.41. а) Алгоритм

```

read(a); gr:= false;
repeat
    gr:= a>100;
    read(a)
until (a=0) or gr;
if gr then CL:=1 else CL:=0

```

Реализация

На ассемблере для организации управления не используются логические переменные, вместо этого применяются команды условных переходов. Следует обратить внимание на выбор удобного начального значения CL.

```

MOV CL, 1
ININT EAX
Fnd: CMP EAX, 100
      JG FIN
      ININT EAX
      CMP EAX, 0
      JNE Fnd
; все числа ≤100
      MOV CL, 0
FIN:

```

5.44. Y DB 'Является',0
 N DB 'Нет',0
 MOV EBX, offset N
 ININT EAX
 CMP EAX, 0
 JE YES
 JG NGT
 POS: ININT EAX
 CMP EAX, 0
 JL NO
 JE YES
 NGT: ININT EAX
 CMP EAX, 0
 JG NO
 JNE POS
 YES: MOV EBX, offset Y
 NO: OUTSTR EBX

6. Массивы

6.1. Неправильные: б, г, д, е, з, и, к, л, н, п, т, у

6.16. в) ; просмотр массива от конца к началу

```

MOV EAX, 7FFF FFFFh ; самое большое число со знаком
MOV ECX, length X
MOV EBX, ECX
FMin: CMP EAX, X[(type X)*ECX]-(type X)
      JGE CONT
      MOV EAX, X[(type X)*ECX]-(type X)
      MOV EBX, ECX
CONT: LOOP FMin
; индекс = EBX-1
DEC EBX

```

6.19. а)

```

MOV EBX, 0
MOV ECX, length A
FNeg: CMP A[EBX], 0
      JL FND
      ADD EBX, type A
      LOOP FNeg
; нет отрицательных
      JMP FIN
FND: MOV A[EBX], 1
FIN:

```

- 6.27. а) Реализуем алгоритм

```
i:=10;
repeat
    d:=символ(n mod 10);
    S[i]:=d;
    i:=i-1;
    n:=n div 10
until n=0
```

Ассемблер

```
MOV EAX,N
MOV ECX,10
MOV EBX, size S-type S
                ; = 9, адрес посл.элемента
DGT: MOV EDX,0
      DIV ECX
      ADD DL,'0'
      MOV S[EBX],DL
      DEC EBX
      CMP EAX,0
      JNE DGT
```

- 6.28. а) Используем массив

a: array ['a'..'z'] of 0..1;

При чтении текста заполняем массив. Когда ввод текста закончен, просуммируем элементы массива – это будет ответ.

Ассемблер

```
A DB 'z'-'a'+1 DUP (0)
.....
MOV EBX,0
INCHAR BL
LTR: MOV A[EBX-'a'],1
      INCHAR BL,
      CMP BL,'.'
      JNE LTR
      MOV EBX,'a'
      MOV ECX,'z'-'a'+1
      MOV EAX,0
CNT: ADD AL,À[EBX-'a']
      INC EBX
      LOOP CNT
OUTU EAX
```

- 6.32. б) MOV ECX, m

```
MOV EBX, 0           ; B[1,1]
PrC: OUTU B[EBX]
NEWLINE
ADD EBX, n*(type B) ; на новую строку
LOOP PrC
```

7. Структуры

- 7.1. б) 1) type D1 = size D1 = size DATE = 15

2) type D1.Y = 2; type D1.D = 1; type D1.WD = 1

3) length D1.Y = 1; length D1.WD = 1; size D1.Y = 2; size D1.WD = 1

4) offset D1.D = 103h

- 7.3. б)

```
N=70
T TIME N DUP (<>)
...
MOV EBX,0
MOV ECX, length
FND: CMP (T[EBX]).H, 12
      JB NEXT
      JA PRINT
      CMP (T[EBX]).M, 0      ; час=12
      JNE PRINT
      CMP (T[EBX]).S, 0      ; час=12, мин=0
      JNE PRINT
NEXT: ADD EBX, type T
      LOOP FND
      ; нет подходящих элементов
      JMP FIN
PRINT: MOZX EAX, (T[EBX]).H
      OUTU EAX
      OUTCHAR ':'
      MOZX EAX, (T[EBX]).M
      OUTU EAX
      OUTCHAR ':'
      MOZX EAX, (T[EBX]).S
      OUTU EAX
FIN:
```

8. Команды работы с битами

- 8.3.** Логические команды работают с битами, реализуя традиционные логические операции. Поэтому если выбрать конкретный бит в качестве представления логического значения, получим корректные логические операции. Например, пусть логическое значение байта хранится в младшем разряде. Тогда любое нечётное число будет изображать истину, а любое чётное число — ложь.

8.8. в) N=200

```
X DD N DUP(?)
    XOR EAX, EAX
    MOV ECX, length X
Cnt: MOV EBX, X[4*ECX - 4]
    SHL EBX, 1           ; CF=знак x[i]
    ADC EAX, 0
    LOOP Cnt
```

г) N=200

```
X DD N DUP(?)
    XOR EAX, EAX
    MOV ECX, length X
Cnt: MOV EBX, X[4*ECX - 4]
    SHL EBX, 1
    ADC EAX, -1          ; (-1)+CF=-1 для x[i]>=0 и (-1)+CF=0 для x[i]<0
    LOOP Cnt
    NEG EAX
```

Другое решение:

```
XOR EAX, EAX
MOV ECX, length X
Cnt: MOV EBX, X[4*ECX - 4]
    SHL EBX, 1
    SBB EAX, -1          ; EAX-(-1)-CF ≡ EAX+1-CF
    LOOP Cnt
```

8.10. Алгоритм:

```
{c:char; n:integer}
n:=0; read(c);
repeat
  n := n*4;
  n := n + ord(c)-ord('0')
  read(c)
until c=''
```

Ассемблер:

```
XOR EBX, EBX
XOR EAX, EAX
INCHAR BL
CALC: SHL EAX, 2      ; EAX*4
      SUB BL, '0'
      ADD EAX, EBX
      INCHAR BL
      CMP BL, ','
      JNE CALC
```

8.11. Алгоритм

```
На хранение четверичной цифры тратится два разряда, поэтому двойное слово n содержит 32/2=16 цифр n = d1d2 ... d16
{d: char; n, i: integer}
for i := 1 to 16 do
begin d := старшая цифра(n);
  сдвиг n << 2;
  write(chr(d + ord('0')))
end
```

Ассемблер

```
MOV ECX, 16
Prn: ROL EAX, 2
; EAX=d2 ... d16d1 на 1-ом шаге, d1 - в AL
MOV BL, 3h
; маска для выделения 2 битов
AND BL, AL ; BL=d1
ADD BL, '0'
OUTCHAR BL
LOOP Prn
; после 16 поворотов EAX вернул
; начальное значение
```

8.14. а) XOR DH, DH
 MOV ECX, 32
 Cnt1: ROL EAX, 1 ; CF = старш.р-д
 ADC DH, 0 ; DH := DH + CF
 LOOP Cnt1
 б) XOR EBX, EBX
 MOV ECX, 32
 Inv: ROL EAX, 1
 RCR EBX, 1
 LOOP Inv
 MOV EAX, EBX

8.15. Рекомендация: записать в EBX перевёрнутое значение EAX (см. 8.14(б)) и сравнить EAX и EBX.

8.20. MOV ECX, N-1 ; x[0] сдвигать на 0 нет смысла
 Shft: ROL X[(type X)*ECX], CL
 LOOP Shft

8.21. 5) X DD ?
 Y DD ?
 ;35*X=32*X+2*X+X
 MOV EAX, X
 MOV EBX, EAX
 SHL EBX, 1 ; EBX=2*X
 ADD EAX, EBX ; EAX=X+2*X
 SHL EBX, 4 ; EBX=16*(2*X)
 ADD EAX, EBX ; EAX=X+2*X+32*X
 MOV Y, EAX

8.25. а) ; C:={ '2', '4', '5' }
 MOV C, 2Ch
 б) ; C:=A ∪ B
 MOV AX, A
 OR AX, B
 MOV C, AX
 в) ; C:=A \ B=A ∩ $\overline{(A \cap B)}$
 MOV BX, A
 AND BX, B
 NOT BX
 MOV AX, A
 AND AX, BX
 MOV C, AX
 г) ; распечатать множество C
 MOV BL, '0' ; текущий элемент
 MOV ECX, 10 ; всего элементов
 MOV AX, C
 PrEl: SHL AX, 1
 JNC NEXT
 OUTCHAR BL
 NEXT: INC BL
 LOOP PrEl

9. Записи

9.1. 3)

	R2	M	N	L
width	27	5	10	12
mask	7FF FFFFh	7C0 0000h	3F F000h	0FFFh
значение поля		22	12	0

9.2

	A	B	C
a) V1 R <>	0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1	2 0	3FFh
b) V2 R <1,2,3>	0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 1	1 2	3
c) V3 R <,7>	0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1	2 0	7
d) V4 R <7>	1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1	7 0	3FFh

9.6. 6] DATE RECORD Y:7=0, M:4=0, D:5=0

```

D1 DATE <>
; type D1 = word

; печать D
    MOV EAX, D1
    AND EAX, mask D
    OUTU EAX
    OUTCHAR '.'

; печать M
    MOV EAX, D1
    AND EAX, mask M
    SHR EAX, M
    OUTU EAX
    OUTCHAR '.'

; печать Y
    MOV EAX, D1
    SHR EAX, Y
    ADD EAX, 2000
    OUTU EAX

[5] D1 DATE <>
Mnth DB 'янв',0, 'фев',0, 'мар',0
DB 'апр',0, 'май',0, 'июн',0
DB 'июл',0, 'авг',0, 'сен',0
DB 'окт',0, 'ноя',0, 'дек',0
; печать D
    MOV EAX, D1
    AND EAX, mask D
    OUTU EAX
    OUTCHAR '.'

; печать M, адрес названия в Mnth = 4(M-1)
    MOV EAX, D1
    AND EAX, mask M
    SHR EAX, M           ; EAX = M
    DEC EAX
    SHL EAX, 2
    ADD EAX, offset Mnth ; или LEA EAX, Mnth[4*EAX-4]
    OUTSTR EAX
    OUTCHAR '.'

; печать Y
    MOV EAX, D1
    SHR EAX, Y
    ADD EAX, 2000
    OUTU EAX

```

10. Стек

```

10.5 INCLUDE settings.inc
INCLUDE io2020.inc
.STACK 4096
.CODE
Start: INTN EAX
        MOV ECX, EAX
ROWS:  PUSH ECX
        MOV ECX, EAX

```

```
ELEMS: OUTCHAR '*'
        LOOP ELEMS
        NEWLINE
        POP ECX
        LOOP ROWS
        EXIT
END Start
```

10.11. 6) Алгоритм

```
{push(n);}
if n<0 then
begin write('>'); n:=-n end;
k:=0;
repeat
d := n mod 10;
push(d); k:=k+1;
n := n div 10
until n=0;
for i:=1 to k do
begin
pop(d);
write(chr(d+ord('0')))
end
{pop(n)}
```

Ассемблер

```
PUSH EAX
CMP EAX, 0
JGE CONT
OUTCHAR '-'
NEG EAX
CONT: MOV ECX, 0
MOV EBX, 10
DGTS: MOV EDX, 0
DIV EBX
PUSH EDX
INC ECX
CMP EAX, 0
JNE DGTS
PR: POP EDX
ADD DL, '0'
OUTCHAR DL
LOOP PR
POP EAX
```

11. Процедуры

11.8. а) Передача параметров в регистрах

```
; процедура
OUTB PROC
; параметр в AL=d1d2
    PUSH EAX
    PUSH ECX
    MOV ECX, 2
Pr16: ROL AL, 4
    MOV AH, 0Fh
    AND AH, AL
    CMP AH, 9
    JA LET
    ADD AH, '0'
    JMP PRINT
LET: ADD AH, 'A'-10 ; цифры 'A'..'F'
PRINT: OUTCHAR AH
    LOOP Pr16
    POP ECX
    POP EAX
    RET
OUTB ENDP
```

; Фрагмент ведущей части.
X DQ ?
; Печатаем 8 байтов переменной X,
; начиная со старшего байта
 MOV ECX, 8
PrNum: MOV AL, byte ptr X[ECX]-1
 CALL OUTB
 LOOP PrNum

11.16. а) Передача параметров в регистрах

```
SYM PROC ; ESI - начало массива, ECX - длина массива
    PUSH EDX
    PUSH ESI
    PUSH EDI
    XOR EAX,EAX ; EAX := false
    LEA EDI,[ESI][4*ECX-4] ; указатель на последний элемент
L: MOV EDX,[ESI]
    CMP EDX,[EDI]
    JNE E
```

```

ADD ESI,4
SUB EDI,4
CMP ESI,EDI
JB L
NOT EAX           ; EAX := true
E: POP EDI
POP ESI
POP EDX
RET
SYM ENDP

```

б) Передача параметров в стеке

```

; function sym(var x: массив; n: dword)
SYM  PROC
    PUSH EBP
    MOV EBP, ESP
    PUSH ECX
    PUSH EDX
    PUSH ESI
    PUSH EDI
    MOV ESI, [EBP+8]          ; адрес начала X
    MOV ECX, [EBP+12]         ; N
    LEA EDI, [ESI][4*ECX-4]   ; адрес последнего элемента X
; ECX := N div 2
    SHR ECX, 1
    JE CXZ Yes
    MOV AL, 0                 ; false
Check: MOV EDX, [ESI]
    CMP EDX, [EDI]
    JNE FIN
    ADD ESI, 4
    SUB EDI, 4
    LOOP Check
; Все элементы совпали
Yes: MOV AL, 0FFh
FIN: POP EDI
    POP ESI
    POP EDX
    POP ECX
    POP EBP
    RET 8
SYM  ENDP

```

11.28 Сформулируем алгоритм с учётом особенностей программирования управляемых конструкций на ассемблере так, чтобы его было удобно реализовать:

```

function Bits(N: integer): byte;
var t: integer;
begin t := N mod 2; N := N div 2;
      if N<>0 then t := t + Bits(N);
      Bits := t
end;
Реализация:
BITS1 PROC
    PUSH EBX
    PUSH ECX
    MOV CL,0    ; переменная t
    SHL EBX,1
    ADC CL,0
    CMP EBX,0  ; TEST EBX,EBX
    JE fin     ; JZ fin
    CALL BITS1
    ADD CL,AL
fin:  MOV AL,CL
    POP ECX
    POP EBX
    RET
BITS1 ENDP

```

12. Строковые команды

- 12.6. e) MOV ECX, N
 STD
 MOV EDI, offset S + N - 1
 MOV ESI, offset S + N - 5
 MOV EAX, dword ptr S + N - 4
 REP MOVSB
 MOV dword ptr S, EAX
- 12.10.** MOV ECX, N/2
 MOV ESI, offset S
 MOV EDI, offset S + (N+1)/2 ; если N нечётно, пропускаем средний символ
 CLD
 MOV AL, 1
 REPE CMPSB
 JE FIN
 MOV AL, 0
 FIN:
- 12.11. a) MOV ECX, N
 MOV EDI, offset S
 CLD
 MOV AL, '*'
 Srch:
 REPNE SCASB
 JNE FIN
 MOV byte ptr [EDI-1], '?'
 CMP ECX, 0
 JNE Srch
 FIN:

Если не анализировать значение ECX после строковой команды и вместо двух последних команд написать JMP SRCh, то для строки, оканчивающейся на '*', получится зацикливание.

13. Макросредства

- 13.6.** a) INC A
 JE L
- b) ADD AX,N
 ADD AX,N+1
 ADD AX,6
- c) K EQU <B KK>
 DB 'p'
 DB 'B Kp'
 LL EQU <B LLLL>
 DB 'p'
 DB 'B LLP'
 M EQU <B MM>
 DB 'p'
 DB 'B Mp'
- d) DW A,ABU,ABA
 DW B,ABU,ABB
 DW U,ABU
 DW C,CU,CC
 DW U,CU
- e) ADD AL,'"'; код sym
 ADD AL,','; код sym
 ADD AL,'%'; код sym
 ADD AL,'2'; код sym
 ADD AL,'+'; код sym
 ADD AL,'1'; код sym
- 13.16.** IF0 MACRO X,L
 LOCAL NO
 IF type X NE qword
 CMP X,0
 JE L
 ELSE
 CMP dword ptr X,0
 JNE NO
 CMP dword ptr X+4,0
 JE L
 NO:
 ENDIF
 ENDM
- 13.20.** NULL MACRO r1,r2,r3,r4,r5,r6,r7,r8
 IFNB <r1>
 MOV r1,0
 NULL r2,r3,r4,r5,r6,r7,r8
 ENDIF
 ENDM

13.32] MAX MACRO X
MOV AL,80h
FOR B,<X>
LOCAL L
CMP AL,B
JGE L
MOV AL,B
L:
ENDM
ENDM

13.37] a) JMP M1
MOV AX,1
M1:ADD AX,1
Γ) MOV AX,4
MOV BX,44

6) MOV AH,2
d) X DB '5+1'
Y5 DW 5+1

b) A EQU 3
ADD DX,A3
e) MOV AH,0
ADD AH,'6'
ADD AH,'-'
ADD AH,'1'

Приложение А. Самостоятельные работы

Самостоятельная работа №1

- Написать для трёхадресной учебной машины программу решения следующей задачи. Реализовать оператор

$$y := x * (-2) \quad (\text{второй вариант } y := x * (-3))$$

Распределение памяти: переменная x — ячейка с адресом 0100, y — 0101.

Константу разместить в ОП за последней командой программы.

Критерии оценивания: нет команды стоп — минус 2/3; не все команды полностью заполнены (есть пропущенные разряды в ячейке в команде останова) — минус 2/3; ошибка в представлении константы, использована неверная команда умножения — минус 1/3.

- Изобразить (в 16-ричном или в двоичном виде), как будут заполнены байты при выполнении следующей последовательности директив. Считать, что код символа 'A' = 41h.

1 вариант	2 вариант
X DB 1, 2, 3	A DB 'DCBA'
DB 0A5h	B DW 'DC'
Y DD 0F84h	X DB 5, 6, 7
A DB 'ABCD'	DB 0E3h
B DW 'AB'	Y DD 0A12h

Критерии оценивания: каждая ошибка — минус 1/3

Самостоятельная работа №2

1. N DD ? ; число со знаком
d DB ? ; число со знаком

Реализовать оператор

(1 вариант) $N := (N - 10) \text{ div } d$

(2 вариант) $N := (N + 10) \text{ div } d$

Считать, что величина $N - 10$ ($N + 10$) умещается в двойное слово.

Критерии оценивания: делимое не EDX:EAX (т. е. возможно переполнение частного) — минус 2/3; неверное получение из байта двойного слова, перепутано расположение частного и остатка — минус 1/3

2. X DD 30 DUP (?) ; X[0..29]

Преобразовать массив так, как преобразует его следующий оператор

for i:=0 to 28 do $X[i] := X[i] + X[29]$

Второй вариант

X DD 40 DUP (?) ; X[0..39]

Преобразовать массив так, как преобразует его следующий оператор

for i:=0 to 38 do $X[i] := X[i] - X[39]$

Критерии оценивания: каждая ошибка — минус 1/3

Самостоятельная работа №3

Задача 1

N DQ ? ; число без знака

Не используя арифметические команды, реализовать оператор

(1 вариант) N := N * 2

(2 вариант) N := N div 2

Решение должно содержать не более трёх команд.

Критерии оценивания: неверное направление сдвига, теряется разряд в середине числа (например, сдвиг не через CF) — сразу минус; не учтено перевёрнутое представление — минус 2/3; другие ошибки, каждая лишняя команда — минус 1/3.

Задача 2

1 вариант: Реализовать процедуру Padd(x,a), которая реализует оператор x:=x+a. Передача параметров в стеке, стандартные соглашения о связях.

2 вариант: Реализовать процедуру Psub(x,a), которая реализует оператор x:=x-a. Передача параметров в стеке, стандартные соглашения о связях.

Критерии оценивания: запись результата в стек, а не в сегмент данных (неумение работать с параметром-переменной) — минус; каждая другая ошибка — минус 1/3.

Приложение Б. Список директив, операторов, мнемокодов

Список директив

Название	Страница
=	8
.CODE	26
.DATA	26
.STACK	26
DB, DW, DD, DQ	8
ELSE, ELSEIF	72
END	26
ENDIF	72
ENDM	72
ENDP	60
ENDS	45
EQU	8
FOR, FORC	72
IF, IFE, IFB, IFNB, IFDIF, IFDIFI, IFIDN, IFIDNI	72
LOCAL	73
MACRO	73
PROC	60
STRUC	45
RECORD	54
REPEAT	72

Список операторов

Обозначение	Страница
+, -, *, /, mod, shl, shr	9
.	45
[]	38
&, %, !	73
offset	26, 38
ptr	15
type, length, size	38
width, mask	54

Список мнемокодов

Название	Страница
ADC	19
ADD	19
AND	48
CALL	60
CBW	19
CDQ	19
CMP	30
CWD	19
DEC	19
DIV	19
IDIV	19
IMUL	19
INC	19
JA	30
JAE	30
JB	30
JBE	30
JC	48
JE	30
JECXZ	30
JG	30
JGE	30
JL	30
JLE	30
JMP	30
JNC	48
JNE	30

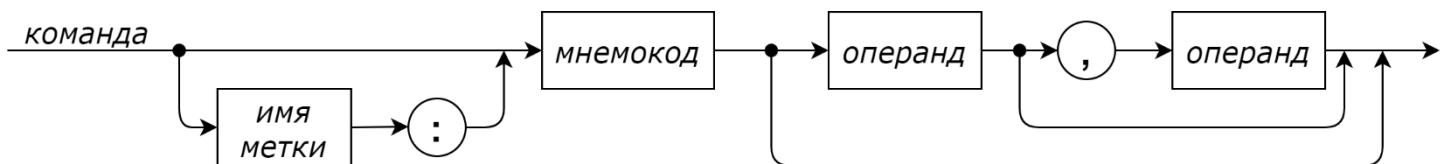
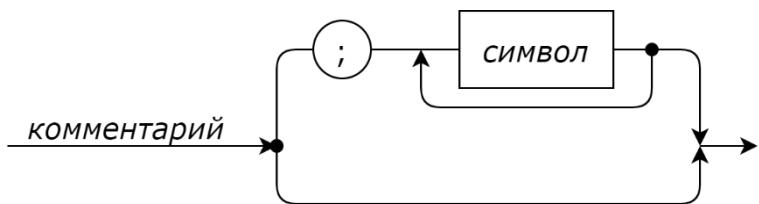
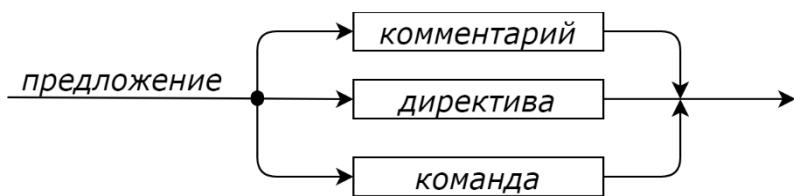
Название	Страница
JNZ	48
JZ	48
LEA	38
LOOP	30
MOV	15
MOVSX	15
MOVZX	15
MUL	19
NEG	19
NOT	48
OR	48
POP	57
POPFD	57
PUSH	57
PUSHFD	57
RCL	48
RCR	48
RET	60
ROL	48
ROR	48
SBB	19
SHL	48
SHR	48
SUB	19
TEST	48
XCHG	15
XOR	48

Приложение В. Синтаксические диаграммы конструкций MASM 6.14

Ниже приведены синтаксические диаграммы конструкций языка MASM 6.14, которые изучаются в курсе «Архитектура ЭВМ и язык ассемблера».

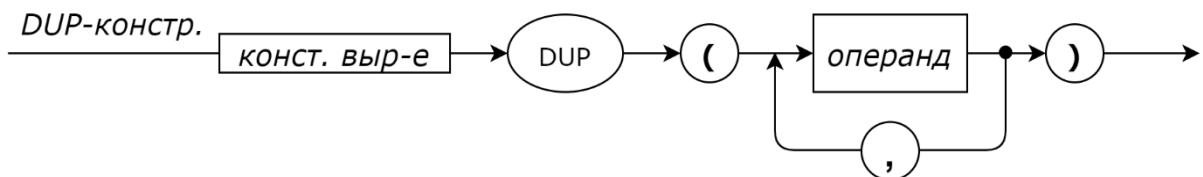
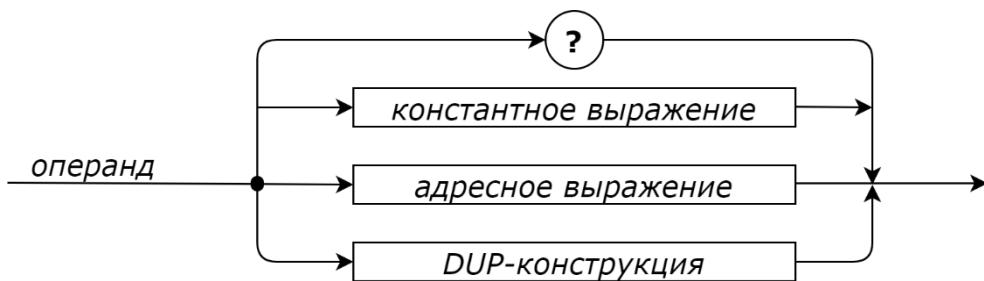
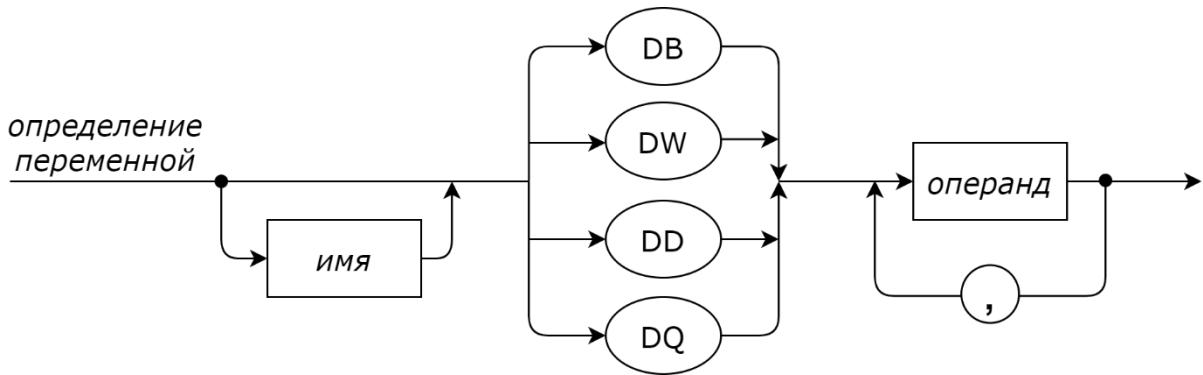


Каждое предложение записывается в отдельной строке.

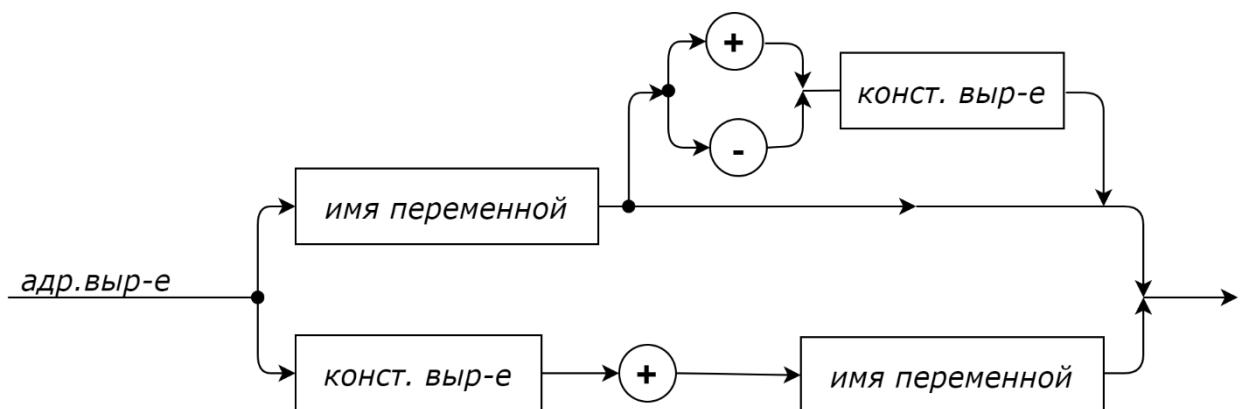


В конце директивы и команды может идти комментарий после точки с запятой.
(Не изображен на диаграммах)

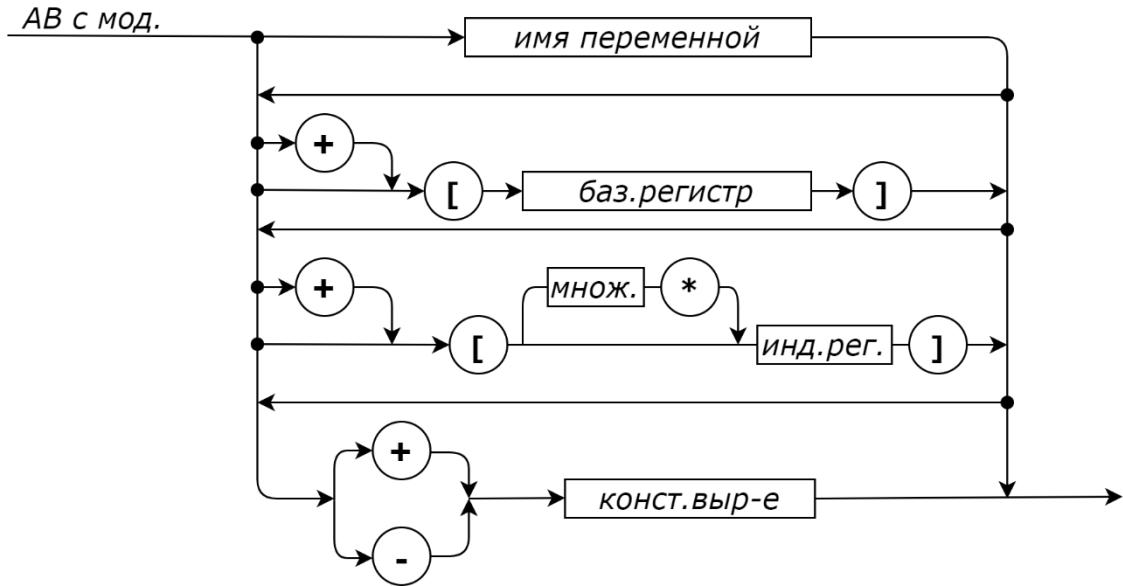
§1. Директивы определения данных.



Константное выражение без ссылок вперёд, значение больше нуля.



§6. Массивы.



Понятия базовый регистр и индексный регистр опишем БНФ

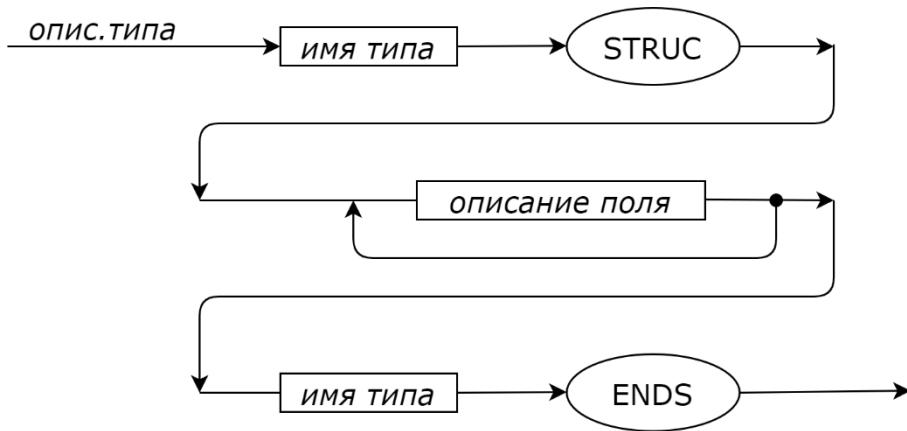
$\langle \text{инд. reg.} \rangle ::= \text{EAX} | \text{EBX} | \text{ECX} | \text{EDX} | \text{ESI} | \text{EDI} | \text{EBP}$

$\langle \text{баз. регистр} \rangle ::= \langle \text{инд. рег.} \rangle | \text{ESP}$



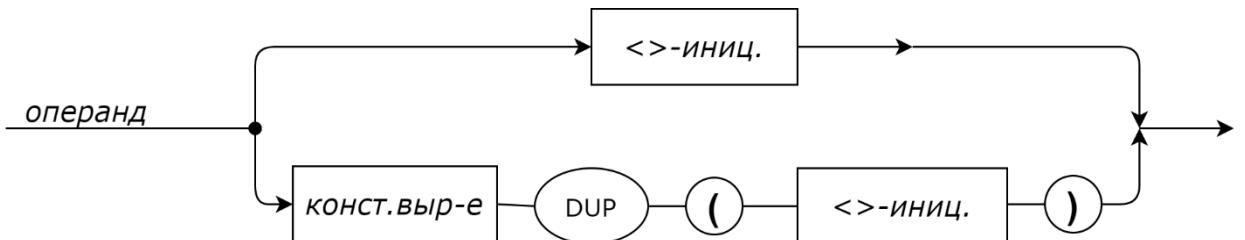
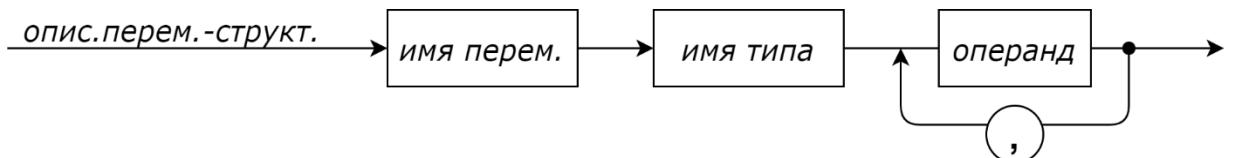
Константное выражение принимает значение 1, 2, 4 или 8.

§7. Структуры.

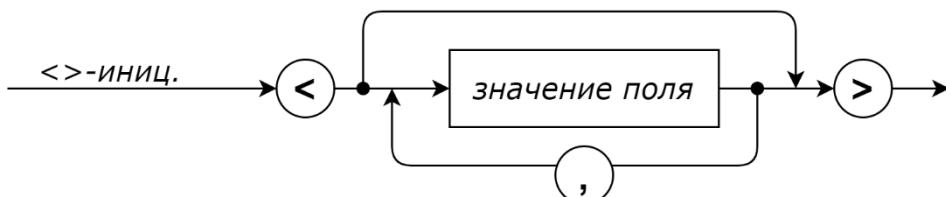


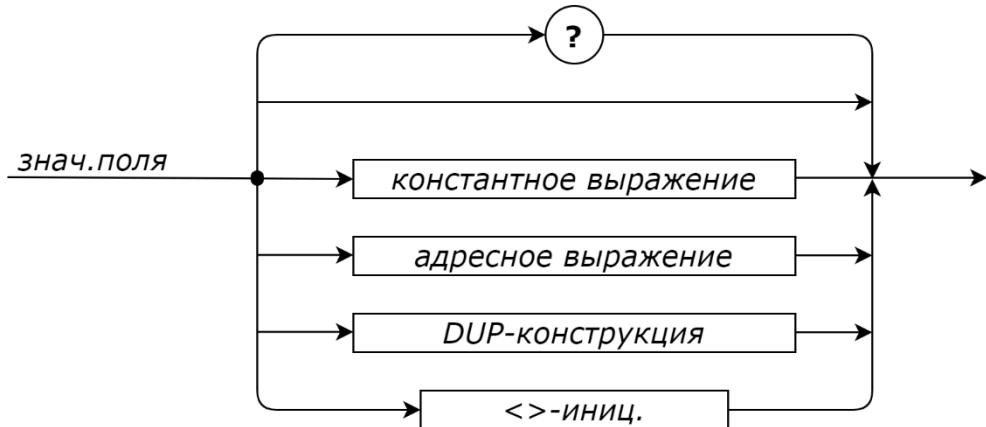
Синтаксис описания поля совпадает с синтаксисом определения переменной из §1 с одним изменением: можно использовать не только слова DB, DW, DD, DQ, но и названия типов структур и записей.

Описание типа структуры состоит из нескольких предложений, каждое – в отдельной строке.

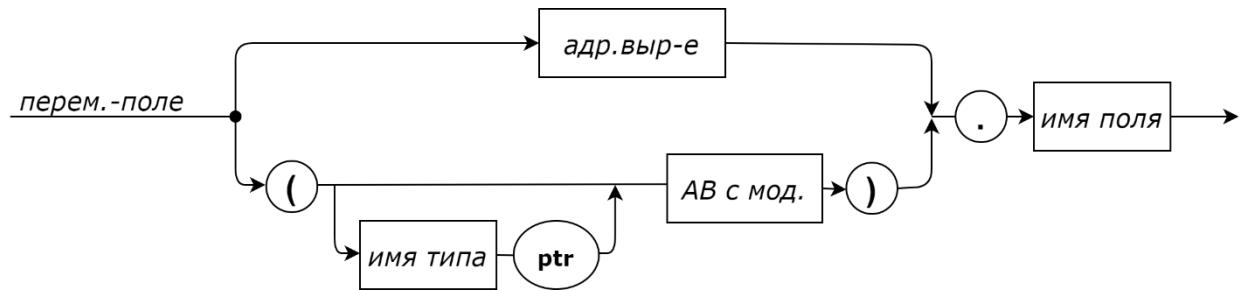


Константное выражение без ссылок вперёд, значение больше нуля.



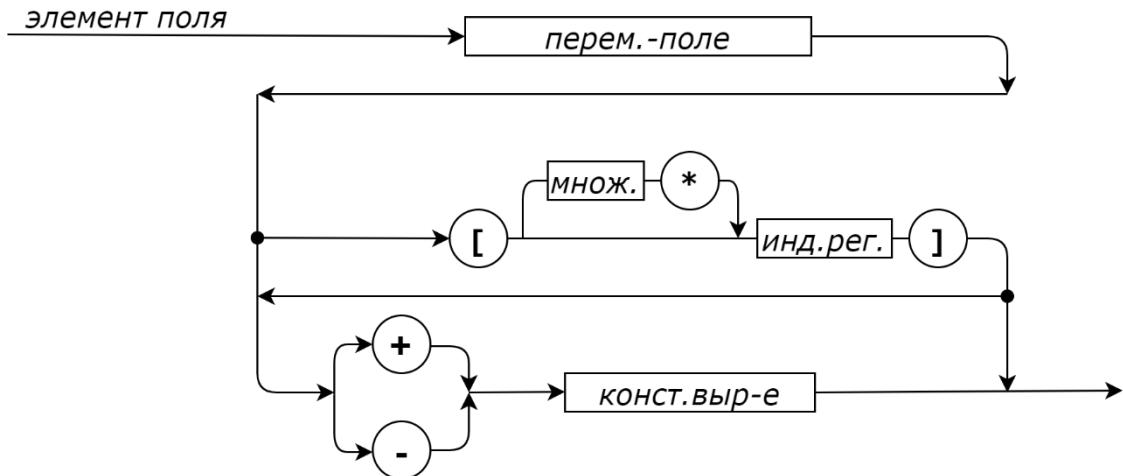


Понятия *адресное выражение* и *DUP-конструкция* описаны в §1.



Понятие *адресное выражение* описано в §1, *AB с модификаторами* – в §6.

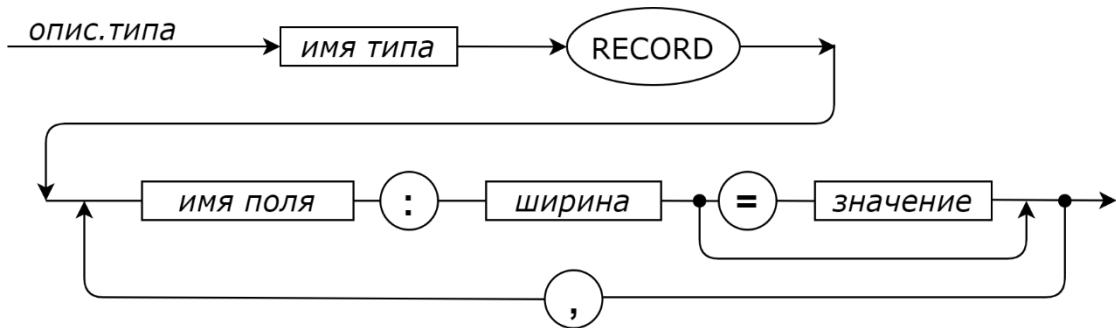
Если поле является массивом, можно дописать модификатор (см. §6):



В этом случае в переменной-поле не должно быть индексного регистра.

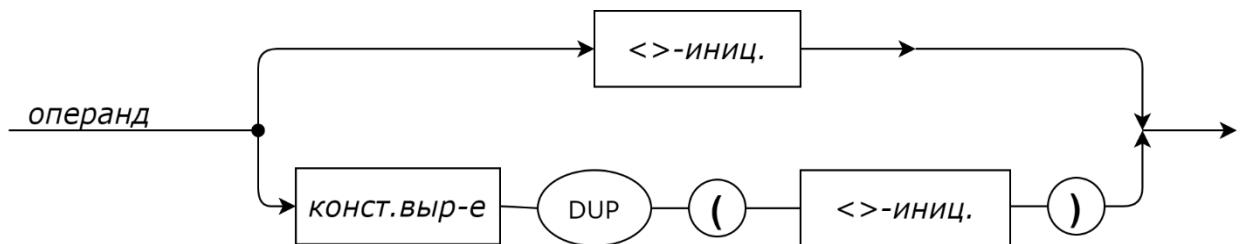
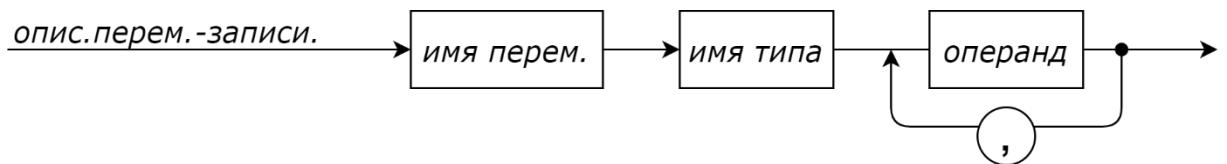
В выражении может быть указано не более двух модификаторов – базовый регистр и индексный регистр.

§9. Записи.

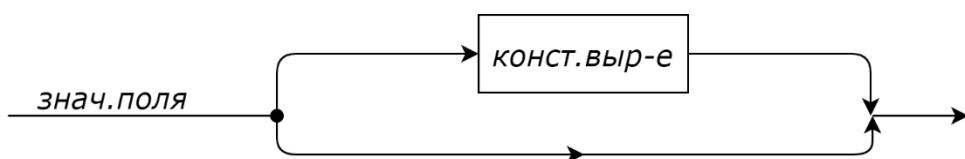
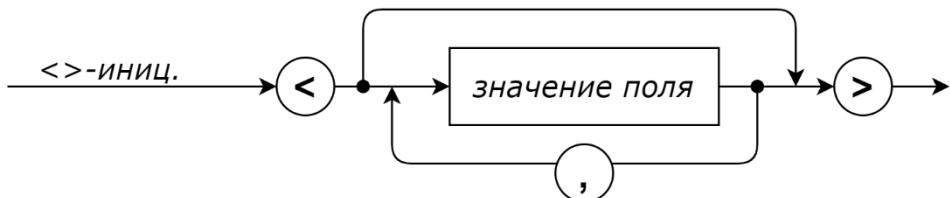


Ширина, значение – константные выражения без ссылок вперёд.

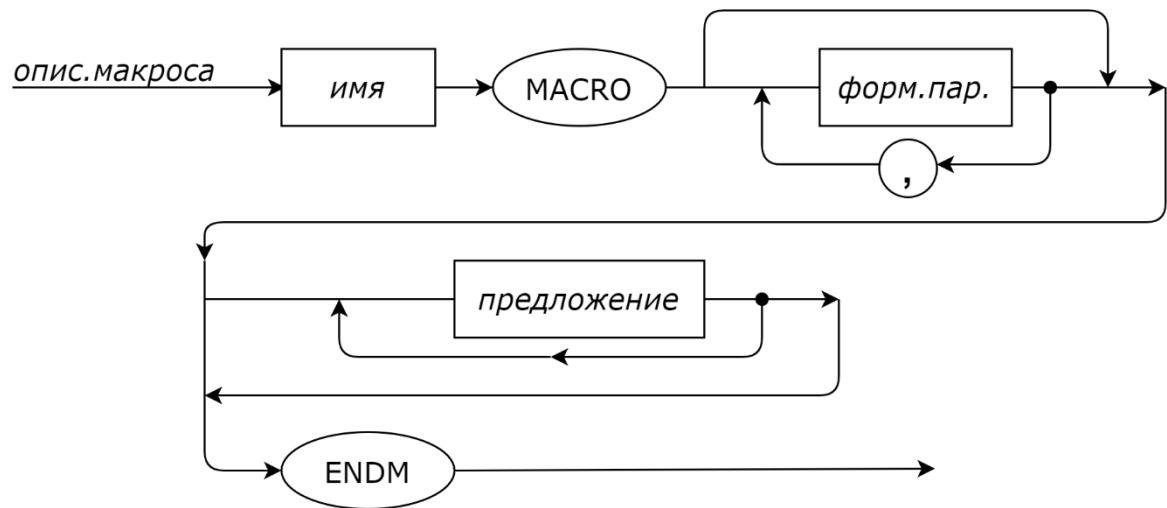
Описание типа – одно предложение, занимает одну строку.



Константное выражение без ссылок вперёд, значение больше нуля.



§13. Макросредства.



Формальный параметр – имя.

Описание макроса состоит из нескольких предложений (строк).

